SIAM J. SCI. COMPUT.
Vol. 28, No. 1, pp. 1–23

© 2006 Society for Industrial and Applied Mathematics

AN INFEASIBLE PRIMAL-DUAL ALGORITHM FOR TOTAL BOUNDED VARIATION–BASED INF-CONVOLUTION-TYPE IMAGE RESTORATION*

M. HINTERMÜLLER† AND G. STADLER‡

Abstract. In this paper, a primal-dual algorithm for total bounded variation (TV)–type image restoration is analyzed and tested. Analytically it turns out that employing a global $\boldsymbol{L}^s$-regularization, with $1 < s \leq 2$, in the dual problem results in a local smoothing of the TV-regularization term in the primal problem. The local smoothing can alternatively be obtained as the infimal convolution of the ℓ_r-norm, with $r^{-1} + s^{-1} = 1$, and a smooth function. In the case $r = s = 2$, this results in Gauss-TV–type image restoration. The globalized primal-dual algorithm introduced in this paper works with generalized derivatives, converges locally at a superlinear rate, and is stable with respect to noise in the data. In addition, it utilizes a projection technique which reduces the size of the linear system that has to be solved per iteration. A comprehensive numerical study ends the paper.

Key words. Fenchel duality, generalized Newton-type methods, image restoration, total bounded variation

AMS subject classifications. 94A08, 49M29, 65K05

DOI. 10.1137/040613263

1. Introduction.

Image denoising based on total bounded variation regularization (TV regularization) was introduced in [26]. Since then it has received a considerable amount of attention and is widely accepted as a reliable tool in edge preserving image restoration; see the selected papers [1, 4, 5, 6, 9, 10, 14, 23, 29] and the recent monograph [32] for further references.

Compared to classical H^1-type regularization, such as $\int_\Omega |\nabla u|^2_{\ell^2}\, dx$ with u representing the reconstructed image, or other techniques based on smooth (at least C^1) functionals, the TV term $\int_\Omega |Du|$ is nondifferentiable in the Fréchet sense. This nondifferentiability is responsible for preserving edges in images, but at the same time, it poses significant numerical challenges. Early computational techniques employed global smoothing of the bounded variation (BV) seminorm to overcome the nondifferentiability [9, 33]; see also the more recent work [2]. In many cases a C^∞-function like $\int_\Omega (|\nabla u|^2_{\ell^2} + \epsilon)^{1/2}\, dx$, with $\epsilon > 0$ fixed, is used as a smooth approximation of the nondifferentiable TV term. For small smoothing parameter ϵ, this technique introduces severe ill-conditioning of second derivatives and adversely affects, e.g., Newton's method for solving the first order conditions associated with the smoothed TV-type problem. To remedy this ill-conditioning, in [6] a (dual) flux variable w is introduced together with the requirement that the iterates w^k stay in the interior of the unit ball (strict feasibility) in the pointwise almost everywhere sense. Another type of global regularization is introduced in the primal affine scaling technique in [21], where

*Received by the editors August 11, 2004; accepted for publication (in revised form) February 22, 2005; published electronically March 3, 2006. This work was supported in part by the "Fonds zur Förderung der wissenschaftlichen Forschung" under "SRC 03, Optimization and Control."
http://www.siam.org/journals/sisc/28-1/61326.html

†Department of Computational and Applied Mathematics–MS 134, Rice University, 6100 Main St., Houston, TX 77005 (hint@caam.rice.edu).

‡Institut für Mathematik und Wissenschaftliches Rechnen, Karl-Franzens-Universität Graz, Heinrichstraße 36, A-8010 Graz, Austria (ge.stadler@uni-graz.at).

the interior point treatment acts like a global regularization within a continuation framework.

As the above-mentioned smoothing of the BV seminorm introduces [illegible] global nonlinearity to the problem, several authors studied local smoothing [5, 10, 28]; see [31] for further references. The idea is to keep the original problem formulation to the largest possible extent. While the above remarks on ill-conditioning persist, in the local case the BV seminorm is changed only close to kinks. The utilized functionals are typically only once continuously differentiable as opposed to the C^∞-type global smoothing.

Yet another approach that avoids global smoothing is based on active set techniques; see [15, 17, 18] for closely related algorithms. Utilizing augmented Lagrangian formulations, these methods predict the set of points where the BV seminorm is nondifferentiable, fix the gradient of the primal variable on this set, and compute it on the complement set. It turns out that the original problem is approximated by a sequence of problems which are almost as difficult to solve as the original one. In [15] this drawback is overcome by using a penalization technique, which, to some extent, acts like an additional regularization. In [18] a heuristic lumping technique is applied for solving the auxiliary problems. In numerical practice these active set-type methods are significantly faster than the widely used gradient descent schemes [26] or fixed point–type iterations [9, 33].

In this paper, we adopt the Hilbert space perspective of [15, 17, 18]; i.e., we consider the problem

$$(\mathcal{P}) \qquad \min_{u\in H_0^1(\Omega)} \frac{\mu}{2}\int_\Omega |\nabla u|_2^2\,dx + \frac{1}{2}\int_\Omega |Ku-z|^2\,dx + \alpha\int_\Omega |\nabla u|_r\,dx,$$

where $\Omega\subset\mathbb{R}^2$ is an open, bounded (image) domain with Lipschitz-continuous boundary $\partial\Omega$, $K : L^2(\Omega)\to L^2(\Omega)$ is a continuous linear operator with K^* its adjoint, $z\in L^2(\Omega)$ represents given noisy data, $\mu,\alpha>0$, and $|\nabla u|_r = (|u_x|^r+|u_y|^r)^{1/r}$, with $1\le r<+\infty$. Typically $0<\mu\ll\alpha$ such that $(\mathcal{P})$ is a close approximation to the TV-regularized problem

$$(\mathcal{P}_{BV}) \qquad \min_{u\in \mathrm{BV}(\Omega)} \frac{1}{2}\int_\Omega |Ku-z|^2\,dx + \alpha\int_\Omega |Du|_r,$$

where $\mathrm{BV}(\Omega)$ denotes the space of functions of bounded variation; i.e., a function u is in $\mathrm{BV}(\Omega)$ if the BV seminorm

$$\int_\Omega |Du|_r = \sup\left\{\int_\Omega u\,\mathrm{div}\ \vec{v}\,dx : \vec{v}\in(C_0^\infty(\Omega))^2, |\vec{v}(x)|_s\le 1\right\},$$

with $r^{-1}+s^{-1}=1$, is finite. We recall that $\mathrm{BV}(\Omega)\subset L^2(\Omega)$ for $\Omega\subset\mathbb{R}^2$ [12]. For the discretized version of $(\mathcal{P})$ and as long as K^*K is invertible, we may even choose $\mu=0$ and obtain stable results; see section 5. A particular instance with K^*K invertible is given by image denoising where $K=\mathrm{id}$.

The aim of the present paper is to devise an implementable algorithm which is based on a primal-dual approach, overcomes the difficulties of the methods in [15, 17, 18], is locally rapidly convergent, and, in contrast to, e.g., [6], takes advantage of local regularization only. The primal-dual framework is obtained by considering the Fenchel dual to $(\mathcal{P})$, which turns out to be a bound-constrained concave quadratic maximization problem. The objective function of the dual problem involves the divergence operator, which is responsible for the solution set not to be a singleton.

To avoid this nonuniqueness property, we propose a simple $\boldsymbol{L}^s$-regularization, with $1 < s \leq 2$ and $\boldsymbol{L}^s(\Omega) = (L^s(\Omega))^2$, such that the resulting regularized dual problem admits a unique solution. Let $\gamma > 0$ denote the corresponding regularization parameter. Then, in the context of the primal problem, for $w \in \mathbb{R}^n$ this regularization is given by the inf-convolution

$$\phi_{r,\gamma}(w) = \inf\{|\hat{w}|_r + (r\gamma)^{-1}|w - \hat{w}|_r^r : \hat{w} \in \mathbb{R}^n\}.$$

As a consequence, the regularized version of $(\mathcal{P})$ involves the term $\int_\Omega \Phi_{r,\gamma}(\nabla u)\,dx$, with $\Phi_{r,\gamma}(\nabla u)(x) = \phi_{r,\gamma}(\nabla u(x))$, rather than $\int_\Omega |\nabla u|_r\,dx$. In the special case $s = 2$ we obtain Gauss-TV regularization [28], which was recently shown to be an excellent choice for medical imaging [20]. The latter statement is mainly based on the fact that the Gauss-TV regularization employs a local smoothing of $|\cdot|_2$ and allows avoidance of the adverse staircase or bulkiness effect well known in pure TV regularization; see, e.g., [3]. Similar effects are obtained for $1 < s < 2$. In our model, the choice of γ influences the amount of local smoothing. Indeed, for small γ one finds that $\Phi_{r,\gamma}$ is close to the BV seminorm, whereas for large γ, strong smoothing of $|\cdot|_r$ takes place. For further techniques to reduce the staircase effect we refer the reader to [19].

The rest of the paper is organized as follows. In section 2 we derive the Fenchel dual problem of $(\mathcal{P})$, introduce the dual regularization, and discuss some of its aspects. The primal-dual algorithm is defined in section 3, where we also provide a discussion of global as well as fast local convergence. In section 4 we consider aspects of the implementation, and in section 5 we report on our numerical experience with the new algorithm, validate the theoretical results, and we study the qualitative behavior of our dual regularization technique.

Throughout the paper we shall frequently use the following notation. For a vector $v \in \mathbb{R}^n$, and $r \in \mathbb{R}$ we define $|v|^r := (|v_1|^r, \ldots, |v_n|^r)^\top$ and $\sigma(v) \in \mathbb{R}^n$ with $\sigma(v) = (\operatorname{sign}(v_1), \ldots, \operatorname{sign}(v_n))^\top$. We use $|\cdot|_s$ for the ℓ_s-vector norm in $\mathbb{R}^n$. Further we denote by $\star$ the Hadamard product of vectors, i.e., $v \star w = (v_1 w_1, \ldots, v_n w_n)^\top$. For vector-valued functions we write $\vec{p}$ and $p(x) \in \mathbb{R}^n$ for its value at x. For a symmetric matrix $M \in \mathbb{R}^{m,m}$, $\lambda_{\min}(M)$ denotes its smallest eigenvalue. In the case when M is nonsymmetric, $\lambda_{\min}(M)$ is the smallest eigenvalue of $\frac{1}{2}(M + M^\top)$, the symmetrization of M.

2. The Fenchel dual of $(\mathcal{P})$ and its regularization. For the reader's convenience we now recall the Fenchel duality theorem in infinite-dimensional spaces; see, e.g., [11]. For this purpose let V and Y be Banach spaces with topological duals V^* and Y^*, respectively. Further, let $\Lambda \in \mathcal{L}(V, Y)$ be the space of continuous linear operators from V to Y, and let $\mathcal{F} : V \to \mathbb{R} \cup \{\infty\}$, $\mathcal{G} : Y \to \mathbb{R} \cup \{\infty\}$ be convex, proper, and lower semicontinuous such that there exists $v_0 \in V$ with $\mathcal{F}(v_0) < \infty$, $\mathcal{G}(\Lambda v_0) < \infty$, and $\mathcal{G}$ continuous at Λv_0. Then

$$\inf_{v \in V} \{\mathcal{F}(v) + \mathcal{G}(\Lambda v)\} = \sup_{q \in Y^*} \{-\mathcal{F}^*(\Lambda^* q) - \mathcal{G}^*(-q)\}, \tag{2.1}$$

where $\Lambda^* \in \mathcal{L}(Y^*, V^*)$ is the adjoint of Λ. The convex conjugates $\mathcal{F}^* : V^* \to \mathbb{R} \cup \{\infty\}$, $\mathcal{G}^* : Y^* \to \mathbb{R} \cup \{\infty\}$ of $\mathcal{F}$ and $\mathcal{G}$, respectively, are defined by

$$\mathcal{F}^*(v^*) = \sup_{v \in V} \{\langle v, v^* \rangle_{V,V^*} - \mathcal{F}(v)\},$$

and analogously for $\mathcal{G}^*$. The conditions imposed on $\mathcal{F}$ and $\mathcal{G}$ guarantee that the dual problem (which is the problem on the right-hand side of (2.1)) admits a solution.

Furthermore, the solutions $\bar v \in V$ and $\bar q \in Y^*$ are characterized by the optimality conditions

$$\begin{aligned} \Lambda^* \bar q &\in \partial \mathcal{F}(\bar v), \\ -\bar q &\in \partial \mathcal{G}(\Lambda \bar v), \end{aligned} \tag{2.2}$$

where ∂ denotes the subdifferential from convex analysis.

2.1. Fenchel dual of ($\mathcal{P}$). Now we apply the Fenchel calculus to

$$\begin{aligned} \mathcal{F}(u) &:= \frac{1}{2}\int_\Omega |Ku - z|^2\, dx + \frac{\mu}{2}\int_\Omega |\nabla u|_2^2 dx, \\ \mathcal{G}(\vec q) &:= \alpha \int_\Omega |\vec q|_r\, dx, \;\; \Lambda u := (u_x, u_y)^\top, \end{aligned}$$

with $V := H_0^1(\Omega)$, $Y := \boldsymbol{L}^2(\Omega)$, $1 \le r < +\infty$. As a result we obtain

$$\begin{aligned} \mathcal{F}^*(u^*) &:= \frac{1}{2}|||u^* + K^* z|||_{H^{-1}}^2 - \frac{1}{2}\|z\|_{L^2}^2, \\ \mathcal{G}^*(\vec q^*) &:= I_{\{\vec w \in \boldsymbol{L}^2(\Omega) : |\vec w|_s \le \alpha \text{ a.e. in } \Omega\}}(\vec q^*), \end{aligned}$$

where $r^{-1} + s^{-1} = 1$, I_S is the indicator function of the set $S \subset \boldsymbol{L}^2(\Omega)$, and $|||v|||_{H^{-1}} = \langle H_{\mu,K} v, v\rangle_{H_0^1, H^{-1}}$, $v \in H^{-1}(\Omega)$, with $H_{\mu,K} = (K^*K - \mu\Delta)^{-1}$. Here $\langle \cdot, \cdot \rangle_{H_0^1, H^{-1}}$ denotes the duality pairing between $H_0^1(\Omega)$ and its dual $H^{-1}(\Omega)$, and $\Delta : H_0^1(\Omega) \to H^{-1}(\Omega)$.

Setting $\vec p = -\vec q$, then, according to (2.1), the dual problem of ($\mathcal{P}$) is

$$\sup_{\substack{\vec p \in \boldsymbol{L}^2(\Omega), \\ |\vec p|_s \le \alpha \text{ a.e. in } \Omega}} \; -\frac{1}{2}|||\operatorname{div} \vec p + K^* z|||_{H^{-1}}^2 + \frac{1}{2}\|z\|_{L^2}^2. \tag{$\mathcal{P}^\star$}$$

Note that the Fenchel dual of ($\mathcal{P}$) is a bound constrained maximization problem with a concave quadratic objective function. Further observe that the solution to ($\mathcal{P}^\star$) is not unique since the divergence operator has a nontrivial kernel.

From (2.2) we obtain the following characterization of solutions $\bar u$ and $\bar{\vec p}$ of ($\mathcal{P}$) and ($\mathcal{P}^\star$), respectively:

$$-\mu\Delta \bar u + K^*K\bar u - \operatorname{div} \bar{\vec p} = K^* z \;\text{ in } H^{-1}(\Omega), \tag{2.3a}$$

$$\left.\begin{aligned} -\alpha(\sigma(\nabla \bar u) \star |\nabla \bar u|^{r-1}) + |\nabla \bar u|_r^{r-1}\bar{\vec p} &= 0 \quad \text{if} \quad |\bar{\vec p}|_s = \alpha, \\ \nabla \bar u &= 0 \quad \text{if} \quad |\bar{\vec p}|_s < \alpha \end{aligned}\right\} \;\text{ in } \boldsymbol{L}^2(\Omega). \tag{2.3b}$$

In (2.3a)–(2.3b) the expression $|\nabla \bar u|_r^{r-1}$ can be interpreted as a Lagrange multiplier for the inequality constraint in the dual problem ($\mathcal{P}^\star$). We summarize the above results in the following theorem.

THEOREM 2.1. [illegible] ($\mathcal{P}$) [illegible] ($\mathcal{P}^\star$) [illegible] (2.1) [illegible] $\bar u$ [illegible] ($\mathcal{P}$) [illegible] $\bar{\vec p}$ [illegible] ($\mathcal{P}^\star$) [illegible] (2.3a)–(2.3b) [illegible] ($\mathcal{P}$) [illegible] $\bar{\vec p}^1$ $\bar{\vec p}^2$ [illegible] ($\mathcal{P}^\star$) [illegible] $\operatorname{div}(\bar{\vec p}^1 - \bar{\vec p}^2) = 0$ [illegible] $\bar{\vec p}^1 = \bar{\vec p}^2$ [illegible] $\nabla \bar u \neq 0$

With respect to the special case $\mu = 0$ we point out that

- in [15, Remark 2.1], for $K = \mathrm{id}$, it is shown that the solution $\bar u = \bar u(\mu)$ satisfies $\bar u(\mu) \rightharpoonup \bar u_{\mathrm{BV}}$ weakly in $L^2(\Omega)$ as $\mu \to 0$, where $\bar u_{BV}$ denotes the unique solution to ($\mathcal{P}_{BV}$);

- for $\mu = 0$, in [14] the Fenchel predual problem of $(\mathcal{P})$ in an ℓ_1-vector-norm setting is derived. Surprisingly it turns out that the predual of the TV problem $(\mathcal{P}_{BV})$ is posed in a Hilbert space. Its structure is similar to $(\mathcal{P}^\star)$. The underlying function space is $H_0(\mathrm{div}) = \{\vec{v} \in \boldsymbol{L}^2(\Omega) : \mathrm{div}\, \vec{v} \in L^2(\Omega),\ \vec{v} \cdot \vec{n} = 0 \text{ on } \partial\Omega\}$, where $\vec{n}$ denotes the outward unit normal to $\partial\Omega$. For a convergence analysis of the proposed semismooth Newton method, however, an H^1-type regularization of the dual problem is necessary; for details see [14, section 3].

2.2. Regularization. Nonuniqueness of the solution to the dual problem may cause instabilities in a numerical scheme utilizing primal and dual variables. As a remedy we propose the following simple Tikhonov-type regularization of $(\mathcal{P}^\star)$:

$$(\mathcal{P}^\star_\gamma) \qquad \sup_{\substack{\vec{p} \in \boldsymbol{L}^2(\Omega),\\ |\vec{p}|_s \le \alpha \text{ a.e. in } \Omega}} -\frac{1}{2}|||\mathrm{div}\, \vec{p} + K^* z|||^2_{H^{-1}} + \frac{1}{2}\|z\|^2_{L^2} - \frac{\gamma^{s-1}}{s\alpha^{s-1}}\|\vec{p}\|^s_{\boldsymbol{L}^s},$$

where $\gamma > 0$ is the regularization parameter and $1 < s \le 2$. Subtracting the term $s^{-1}\alpha^{1-s}\gamma^{s-1}\|\vec{p}\|^s_{\boldsymbol{L}^s}$ from the objective functional of $(\mathcal{P}^\star)$ results in an $\boldsymbol{L}^s$-uniformly concave objective functional in $(\mathcal{P}^\star_\gamma)$. Thus, $(\mathcal{P}^\star_\gamma)$ admits a unique solution $\vec{p}_\gamma \in \boldsymbol{L}^2(\Omega)$ for every fixed $\gamma > 0$.

Due to the reflexivity of the involved function spaces, we compute the Fenchel dual of $(\mathcal{P}^\star_\gamma)$ and study the effect of the dual regularization in the primal problem. For the Fenchel calculus, we set $\vec{p} = -\vec{q}$ and

$$\begin{aligned}\mathcal{F}(\vec{q}) &:= I_{\{\vec{w} \in \boldsymbol{L}^2(\Omega) : |\vec{w}|_s \le \alpha\}}(\vec{q}) + \frac{\gamma^{s-1}}{s\alpha^{s-1}}\|\vec{q}\|^s_{\boldsymbol{L}^s},\\ \mathcal{G}(\Lambda\vec{q}) &:= \frac{1}{2}|||K^* z - \mathrm{div}\, \vec{q}|||^2_{H^{-1}} - \frac{1}{2}\|z\|^2_{L^2},\ \ \Lambda\vec{q} := \mathrm{div}\, \vec{q},\end{aligned}$$

with $Y := H^{-1}(\Omega)$, $V := \boldsymbol{L}^2(\Omega)$. The Fenchel dual problem of $(\mathcal{P}^\star_\gamma)$ is given by

$$(\mathcal{P}_\gamma) \qquad \min_{u \in H^1_0(\Omega)} \frac{\mu}{2}\int_\Omega |\nabla u|^2\, dx + \frac{1}{2}\int_\Omega |Ku - z|^2\, dx + \alpha \int_\Omega \Phi_{r,\gamma}(\nabla u)\, dx,$$

where $r^{-1} + s^{-1} = 1$ and, for $\vec{w} \in \boldsymbol{L}^2(\Omega)$,

$$\Phi_{r,\gamma}(\vec{w})(x) := \begin{cases} |w(x)|_r - \frac{r-1}{r}\gamma^{1/(r-1)} & \text{if } |w(x)|_r \ge \gamma^{1/(r-1)},\\ \frac{1}{r\gamma}|w(x)|^r_r & \text{if } |w(x)|_r < \gamma^{1/(r-1)}.\end{cases}$$

The function $\Phi_{r,\gamma}$ is continuously differentiable. Hence, the objective functional in $(\mathcal{P}_\gamma)$ is continuously differentiable and strictly convex. Thus, $(\mathcal{P}_\gamma)$ admits a unique solution $\bar{u}_\gamma \in H^1_0(\Omega)$. Further, $\Phi_{r,\gamma}(\nabla u)$ represents a local smoothing of $\int_\Omega |\nabla u|_r\, dx$ and, as our analysis shows, it is connected to a particular regularization of the Fenchel dual of $(\mathcal{P}^\star)$. In the case $r = 2$ we have

$$\Phi_{2,\gamma}(\vec{w})(x) := \begin{cases} |w(x)|_2 - \frac{\gamma}{2} & \text{if } |w(x)|_2 \ge \gamma,\\ \frac{1}{2\gamma}|w(x)|^2_2 & \text{if } |w(x)|_2 < \gamma.\end{cases}$$

This choice is related to results in [5, p. 186] and [28], and it is known as Gauss-TV regularization. In medical imaging it is in many cases superior to the Perona–Malik filter [24], edge-flat-gray filter [16], and pure TV filtering. Let us also mention that $\Phi_{2,\gamma}$ is sometimes referred to as Huber function; see [10, 32].

By (2.2), the solutions $\bar{u}_\gamma$ and $\bar{\vec{p}}_\gamma$ of the regularized problems $(\mathcal{P}_\gamma)$ and $(\mathcal{P}_\gamma^\star)$, respectively, satisfy

$$-\mu\Delta\bar{u}_\gamma + K^*K\bar{u}_\gamma - \operatorname{div}\bar{\vec{p}}_\gamma = K^*z \ \text{ in } H^{-1}(\Omega), \tag{2.4a}$$

$$\left.\begin{array}{rll} \gamma\bar{\vec{p}}_\gamma - \alpha(|\nabla\bar{u}_\gamma|^{r-1} \star \sigma(\nabla\bar{u}_\gamma)) &= 0 \quad \text{if} \quad |\bar{\vec{p}}_\gamma|_s < \alpha, \\ |\nabla\bar{u}_\gamma|_r^{r-1}\bar{\vec{p}}_\gamma - \alpha(|\nabla\bar{u}_\gamma|^{r-1} \star \sigma(\nabla\bar{u}_\gamma)) &= 0 \quad \text{if} \quad |\bar{\vec{p}}_\gamma|_s = \alpha \end{array}\right\} \ \text{in } \boldsymbol{L}^2(\Omega). \tag{2.4b}$$

One can easily verify that (2.4b) can be equivalently expressed as

$$\max(\gamma, |\nabla\bar{u}_\gamma|_r^{r-1})\bar{\vec{p}}_\gamma - \alpha(|\nabla\bar{u}_\gamma|^{r-1} \star \sigma(\nabla\bar{u}_\gamma)) = 0. \tag{2.5}$$

This reformulation is fundamental for the algorithm which we propose in the next section.

Comparing (2.4b) and (2.3b) one finds that our regularization changes the optimality conditions only on subsets of Ω where $|\nabla\bar{u}_\gamma|_r < \gamma^{1/(r-1)}$, i.e., on sets where the pointwise constraint $|\bar{\vec{p}}_\gamma|_s \le \alpha$ is inactive. At sharp edges in the image, $|\nabla\bar{u}_\gamma|_r$ is large and (2.4b) becomes

$$|\nabla\bar{u}_\gamma|_r^{r-1}\bar{\vec{p}}_\gamma = \alpha(|\nabla\bar{u}_\gamma|^{r-1} \star \sigma(\nabla\bar{u})),$$

as it would be the case for the original problem $(\mathcal{P})$. As a consequence, our regularization does not smear sharp edges in the image. In addition, our local smoothing prevents the adverse [illegible] or [illegible] effect which typically occurs in TV regularization [16, 20]; see the discussion in section 5.2.

The following convergence result for $(\bar{u}_\gamma, \bar{\vec{p}}_\gamma)$ as $\gamma \to 0$ justifies our dual regularization analytically. The proof is given in Appendix A.

THEOREM 2.2. [illegible] $\bar{u}_\gamma$ [illegible] $(\mathcal{P}_\gamma)$ [illegible] $\bar{u}$ [illegible] $(\mathcal{P})$ [illegible] $H_0^1(\Omega)$ [illegible] $\gamma \to 0$ [illegible] $\gamma \to 0$ [illegible] $\bar{\vec{p}}_\gamma$ [illegible] $(\mathcal{P}_\gamma^\star)$ [illegible] $\bar{\vec{p}}$ [illegible] $(\mathcal{P}^\star)$ [illegible] $\boldsymbol{L}^2(\Omega)$

We conclude this section by pointing out another interesting relation between the dual regularization, $\Phi_{r,\gamma}$, and inf-convolution. In fact, $\Phi_{r,\gamma}$ can be obtained in a purely primal fashion. For this purpose consider for $w \in \mathbb{R}^n$ the inf-convolution

$$\phi_{r,\gamma}(w) := \inf\{|\hat{w}|_r + (r\gamma)^{-1}|w - \hat{w}|_r^r : \hat{w} \in \mathbb{R}^n\}.$$

One finds

$$\phi_{r,\gamma}(w) = \begin{cases} |w|_r - \frac{r-1}{r}\gamma^{1/(r-1)} & \text{if } |w|_r > \gamma^{1/(r-1)}, \\ \frac{1}{r\gamma}|w|_r^r & \text{if } |w|_r \le \gamma^{1/(r-1)}. \end{cases}$$

Thus, $\Phi_{r,\gamma}(\vec{q})(x) = \phi_{r,\gamma}(q(x))$ for $x \in \Omega$.

REMARK 2.3. [illegible] $(\mathcal{P}_\gamma^\star)$ [illegible] $\gamma^{s-1}/(s\alpha^{s-1})$ [illegible] $\tilde{\gamma}/s$ $\tilde{\gamma} > 0$ [illegible] (2.5) [illegible]

$$\max(\alpha\tilde{\gamma}^{r-1}, |\nabla\bar{u}_{\tilde{\gamma}}|_r^{r-1})\bar{\vec{p}}_{\tilde{\gamma}} - \alpha(|\nabla\bar{u}_{\tilde{\gamma}}|^{r-1} \star \sigma(\nabla\bar{u}_{\tilde{\gamma}})) = 0. \tag{2.6}$$

[illegible] $\tilde{\gamma} = \gamma^{s-1}/\alpha^{s-1}$ (2.6) [illegible] (2.5)

3. Primal-dual algorithm and its convergence. As it turned out in recent work, primal-dual schemes are typically superior to purely primal methods; see [6, 14]. Thus, we take (2.4a) and (2.5) as the starting point for devising an algorithm for solving the discrete analogue of $(\mathcal{P}_\gamma)$. In this way we naturally involve the primal variable

u and the dual variable $\vec{p}$. Consequently, we compute the solutions to the discrete versions of $(\mathcal{P}_\gamma)$ and $(\mathcal{P}_\gamma^\star)$ simultaneously. The algorithm proposed here is related to semismooth Newton methods [13, 25, 30]. In what follows we denote discretized quantities by superscript h. For a vector $w \in \mathbb{R}^l$ we denote by $D(w) := \operatorname{diag}(w)$ the $l \times l$ diagonal matrix with diagonal entries w_i.

The discrete analogues of (2.4a) and (2.5) are

$$B_\mu^h \bar{u}_\gamma^h - \operatorname{div}^h \bar{p}_\gamma^h = (K^h)^\top z^h, \tag{3.1a}$$

$$\max(\gamma e^h, \eta_r(\nabla^h \bar{u}_\gamma^h)) \star \bar{p}_\gamma^h - \alpha\big(|\nabla^h \bar{u}_\gamma^h|^{r-1} \star \sigma(\nabla^h \bar{u}^h)\big) = 0, \tag{3.1b}$$

where $\bar{u}_\gamma^h \in \mathbb{R}^m$, $\bar{p}_\gamma^h \in \mathbb{R}^{2m}$, for some $m \in \mathbb{N}$ which depends on the image size, solve the discrete analogues of $(\mathcal{P}_\gamma)$ and $(\mathcal{P}_\gamma^\star)$. Further, $z^h \in \mathbb{R}^m$ and $\nabla^h \in \mathbb{R}^{2m\times m}$, and $e^h \in \mathbb{R}^{2m}$ is the vector of all ones. We use $B_\mu^h = -\mu\Delta^h + (K^h)^\top K^h$, with $\Delta^h, (K^h)^\top K^h \in \mathbb{R}^{m\times m}$. Throughout we assume that B_μ^h is symmetric and positive definite. Note that this allows $\mu = 0$ if $(K^h)^\top K^h$ is invertible. The function $\eta_r : \mathbb{R}^{2m} \to \mathbb{R}^{2m}$ is defined by

$$(\eta_r(v))_i = (\eta_r(v))_{i+m} = |(v_i, v_{i+m})^\top|_r^{r-1} \quad \text{for } v \in \mathbb{R}^{2m},\ 1 \le i \le m.$$

The discrete gradient operator is composed as

$$\nabla^h = \begin{pmatrix} \nabla_x^h \\ \nabla_y^h \end{pmatrix} \in \mathbb{R}^{2m\times m},$$

where $\nabla_x^h \in \mathbb{R}^{m\times m}$ corresponds to the discrete derivative in the x-direction and $\nabla_y^h \in \mathbb{R}^{m\times m}$ corresponds to the derivative in the y-direction. In our implementation we use

$$\operatorname{div}^h = -(\nabla^h)^\top$$

and the implied ordering for the components of $\bar{p}_\gamma^h$. Subsequently we will frequently use the mapping $\xi_t : \mathbb{R}^{2m} \to \mathbb{R}^{2m}$ defined by

$$\xi_t(v)_i = \xi_t(v)_{i+m} := |(v_i, v_{i+m})^\top|_t$$

for $v \in \mathbb{R}^{2m}$, $i = 1, \dots, m$, and $1 < t < +\infty$. Note that $|\xi_t(v)|^{t-1} = \eta_t(v)$.

3.1. Approximate generalized Newton step. In the following we present our algorithm for the solution of (3.1a)–(3.1b). We restrict our discussion to the case $2 \le r < +\infty$ (and thus $1 < s \le 2$). It is well known that the max-operator and the ℓ_r-norm involved in (3.1b) are generalized differentiable and semismooth [13, 22, 25]. This is also true for the composition of the semismooth functions occurring in (3.1b). A particular element of the generalized Jacobian (see [8, 13, 25]) of $\max : \mathbb{R}^l \to \mathbb{R}^l$ is the diagonal matrix $G_{\max} \in \mathbb{R}^{l\times l}$ with

$$(G_{\max}(w))_{ii} := \begin{cases} 1 & \text{if } w_i \ge 0, \\ 0 & \text{if } w_i < 0 \end{cases} \quad \text{for } 1 \le i \le l.$$

Given approximations u_k^h, p_k^h of $\bar{u}_\gamma^h$, $\bar{p}_\gamma^h$, the results on generalized differentiability and semismoothness alluded to above and our assumption that $r \ge 2$ justify the

application of a Newton step to (3.1a)–(3.1b) at (u_k^h, p_k^h):

$$\begin{pmatrix} B_\mu^h & (\nabla^h)^\top \\ \big(-\alpha(r-1)D(|\nabla^h u_k^h|^{r-2}) + \chi_{\mathcal{A}_{k+1}} D(p_k^h) N_r^h(\nabla^h u_k^h)\big)\nabla^h & D(m_k^h) \end{pmatrix} \begin{pmatrix} \delta_u \\ \delta_p \end{pmatrix} = \begin{pmatrix} -B_\mu^h u_k^h - (\nabla^h)^\top p_k^h + (K^h)^\top z^h \\ \alpha\big(|\nabla^h u_k^h|^{r-1} \star \sigma(\nabla^h u_k^h)\big) - D(m_k^h) p_k^h \end{pmatrix}, \tag{3.2}$$

where we use $m_k^h = \max(\gamma e^h, \eta_r(\nabla^h u_k^h)) \in \mathbb{R}^{2m}$ for convenience, and $\chi_{\mathcal{A}_{k+1}} = D(t_k^h) \in \mathbb{R}^{2m\times 2m}$ with

$$(t_k^h)_i := \begin{cases} 1 & \text{if } \eta_r(\nabla^h u_k^h)_i \geq \gamma, \\ 0 & \text{else.} \end{cases} \tag{3.3}$$

Further, N_r^h denotes the Jacobian of η_r; i.e., for $v = (v_x, v_y)^\top \in \mathbb{R}^{2m}$, with $v_x, v_y \in \mathbb{R}^m$, we have

$$N_r^h(v) = (r-1)\big(D(\xi_r(v))\big)^{-1} \begin{pmatrix} D(|v_x|^{r-1} \star \sigma(v_x)) & D(|v_y|^{r-1} \star \sigma(v_y)) \\ D(|v_x|^{r-1} \star \sigma(v_x)) & D(|v_y|^{r-1} \star \sigma(v_y)) \end{pmatrix}.$$

Since $D(m_k^h)$ is invertible, we can eliminate δ_p from the Newton system (3.2). The remaining equation for δ_u is written as

$$H_k \delta_u = f_k, \tag{3.4}$$

where the matrix H_k and the right-hand side f_k are defined by

$$H_k := B_\mu^h + (\nabla^h)^\top D(m_k^h)^{-1}\big(\alpha(r-1)D(|\nabla^h u_k^h|^{r-2}) - \chi_{\mathcal{A}_{k+1}} D(p_k^h) N_r^h(\nabla^h u_k^h)\big)\nabla^h,$$

$$f_k := -B_\mu^h u_k^h + (K^h)^\top z^h - \alpha \nabla^h D(m_k^h)^{-1}\big(|\nabla^h u_k^h|^{r-1} \star \sigma(\nabla^h u_k^h)\big).$$

Note that H_k is the Schur complement of $D(m_k^h)$ in the system matrix in (3.2). Further properties of H_k are addressed next.

First, note that H_k is in general not symmetric, since

$$C_k^h := \alpha(r-1)D(|\nabla^h u_k^h|^{r-2}) - \chi_{\mathcal{A}_{k+1}} D(p_k^h) N_r^h(\nabla^h u_k^h)$$

is not. However, in the solution, i.e., for $(u_k^h, p_k^h) = (\bar{u}_\gamma^h, \bar{p}_\gamma^h)$ we infer from (3.1b) that $(p_k^h)_i = \alpha(\eta_r \nabla^h u_k^h))_i^{-1} |\nabla^h u_k^h|_i^{r-2} \sigma(\nabla^h u_k^h)_i$ holds for all $i \in \{1, \dots, 2m\}$ with $\eta_r(\nabla^h u_k^h)_i \geq \gamma$. Thus,

$$\begin{aligned} &\chi_{\mathcal{A}^{k+1}} D(p_k^h) N_r^h(\nabla^h u_k^h) \\ &\quad = (r-1)\chi_{\mathcal{A}^{k+1}}\big(D(\xi_r(\nabla^h u_k^h))\big)^{-1} D(p_k^h) \\ &\qquad\qquad \cdot \begin{pmatrix} D(|\nabla_x^h u_k^h|^{r-1} \star \sigma(\nabla_x^h u_k^h)) & D(|\nabla_y^h u_k^h|^{r-1} \star \sigma(\nabla_y^h u_k^h)) \\ D(|\nabla_x^h u_k^h|^{r-1} \star \sigma(\nabla_x^h u_k^h)) & D(|\nabla_y^h u_k^h|^{r-1} \star \sigma(\nabla_y^h u_k^h)) \end{pmatrix} \\ &\quad = \alpha(r-1)\chi_{\mathcal{A}^{k+1}}\big(D(\xi_r(\nabla^h u_k^h))\big)^{-r} \begin{pmatrix} D_1 & D_2 \\ D_2 & D_3 \end{pmatrix}, \end{aligned}$$

where

$$\begin{aligned} D_1 &= D\big(|\nabla_x^h u_k^h|^{2r-2}\big), \\ D_2 &= D\big(|\nabla_x^h u_k^h|^{r-1} \star |\nabla_y^h u_k^h|^{r-1} \star \sigma(\nabla_x^h u_k^h) \star \sigma(\nabla_y^h u_k^h)\big), \\ D_3 &= D\big(|\nabla_y^h u_k^h|^{2r-2}\big). \end{aligned}$$

Hence, in the solution the matrix C_k^h—and thus the entire system matrix H_k—is symmetric.

In what follows we investigate definiteness properties of the matrix H_k. Whenever H_k is positive definite, these observations lead us to simple modifications to obtain a uniformly positive definite matrix. This guarantees that in each (approximate) Newton step a descent direction is computed.

We start by considering properties of C_k^h. Note that one can reorder the indices such that $D(p_k^h)N_r^h(\nabla^h u_k^h)$ becomes a block-diagonal matrix where every 2×2 diagonal block has the structure

$$(n_k^h)_i := \frac{r-1}{(\xi_r(\nabla^h u_k^h))_i}\begin{pmatrix} (p_k^h)_i|\nabla_x^h u_k^h|_i^{r-1}\sigma(\nabla_x^h u_k^h)_i & (p_k^h)_i|\nabla_y^h u_k^h|_i^{r-1}\sigma(\nabla_y^h u_k^h)_i \\ (p_k^h)_{i+m}|\nabla_x^h u_k^h|_i^{r-1}\sigma(\nabla_x^h u_k^h)_i & (p_k^h)_{i+m}|\nabla_y^h u_k^h|_i^{r-1}\sigma(\nabla_y^h u_k^h)_i \end{pmatrix}$$

for $1 \le i \le m$. By this reordering, C_k^h is transformed into a block-diagonal matrix with diagonal blocks

$$(c_k^h)_i := \alpha(r-1)\begin{pmatrix} |\nabla_x^h u_k^h|_i^{r-2} & 0 \\ 0 & |\nabla_y^h u_k^h|_i^{r-2} \end{pmatrix} - (t_k^h)_i(n_k^h)_i, \tag{3.5}$$

where $(t_k^h)_i \in \{0,1\}$ is defined in (3.3). Obviously, $(c_k^h)_i$ is positive semidefinite for all inactive indices, i.e., for all i with $(t_k^h)_i = 0$. It remains to find conditions which imply that $(c_k^h)_i$ is positive semidefinite whenever $(t_k^h)_i = 1$. It is well known that positive semidefiniteness of $(c_k^h)_i$ is characterized by nonnegativity of all eigenvalues of its symmetrization. Therefore, we compute the eigenvalues $\lambda((\hat{c}_h^k)_i)^+$ and $\lambda((\hat{c}_h^k)_i)^-$ of $(\hat{c}_k^h)_i := \big((c_k^h)_i + (c_k^h)_i^\top\big)/2$, where

$$(c_k^h)_i = (r-1)\begin{pmatrix} a_{k,i} & b_{k,i} \\ c_{k,i} & d_{k,i} \end{pmatrix}$$

with

$$\begin{aligned}
a_{k,i} &= \alpha|\nabla_x^h u_k^h|_i^{r-2} - (\xi_r(\nabla^h u_k^h))_i^{-1}(p_k^h)_i|\nabla_x^h u_k^h|_i^{r-1}\sigma(\nabla_x^h u_k^h)_i, \\
b_{k,i} &= -(\xi_r(\nabla^h u_k^h))_i^{-1}(p_k^h)_i|\nabla_y^h u_k^h|_i^{r-1}\sigma(\nabla_y^h u_k^h)_i, \\
c_{k,i} &= -(\xi_r(\nabla^h u_k^h))_i^{-1}(p_k^h)_{i+m}|\nabla_x^h u_k^h|_i^{r-1}\sigma(\nabla_x^h u_k^h)_i, \\
d_{k,i} &= \alpha|\nabla_y^h u_k^h|_i^{r-2} - (\xi_r(\nabla^h u_k^h))_i^{-1}(p_k^h)_{i+m}|\nabla_y^h u_k^h|_i^{r-1}\sigma(\nabla_y^h u_k^h)_i.
\end{aligned}$$

Since we are considering the case $(t_k^h)_i = 1$, from (3.3) it follows that $(\xi_r(\nabla^h u_k^h))_i > 0$ and thus the above expressions are well defined. For the eigenvalues of $(\hat{c}_k^h)_i$ we obtain

$$\lambda((\hat{c}_k^h)_i)^\pm = \frac{a_{k,i}+d_{k,i}}{2} \pm \sqrt{\left(\frac{a_{k,i}+d_{k,i}}{2}\right)^2 + \left(\frac{b_{k,i}+c_{k,i}}{2}\right)^2 - a_{k,i}d_{k,i}}. \tag{3.6}$$

Let us now assume that the following two conditions hold:

$$\text{(C)} \qquad \xi_s(p_k^h)_i \le \alpha \quad \text{and} \quad (b_{k,i}+c_{k,i})^2 \le 4a_{k,i}d_{k,i}.$$

The first condition implies that $a_{k,i} \ge 0$ and $d_{k,i} \ge 0$. The second condition yields that $(b_{k,i}+c_{k,i})^2/4 - a_{k,i}d_{k,i}$ is nonpositive. Comparing with (3.6) we obtain that both eigenvalues $\lambda((\hat{c}_k^h)_i)^\pm$ are nonnegative real numbers, and hence $(c_k^h)_i$ is positive semidefinite.

Let us briefly interpret the conditions in (C). The condition $\xi_s(p_k^h)_i \leq \alpha$ corresponds to feasibility of the dual variable. Assuming that this condition holds, the latter condition in (C) is more likely to be satisfied if the matrix $(c_k^h)_i$ is close to a symmetric matrix, which can be seen as follows: assuming $b_{k,i} = c_{k,i}$ we obtain

(3.7)
$$\begin{aligned}\frac{(b_{k,i}+c_{k,i})^2}{4} - a_{k,i}d_{k,i} &= b_{k,i}c_{k,i} - a_{k,i}d_{k,i}\\ &= -\alpha^2|\nabla_x^h u_k^h|_i^{r-2}|\nabla_y^h u_k^h|_i^{r-2} + \alpha(\xi_r(\nabla^h u_k^h))_i^{-1}\\ &\qquad \cdot |\nabla_x^h u_k^h|_i^{r-2}|\nabla_y^h u_k^h|_i^{r-2}\left((p_k^h)_{i+m}(\nabla_y^h u_k^h)_i + (p_k^h)_i(\nabla_x^h u_k^h)_i\right)\\ &\leq \alpha|\nabla_x^h u_k^h|_i^{r-2}|\nabla_y^h u_k^h|_i^{r-2}(-\alpha + (\xi_s(p_k^h))_i)\\ &\leq 0.\end{aligned}$$

Above, the Hölder inequality and the assumption that $\xi_s(p_k^h)_i \leq \alpha$ are used. We summarize our results in the following lemma.

LEMMA 3.1. [illegible] (C) [illegible] $i \in \{1,\dots,m\}$ [illegible] C_k^h [illegible]

An immediate corollary of Lemma 3.1 is stated next.

COROLLARY 3.2. [illegible] (C) [illegible] $i \in \{1,\dots,m\}$ [illegible] $k \in \mathbb{N}$ [illegible] $k \in \mathbb{N}$ [illegible] H_k [illegible] $\lambda_{\min}(H_k) \geq \lambda_{\min}(B_\mu^h) > 0$ [illegible] $\{H_k^{-1}\}_{k\in\mathbb{N}}$ [illegible]

[illegible]. The assertions follow from the fact that $B_\mu^h = -\mu\Delta^h + (K^h)^\top K^h$ is symmetric and positive definite by assumption and from Lemma 3.1. ∎

In the case $r = s = 2$ the second assumption in (C) is automatically satisfied, as can be seen from the next lemma.

LEMMA 3.3. [illegible] $r = s = 2$ [illegible] (C) [illegible] C_k^h [illegible]. Suppose that $r = s = 2$ and that the first condition in (C) is satisfied. Then, for $k \in \mathbb{N}$ and $i \in \{1,\dots,m\}$,

$$\begin{aligned}&(b_{k,i}+c_{k,i})^2/4 - a_{k,i}d_{k,i}\\ &\quad= (2\xi_2(\nabla^h u_k^h)_i)^{-2}\Big((p_k^h)_i^2(\nabla_y u_k^h)_i^2 + (p_k^h)_{i+m}^2(\nabla_x u_k^h)_i^2\\ &\qquad\qquad - 2(p_k^h)_i(p_k^h)_{i+m}(\nabla_x u_k^h)_i(\nabla_y u_k^h)_i\Big) - \alpha^2\\ &\qquad + \alpha\xi_2(\nabla^h u_k^h)_i^{-1}\left((p_k^h)_i(\nabla_x u_k^h)_i + (p_k^h)_{i+m}(\nabla_y u_k^h)_i\right)\\ &\quad= (2\xi_2(\nabla^h u_k^h)_i)^{-2}\Big(\left((p_k^h)_i^2 + (p_k^h)_{i+m}^2\right)\left((\nabla_x u_k^h)_i^2 + (\nabla_y u_k^h)_i^2\right)\\ &\qquad\qquad - \left((p_k^h)_i(\nabla_x u_k^h)_i + (p_k^h)_{i+m}(\nabla_y u_k^h)_i\right)^2\Big) - \alpha^2\\ &\qquad + \alpha\xi_2(\nabla^h u_k^h)_i^{-1}\left((p_k^h)_i(\nabla_x u_k^h)_i + (p_k^h)_{i+m}(\nabla_y u_k^h)_i\right)\\ &\quad= -\Big(\alpha - (2\xi_2(\nabla^h u_k^h)_i)^{-1}\left((p_k^h)_i(\nabla_x u_k^h)_i + (p_k^h)_{i+m}(\nabla_y u_k^h)_i\right)\Big)^2\\ &\qquad + \left((p_k^h)_i^2 + (p_k^h)_{i+m}^2\right)/4\\ &\quad\leq -\alpha^2/4 + \left((p_k^h)_i^2 + (p_k^h)_{i+m}^2\right)/4\\ &\quad\leq 0.\end{aligned}$$

Thus, for $r = s = 2$, the second condition in (C) follows from $\xi_s(p_k^h)_i \le \alpha$ as asserted in the lemma. ∎

Whenever (C) is satisfied, Lemma 3.1 and Corollary 3.2 guarantee that the solution of (3.4) exists for all k and that it is a descent direction for the objective function in $(\mathcal{P}_\gamma)$. In general, however, the assumptions in (C) are unlikely to hold for all $i \in \{1, \dots, m\}$ and $k \in \mathbb{N}$. To guarantee the positive definiteness of the system matrix in our Newton-type step, in the subsequent algorithm we modify the term involving $D(p_k^h)N_r^h(\nabla^h u_k^h)$ for indices i which violate (C).

As above, we work with the reordered system matrix and consider for $i \in \{1, \dots, m\}$ the 2×2 matrix $(c_k^h)_i$ as defined in (3.5). To enforce the first condition in (C) we replace $\big((p_k^h)_i, (p_k^h)_{i+m}\big)$ by $\alpha \max\big(\alpha, \xi_s(p_k^h)_i\big)^{-1}\big((p_k^h)_i, (p_k^h)_{i+m}\big)$ when assembling the system matrix; i.e., we project p_k^h onto the feasible set. The resulting 2×2 matrix is denoted by

$$(\tilde{c}_k^h)_i = \begin{pmatrix} \tilde{a}_{k,i} & \tilde{b}_{k,i} \\ \tilde{c}_{k,i} & \tilde{d}_{k,i} \end{pmatrix}.$$

Due to the projection step we have $\tilde{a}_{k,i} \ge 0$ and $\tilde{d}_{k,i} \ge 0$. Then we check whether $(\tilde{b}_{k,i} + \tilde{c}_{k,i})^2/4 - \tilde{a}_{k,i}\tilde{d}_{k,i} \le 0$ is satisfied. If this is not the case we dampen the off-diagonal elements by $\nu_{k,i} = 2\sqrt{\tilde{a}_{k,i}\tilde{d}_{k,i}}/|\tilde{b}_{k,i} + \tilde{c}_{k,i}|$. From this we obtain

$$(\tilde{\tilde{c}}_k^h)_i = \begin{pmatrix} \tilde{a}_{k,i} & \nu_{k,i}\tilde{b}_{k,i} \\ \nu_{k,i}\tilde{c}_{k,i} & \tilde{d}_{k,i} \end{pmatrix}.$$

The resulting 2×2 matrix is positive semidefinite; i.e., its symmetrization has nonnegative eigenvalues. Revoking the reordering of the indices results in a modified system matrix which we denote by H_k^+. It replaces H_k in (3.4). We have the following result (compare with Corollary 3.2, also for a proof).

LEMMA 3.4. [illegible] H_k^+ [illegible] $\lambda_{\min}(H_k^+) \ge \lambda_{\min}(B_\mu^h) > 0$ [illegible] $k \in \mathbb{N}$ [illegible] $\{(H_k^+)^{-1}\}_{k\in\mathbb{N}}$ [illegible]

3.2. Algorithm. Now we are prepared for defining our approximate generalized Newton method for solving (3.1a)–(3.1b). Whenever (C) is not satisfied, it operates with H_k^+; otherwise the true system matrix H_k is used. In the next section we show that the new algorithm converges at a locally superlinear rate. This convergence result parallels the one for generalized Newton methods; see, e.g., [13, 30].

We propose the following algorithm.

ALGORITHM.

1. Initialize $(u_0^h, p_0^h) \in \mathbb{R}^m \times \mathbb{R}^{2m}$ and set $k = 0$.
2. Estimate the active sets, i.e., determine $\chi_{\mathcal{A}_{k+1}} \in \mathbb{R}^{2m\times 2m}$.
3. Compute H_k^+ if (C) is not satisfied for all $i = \{1, \dots, m\}$; otherwise set $H_k^+ = H_k$. Solve
$$H_k^+ \delta_u = f_k$$
for δ_u.
4. Compute δ_p according to (3.2).
5. Update $u_{k+1}^h := u_k^h + \delta_u$, $p_{k+1} := p_k^h + \delta_p$.
6. Stop, or set $k := k + 1$ and go to step 2.

Note that during the iteration of our algorithm the dual variable p is allowed to become infeasible with respect to the discrete version of the Fenchel dual problem $(\mathcal{P}_\gamma^\star)$. This differs significantly from the method in [6], where the dual iterates are forced to stay in the interior of the feasible region. Such a restriction may lead to prohibitively small step sizes in Newton's method.

3.3. Convergence analysis. In this section we investigate convergence properties of the above algorithm.

LEMMA 3.5. [illegible] $\bar{u}_\gamma^h$ $\bar{p}_\gamma^h$ [illegible] $(\mathcal{P}_\gamma)$ [illegible] $(\mathcal{P}_\gamma^\star)$ [illegible] $p_k^h \to \bar{p}_\gamma^h$ [illegible] $u_k^h \to \bar{u}_\gamma^h$ [illegible] H_k^+ [illegible] H_k [illegible] $k \to \infty$

[illegible]. Our argumentation again utilizes the reordered system. As above we argue for each 2×2 block separately. For inactive indices, i.e., for i with $(t_k^h)_i = 0$, the original and the modified 2×2 blocks coincide.

Let us now turn to active indices, i.e., to indices where $(t_k^h)_i = 1$. First note that the assumption $p_k^h \to \bar{p}_\gamma^h$ implies that $(\xi_s(p_k^h))_i \to (\xi_s(\bar{p}_\gamma^h))_i \leq \alpha$. Hence, for the projected dual variables it holds that

$$\alpha \max\big(\alpha, \xi_s(p_k^h)\big)^{-1}\big((p_k^h)_i, (p_k^h)_{i+m}\big) \to \big((\bar{p}_\gamma^h)_i, (\bar{p}_\gamma^h)_{i+m}\big).$$

Moreover, the matrix $(c_k^h)_i$ converges to a symmetric matrix as $(u_k^h, p_k^h) \to (\bar{u}_\gamma^h, \bar{p}_\gamma^h)$ (recall the discussion on page 8). Hence, the estimate (3.7) implies that $e_{k,i} := (b_{k,i} + c_{k,i})^2/4 - a_{k,i}d_{k,i} \to e_{\gamma,i} \leq 0$ for all i. If $e_{\gamma,i} < 0$, then $e_{k,i} < 0$ for all $k \geq k_0$ with some $k_0 \in \mathbb{N}$. Thus, for all $k \geq k_0$ no dampening of the off-diagonal elements occurs. If $e_{\gamma,i} = 0$, it is easy to see that $\nu_{k,i} \to 1$ as $p_k^h \to \bar{p}_\gamma^h$ and $u_k^h \to \bar{u}_\gamma^h$. Hence, all modified 2×2 diagonal blocks converge to the original blocks as $k \to \infty$. Revoking the reordering of the indices ends the proof. ∎

The local convergence of the above algorithm is the subject of the next theorem.

THEOREM 3.6. [illegible] (u_k^h, p_k^h) [illegible] $(\bar{u}_\gamma^h, \bar{p}_\gamma^h)$ [illegible] (u_0^h, p_0^h) [illegible] $(\bar{u}_\gamma^h, \bar{p}_\gamma^h)$

[illegible]. Recall that (3.1a)–(3.1b) are semismooth. However, since we possibly modify our system matrix, we end up with an approximate semismooth Newton method, and fast local convergence does not follow from standard arguments.

Utilizing a technique similar to that in the proof of Lemma 3.5 we can show that for each $\Delta > 0$ there exists a radius $\rho > 0$ such that if (u_k^h, p_k^h) is in a ρ-ball around $(\bar{u}_\gamma^h, \bar{p}_\gamma^h)$, then there exists $C_2 > 0$ with

$$\|H_k - H_k^+\| \leq \Delta \quad \text{and} \quad \|H_k^+\| \leq C_2.$$

Thus, the assumptions of Theorem 4.1. in [30] are satisfied. Using this theorem we obtain that the iterates (u_k^h, p_k^h) converge to $(\bar{u}_\gamma^h, \bar{p}_\gamma^h)$ at a linear rate, provided that (u_0^h, p_0^h) is sufficiently close to $(\bar{u}_\gamma^h, \bar{p}_\gamma^h)$. Then, Lemma 3.5 implies that $H_k^+ \to H_k$. As a consequence, the assumptions of Theorem 4.2. in [30] are satisfied, which yields that the iterates converge locally superlinearly. ∎

As a globalization strategy for the above algorithm we use an Armijo linesearch technique with backtracking for the functional of the primal regularized problem $(\mathcal{P}_\gamma)$. Let us point out, however, that in our numerical experiments the step size is almost always equal to one; i.e., a full step of our generalized Newton algorithm leads to a sufficient decrease in the functional.

4. Aspects of the implementation. To illustrate the feasibility of our approach for large scale systems, our implementation utilizes an inexact solve of the linear system arising in step 3 of the algorithm.

In the case $r = 2$ (Gauss-TV regularization) the system matrix arising in our algorithm simplifies and no dampening of the off-diagonal elements is necessary to obtain positive definiteness of the system matrix (see Lemma 3.3). However, an immediate application of the preconditioned conjugate gradient (PCG) method for the solution of the linear system in each Newton step is not feasible since H_k^+ (like H_k) is in general not symmetric. Therefore, in our implementation we replace $N_r^h(\nabla^h u_k^h)$ by its symmetrization $\tilde{N}_k^h(\nabla^h u_k^h) := 1/2\big(N_r^h(\nabla^h u_k^h) + N_r^h(\nabla^h u_k^h)^\top\big)$. As a result, the system matrix $\tilde{H}_k^+$ is symmetric and the PCG method can be applied.

The symmetrization of H_k^+ is justified by the fact that in the solution the system matrix H_k (respectively, H_k^+) is symmetric and therefore coincides with its symmetrization $\tilde{H}_k$ (see page 8). Thus, using the symmetrization of H_k^+ in step 3 of our algorithm still results in a locally superlinearly convergent algorithm. Obviously, a similar observation holds true for the case $r > 2$. However, our numerical experience shows that proceeding with the nonsymmetric matrix gives slightly better results with respect to the number of primal-dual iterations (usually 0–3 fewer iterations are needed). Therefore, we utilize the biconjugate gradient stabilized (BICGSTAB) algorithm (see, e.g., [27]) to solve the nonsymmetric positive definite linear system arising in each Newton step. We point out that alternatively one may use specialized techniques such as the recently developed semiconjugate direction method; see [34].

Next we address the accuracy required when solving the Newton-type system in every iteration. For this purpose let res_k denote the residual of the nonlinear system. Motivated by the superlinear convergence rate we stop the PCG or the BICGSTAB iteration as soon as the norm of the residual of the linearized system drops below

$$\mathrm{tol}_{k+1} := 0.1 \min\left\{ \left(\frac{\mathrm{res}_k}{\mathrm{res}_0}\right)^{3/2}, \frac{\mathrm{res}_k}{\mathrm{res}_0} \right\}.$$

As a preconditioner for the system matrix we use an incomplete Choleski factor of the symmetrization of H_k^+. The algorithm is initialized with a Gaussian smoothing of the noisy image z and with $p_0^h = 0$. We stop the Newton iteration as soon as the initial residual is reduced by a factor of 10^{-6}.

5. Numerical results. In this section we present numerical results for the Gauss-TV case $r = 2$, as well as for the case $r > 2$. We discretize the Laplace operator with the standard five-point stencil with homogenous Dirichlet boundary conditions and use forward differences for the gradient and the corresponding discrete adjoint scheme for the divergence operator. Unless otherwise specified we concentrate on image denoising, i.e., $K = \mathrm{id}$. The choice of the parameters α and γ is based on the quality of the restored images, i.e., the parameters are adjusted manually.

5.1. Behavior of the algorithm. Here, we study the behavior of our proposed method with respect to its reconstruction ability, convergence properties such as speed of convergence, stability with respect to the noise level, the choice of r, and possible violations of the dual bound constraint during the iteration. In the following section we provide a qualitative study of the influence of the regularization parameter γ on, e.g., the staircase effect.

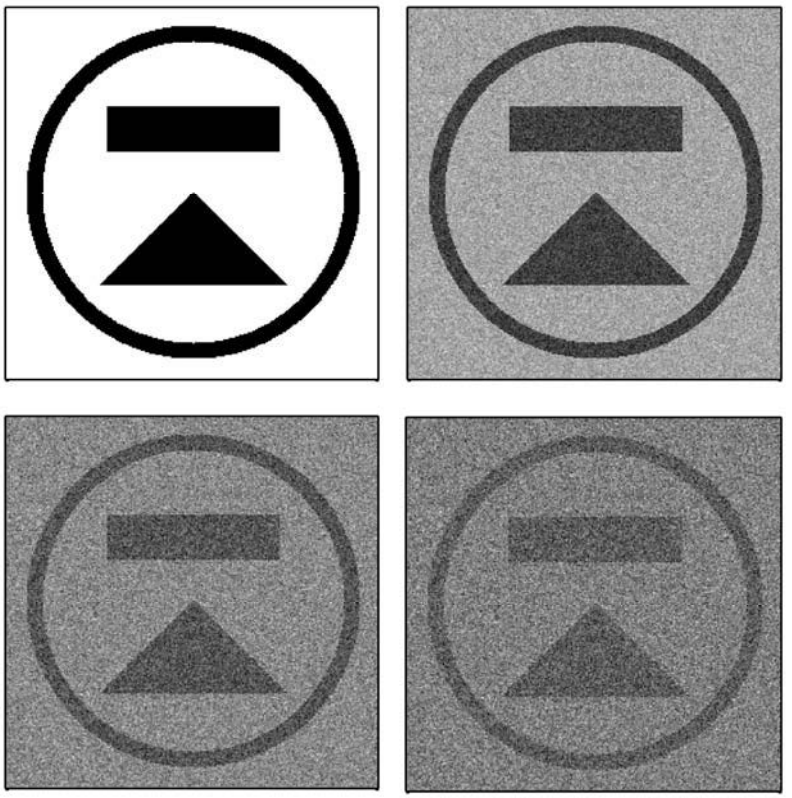


FIG. 5.1. *Example* 1: *Original image (*256 × 256 *pixels) and image with* 20% *Gaussian white noise (upper row), images containing* 50% *and* 80% *noise (lower row).*

TABLE 5.1
Example 1: *Number of primal-dual iterations (*#PD*) and overall number of CG iterations (*#CG*) for different noise levels.*

Noise	#PD	#CG
20%	11	48
50%	12	58
80%	13	61

5.1.1. Example 1. Our first test examples are displayed in Figure 5.1. On the upper left-hand side the original 256 × 256 image, which is similar to the ones used in [6, 14], is shown. A degraded image containing 20% of Gaussian white noise is given by the upper right plot. In the lower row images with a noise level of 50% respectively 80% are depicted. First we consider Gauss-TV regularization, i.e., $r = 2$ in $(\mathcal{P}_\gamma)$. We choose $\mu = 0$, $\gamma = 10^{-3}$, and $\alpha = 0.35$ for denoising the image with 20% noise. For the images containing 50% and 80% Gaussian white noise $\alpha = 0.90$ and $\alpha = 1.35$ are chosen, respectively. The numbers of primal-dual iterations of the algorithm for the three different noise levels are shown in Table 5.1, where we also provide the overall number of CG iterations. Note that our method behaves very stably with respect to the noise level. In our tests we also observe that the algorithm's performance is

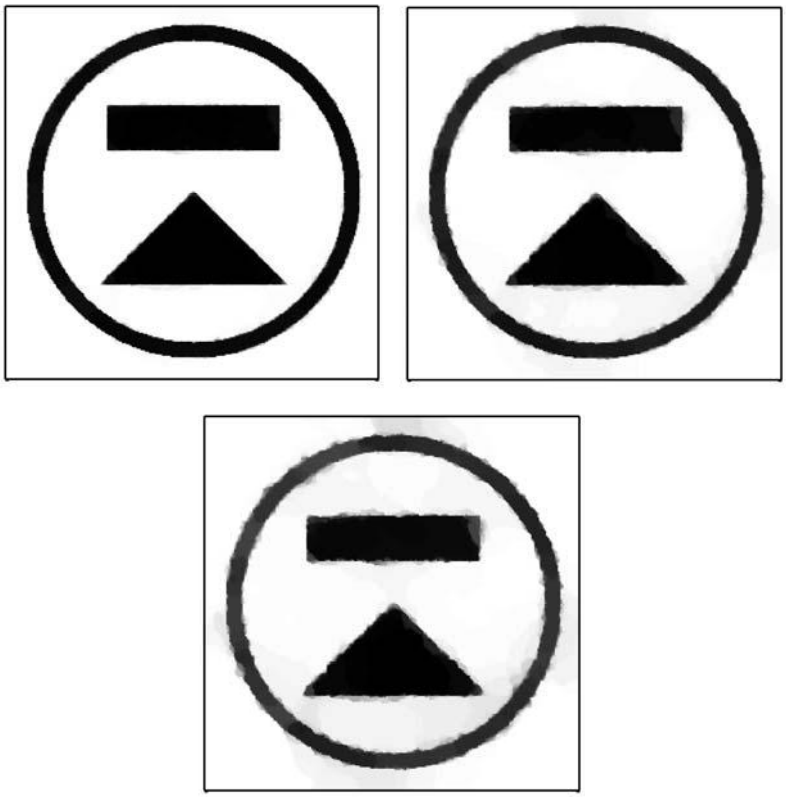


FIG. 5.2. *Example* 1: *Results of our algorithm with* $r = 2$ *for the image with* 20% *noise (upper row, left), with* 50% *noise (upper row, right), and with* 80% *noise (lower row).*

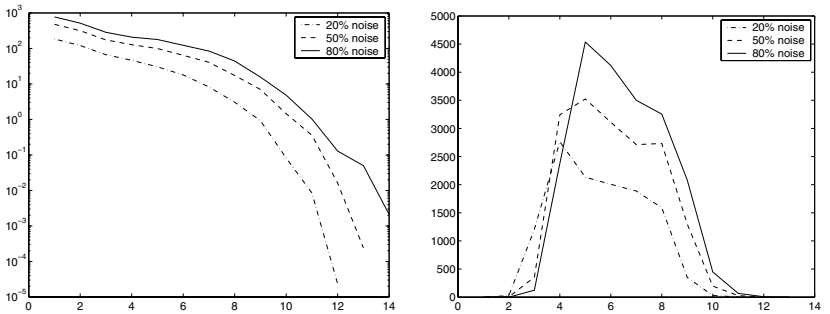


FIG. 5.3. *Example* 1: *Norm of the residual (left) and number of pixels where the dual variable is infeasible (right) versus iterations.*

rather independent of its initialization. Even for seemingly poor initializations u_0^h the algorithm converges and requires only two to three additional iterations when compared to a run initialized by the smoothed noisy image. The denoised images produced by our algorithm are shown in Figure 5.2. To document the convergence behavior we show the residual of the nonlinear system for the three noise levels in a logarithmic plot (Figure 5.3, left). We observe a fast decrease towards the end of

the iterations, which indicates a rapid local convergence of the method. To emphasize that our method, in contrast to the algorithm in, e.g., [6], allows violations of the dual constraints, in the right plot of Figure 5.3 we depict the number of points, where the iterates p_k^h are infeasible, i.e., where $|p_k^h|_s > (1+\varepsilon)\alpha$. Here $\varepsilon = 10^{-5}$ is introduced to compensate roundoff errors. Since the dual variable is initialized with $p_0^h = 0$, the number of infeasible points is small for the starting iterations, then it increases before it tends to zero as the algorithm approaches the solution.

[illegible] μ [illegible] γ. In our numerical tests we observe that the performance of the algorithm and the reconstructions are comparable for μ with $0 \le \mu \ll \alpha$. In particular, our choice $\mu = 0$ appears unproblematic. For large values of μ (i.e., $\mu \sim \alpha$ and larger), the quality of the reconstruction degrades (sharp edges are smoothed) and the number of overall PCG steps decreases. Concerning the influence of the parameter γ on the convergence behavior we observe that the method is not very sensitive with respect to γ. However, let us point out that large γ tends to decrease the iteration numbers.

[illegible]. We also tested our algorithm for images having different resolutions. For this purpose, the image containing 50% noise in Figure 5.1 was chosen, and reconstructions for analog images having 128×128 and 512×512 pixels were computed. Our algorithm stopped after 11, 12, and 11 iterations and overall 51, 58, and 57 PCG iterations for the resolutions 128×128, 256×256, and 512×512, respectively. This shows a remarkable stability with respect to the image size.

[illegible] $r > 2$. Next we test our algorithm with $r = 2.5$. For our test we use the image from Figure 5.1 containing 50% of Gaussian white noise and a rotated version of the same image. The parameters are set to $\mu = 0$, $\alpha = 0.9$, and $\gamma = 2.5 \cdot 10^{-5}$. We point out that for $r > 2$ the parameter γ must be chosen smaller as for $r = 2$ in order to obtain a reconstruction of comparable quality. This can be seen from the dual problem formulation $(\mathcal{P}_\gamma^\star)$, where s also influences the weight of the regularizing term; see the discussion in Remark 2.3. For both, the original and the rotated image, our algorithm terminates after 20 primal-dual iterations. Overall, 61 and 65 BICGSTAB iterations are needed, respectively. We observe local fast convergence of the iterates as expected from Theorem 3.6. Furthermore, during the iterations many components of the dual iterates violate the pointwise constraints. Since we are in the case $r > 2$, off-diagonal entries may be dampened in order to guarantee positive definiteness of the system matrix (see the discussion on page 11). In our computations, for the rotated image only in the first 10 iterations a few (at most 71) off-diagonal elements are scaled by dampening factors between 0.7 and 1. After iteration 10 no dampening is necessary. We contribute this behavior to the fact that the system matrix is already close to a symmetric matrix for which no dampening of off-diagonal elements is required (see the discussions on page 8 and (3.7)).

Results of our computations are shown in Figure 5.4. The reconstruction of the rotated image differs from the rotated reconstruction of the original image. This is partly due to the fact that the unit ball of the ℓ_r-norm is not invariant with respect to rotations for $r \neq 2$. Obviously, there are other reasons for the different reconstructions as well: rotating introduces errors to the image. Furthermore, one-sided differences for the gradients influence the flow of information in the reconstruction process.

[illegible]. Finally, we illustrate the restoration ability of our algorithm for blurred noisy images, i.e., for $K \neq \mathrm{id}$. We consider the image from Figure 5.1 without noise and choose a block Toeplitz matrix (see, e.g., [32]) for the discrete blurring operator K^h. Block Toeplitz matrices occur as

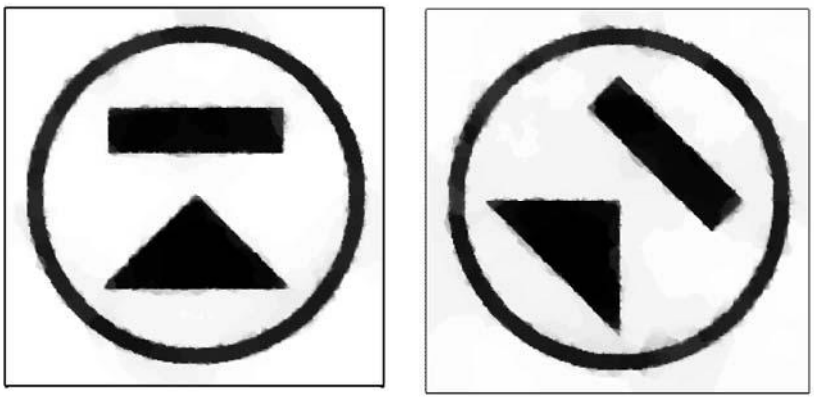


FIG. 5.4. *Example* 1: *Results of our calculations with* $r = 2.5$ *for the image from Figure* 5.1 *containing* 50% *of noise (left) and a rotated version of this image (right).*

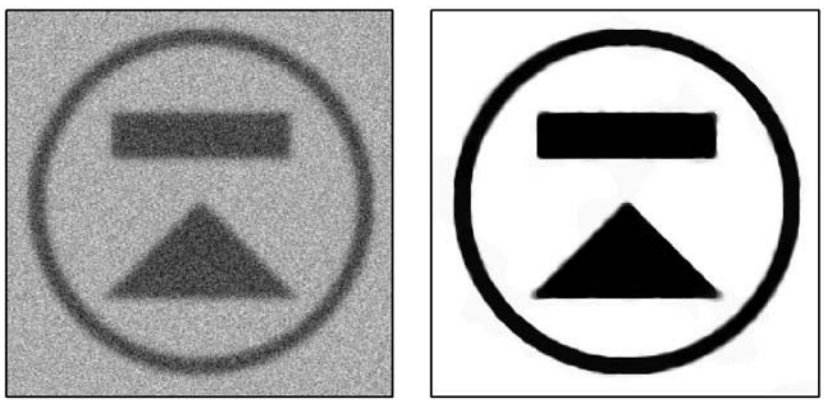


FIG. 5.5. *Example* 1: *Blurred noisy image (left) and reconstruction with* $r = 2$, $\alpha = 0.3$, *and* $\gamma = 10^{-3}$ *(right).*

matrix representations of discrete convolution operators, which represent a model for blurring of images. After applying K^h to the image, 20% of Gaussian white noise is added. The resulting degraded image can be seen on the left of Figure 5.5. For the reconstruction process we choose $r = 2$, $\alpha = 0.3$, $\gamma = 10^{-3}$, and $\mu = 0$. The result obtained after 16 primal-dual iterations can be seen on the right side of Figure 5.5.

5.1.2. Example 2 (validation of Theorem 3.6). In a second example we apply our method for enhancing the image shown in the left plot of Figure 5.6 that contains 10% of Gaussian white noise. Its resolution is 512×512 pixels. We apply Gauss-TV regularization, i.e., $r = 2$, with parameters $\mu = 0$, $\alpha = 0.08$, and $\gamma = 10^{-3}$. The result obtained after 11 iterations is shown in the right plot of Figure 5.6. To illustrate the convergence behavior of the method, in Table 5.2 we report on the norm of the nonlinear residual, the number of dually infeasible points, the step length, and the quotient

$$q_k = \frac{\|u_{k+1}^h - \bar{u}_\gamma^h\|}{\|u_k^h - \bar{u}_\gamma^h\|} \tag{5.1}$$

for each iteration. Above, $\bar{u}_\gamma^h$ denotes a beforehand calculated solution of the problem.

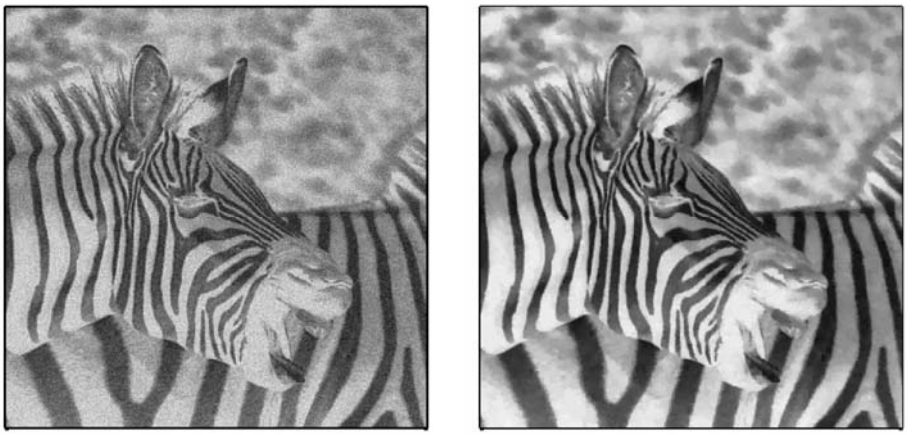


FIG. 5.6. *Example* 2: *Noisy* 512×512 *image (left) and results of our calculations with* $r = 2$, $\gamma = 10^{-3}$, *and* $\alpha = 0.08$ *(right).*

TABLE 5.2

Example 2: *For each iteration* k *we show the nonlinear residual* $\|res_k\|$, *the number* $\#CG_k$ *of PCG-iterations, the number* $\#|p_k^h|_2 > \alpha$ *of components of* p_k^h *that violate the dual constraint, the step size* β_k *and* q_k, *which is defined in* (5.1).

k	$\|res_k\|$	$\#\mathrm{CG}_k$	$\#\lvert p_k^h\rvert_2 > \alpha$	β_k	q_k
1	7.84e+1	2	0	1	0.412
2	4.12e+1	2	1425	1	0.391
3	2.01e+1	2	17284	1	0.361
4	1.31e+1	3	38990	1	0.315
5	9.48e 0	3	50246	1	0.265
6	7.10e 0	3	40267	1	0.249
7	3.68e 0	2	18641	1	0.272
8	1.37e 0	2	5346	1	0.239
9	3.81e–1	2	878	1	0.143
10	6.12e–2	3	73	1	0.086
11	3.44e–3	2	6	1	0.009
12	1.42e–5	2	0	1	0.000

We observe a decrease of q^k (except for iteration 7) which implies a locally superlinear convergence of the iterates (compare with Theorem 3.6). Again, the algorithm makes use of iterates that heavily violate the inequality constraints imposed on the dual variable.

Numerical experiments show that a robust and fast convergence of the method is essentially based on the replacement of the system matrix H_k by H_k^+. In fact, due to our approximation technique based on (C), a positive definite matrix is obtained which guarantees that δ_u is a descent direction reducing the objective function of $(\mathcal{P}_\gamma)$. Indeed, our line search always accepts step size $\beta_k = 1$ for achieving a sufficient decrease.

5.2. Qualitative study of the dual regularization. The image restoration model $(\mathcal{P}_{BV})$ tends to produce piecewise constant, i.e., "blocky" images. This may be useful in some applications. However, for images containing affine regions with noise, the result will be an adverse "staircase" effect. To prevent the method from

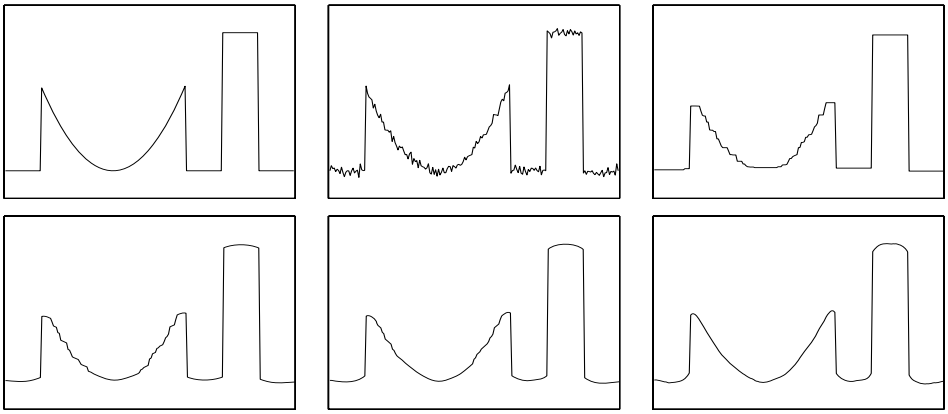


FIG. 5.7. *Upper row: Original one-dimensional image (left), noisy image (middle), and result of our calculations with* $\gamma = 10^{-10}$ *(right). Lower row: results obtained with* $\gamma = 0.1$ *(left),* $\gamma = 0.2$ *(middle), and* $\gamma = 0.5$ *(right).*

this oversharpening while preserving sharp edges, recently several strategies have been proposed and tested (see, e.g., [3, 7, 16]). In our approach, the parameter γ allows to control the BV model and the blocky structure in the image reconstruction. The regularization attached to γ induces a switch between a Gaussian-like regularization for small gradients and TV regularization for large gradients. To illustrate this influence of γ, a one-dimensional analogue to our algorithm was implemented. In Figure 5.7 the results for a one-dimensional image are shown, which were obtained by our algorithm with $\alpha = 10^{-2}$ and for various values of γ. The result for $\gamma = 0.5$ has exactly four active points at the solution. This implies that only at the four discontinuities of the original image TV regularization is applied, whereas all other regions are treated with a Gaussian regularization.

5.2.1. Example 3. Finally, we discuss the influence of γ on the two-dimensional noisy magnetic resonance (MR) image shown in Figure 5.8 (upper left). Its resolution is 256×256 pixels. A certain noise level in MR images is unavoidable due to background interference or other disturbances. Filters are used either to enhance the image quality without altering any image details or to prepare the image for a qualitative analysis, such as the segmentation of structures.

In [20] Gauss-TV denoising was shown to give good results for medical images. For this reason, we choose $r = 2$, the parameters $\mu = 0$ and $\alpha = 0.06$ and investigate the influence of the parameter γ on the reconstructed image. The result for $\gamma = 10^{-5}$ (see Figure 5.8, upper row, right) is obtained after 16 iterations. The reconstructions for $\gamma = 5 \cdot 10^{-3}$ and $\gamma = 5 \cdot 10^{-2}$ are obtained after 8 and 7 iterations of our algorithm, respectively. The corresponding results are shown in the lower row of Figure 5.8. Details of Figure 5.8 are shown in Figure 5.9. We observe that the reconstruction obtained with $\gamma = 10^{-5}$ is close to a piecewise constant image. This effect can be reduced by using larger γ values, e.g., $\gamma = 5 \cdot 10^{-3}$. However, depending on the noise level, if γ becomes too large, some of the noisy patterns may remain; see the result for $\gamma = 5 \cdot 10^{-2}$. In conclusion, this shows that the parameter γ can be useful in avoiding oversharpening of images while preserving sharp edges.

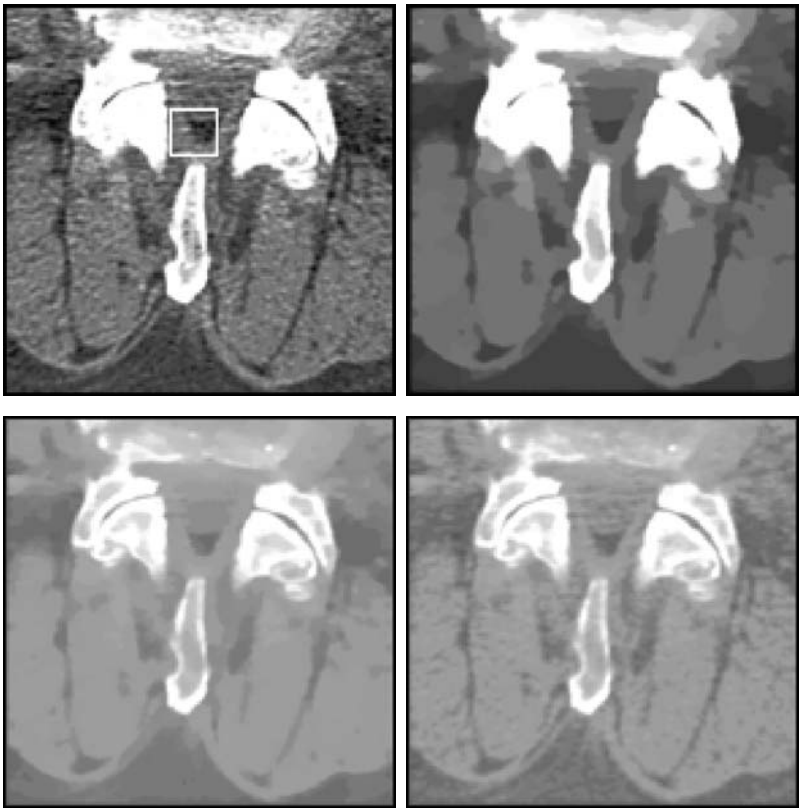


FIG. 5.8. *Example* 3: *Noisy magnetic resonance image (upper row, left, white box indicates the detail shown in Figure* 5.9*) and results of our calculations with* $r = 2$, $\mu = 0$, $\alpha = 0.06$ *and with* $\gamma = 10^{-5}$ *(upper row, right),* $\gamma = 5 \cdot 10^{-3}$ *(lower row, left), and* $\gamma = 5 \cdot 10^{-2}$ *(lower row, right).*

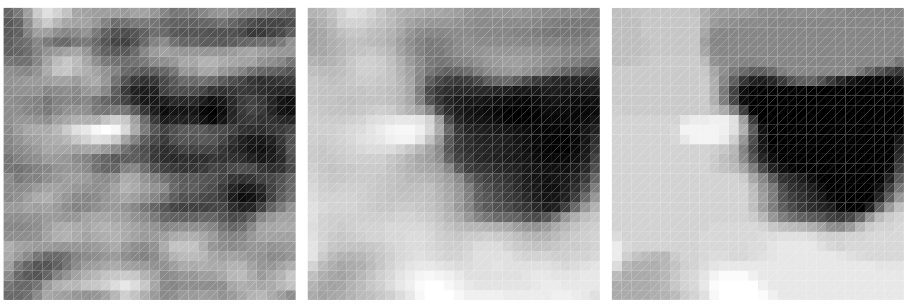


FIG. 5.9. *Example* 3: *Details of the images shown in Figure* 5.8. *Detail of the original image (right), reconstruction with* $\gamma = 5 \cdot 10^{-3}$ *(middle), and with* $\gamma = 10^{-5}$ *(right).*

Appendix A. Proof of Theorem 2.2. Subtracting the weak form of (2.4a) from the weak form of (2.3a) results in

$$\int_\Omega (\bar{u} - \bar{u}_\gamma) v \, dx + \mu \int_\Omega \nabla(\bar{u} - \bar{u}_\gamma)^\top \nabla v \, dx = \int_\Omega (\bar{\vec{p}}_\gamma - \bar{\vec{p}})^\top \nabla v \, dx \tag{A.1}$$

for all $v \in H_0^1(\Omega)$. Choosing $v := \bar{u} - \bar{u}_\gamma$ yields

$$\int_\Omega (\bar{u} - \bar{u}_\gamma)^2 \, dx + \mu \int_\Omega |\nabla(\bar{u} - \bar{u}_\gamma)|_2^2 \, dx = \int_\Omega (\bar{\vec{p}}_\gamma - \bar{\vec{p}})^\top \nabla(\bar{u} - \bar{u}_\gamma) \, dx. \tag{A.2}$$

Pointwise bounds for the term $(\bar{\vec{p}}_\gamma - \bar{\vec{p}})^\top \nabla(\bar{u} - \bar{u}_\gamma)$ are established as follows: First we split Ω into disjoint sets according to the complementarity conditions (2.3b) and (2.4b); namely, $\Omega = \mathcal{A}_\gamma \cup \mathcal{I}_\gamma = \mathcal{A} \cup \mathcal{I}$ with

$$\begin{aligned} \mathcal{A}_\gamma &= \{x \in \Omega : |\nabla \bar{u}_\gamma|_r^{r-1} > \gamma\}, \quad & \mathcal{I}_\gamma &= \Omega \setminus \mathcal{A}_\gamma, \\ \mathcal{A} &= \{x \in \Omega : |\nabla \bar{u}|_r^{r-1} > 0\}, & \mathcal{I} &= \Omega \setminus \mathcal{A}. \end{aligned} \tag{A.3}$$

Then we bound $(\bar{\vec{p}}_\gamma - \bar{\vec{p}})^\top \nabla(\bar{u} - \bar{u}_\gamma)$ separately on the four sets $\mathcal{A}_\gamma \cap \mathcal{A}$, $\mathcal{A}_\gamma \cap \mathcal{I}$, $\mathcal{I}_\gamma \cap \mathcal{A}$, and $\mathcal{I}_\gamma \cap \mathcal{I}$. Subsequently, we frequently dismiss the argument (x) when referring to function values at x. We start with $\mathcal{A}_\gamma \cap \mathcal{A}$, where we use the facts that $|\bar{\vec{p}}|_s = |\bar{\vec{p}}_\gamma|_s = \alpha$ and that $\bar{\vec{p}}_\gamma = \alpha\big(|\nabla \bar{u}_\gamma|^{r-1} \star (|\nabla \bar{u}_\gamma|)\big)/|\nabla \bar{u}_\gamma|_r^{r-1}$ and $\bar{\vec{p}} = \alpha\big(|\nabla \bar{u}|^{r-1} \star (|\nabla \bar{u}|)\big)/|\nabla \bar{u}|_r^{r-1}$:

$$\begin{aligned} (\bar{\vec{p}}_\gamma - \bar{\vec{p}})^\top \nabla(\bar{u} - \bar{u}_\gamma) &\le |\bar{\vec{p}}_\gamma|_s |\nabla \bar{u}|_r - \frac{\alpha}{|\nabla \bar{u}_\gamma|_r^{r-1}} |\nabla \bar{u}_\gamma|_r^r - \frac{\alpha}{|\nabla \bar{u}|_r^{r-1}} |\nabla \bar{u}|_r^r + |\bar{\vec{p}}|_s |\nabla \bar{u}_\gamma|_r \\ &= \alpha |\nabla \bar{u}|_r - \alpha |\nabla \bar{u}_\gamma|_r - \alpha |\nabla \bar{u}|_r + \alpha |\nabla \bar{u}_\gamma|_r = 0. \end{aligned} \tag{A.4}$$

Next we turn to the set $\mathcal{A}_\gamma \cap \mathcal{I}$, where we exploit the facts that $\nabla \bar{u} = 0$ and $|\bar{\vec{p}}_\gamma|_s = \alpha$:

$$\begin{aligned} (\bar{\vec{p}}_\gamma - \bar{\vec{p}})^\top \nabla(\bar{u} - \bar{u}_\gamma) &= (\bar{\vec{p}} - \bar{\vec{p}}_\gamma)^\top \nabla \bar{u}_\gamma \\ &\le |\bar{\vec{p}}|_s |\nabla \bar{u}_\gamma|_r - \frac{\alpha}{|\nabla \bar{u}_\gamma|_r^{r-1}} |\nabla \bar{u}_\gamma|_r^r \\ &= |\bar{\vec{p}}|_s |\nabla \bar{u}_\gamma|_r - \alpha |\nabla \bar{u}_\gamma|_r \le 0. \end{aligned} \tag{A.5}$$

On the set $\mathcal{I}_\gamma \cap \mathcal{A}$ one uses $|\bar{\vec{p}}_\gamma|_s \le \alpha$ and obtains similar to the above that

$$(\bar{\vec{p}}_\gamma - \bar{\vec{p}})^\top \nabla(\bar{u} - \bar{u}_\gamma) \le 0. \tag{A.6}$$

Finally, we turn to the set $\mathcal{I}_\gamma \cap \mathcal{I}$ where we have $\nabla \bar{u} = 0$, $|\nabla \bar{u}_\gamma|_r^{r-1} \le \gamma$, $|\bar{\vec{p}}|_s \le \alpha$, and $|\bar{\vec{p}}_\gamma|_s \le \alpha$. We obtain

$$\begin{aligned} (\bar{\vec{p}}_\gamma - \bar{\vec{p}})^\top \nabla(\bar{u} - \bar{u}_\gamma) &= (\bar{\vec{p}} - \bar{\vec{p}}_\gamma)^\top \nabla \bar{u}_\gamma \\ &\le |\bar{\vec{p}} - \bar{\vec{p}}_\gamma|_s |\nabla \bar{u}_\gamma|_r \\ &\le 2\alpha \gamma^{1/(r-1)}. \end{aligned} \tag{A.7}$$

Since $(\mathcal{A}_\gamma \cap \mathcal{A})$, $(\mathcal{A}_\gamma \cap \mathcal{I})$, $(\mathcal{I}_\gamma \cap \mathcal{A})$, $(\mathcal{I}_\gamma \cap \mathcal{I})$ provide a disjoint partitioning of Ω, the estimates (A.4)–(A.7) yield, together with (A.2),

$$\int_\Omega (\bar{u} - \bar{u}_\gamma)^2 \, dx + \mu \int_\Omega |\nabla(\bar{u} - \bar{u}_\gamma)|_2^2 \, dx \le \int_\Omega 2\alpha \gamma^{1/(r-1)} \, dx. \tag{A.8}$$

Hence, we infer that $\bar{u}_\gamma \to \bar{u}$ strongly in $H_0^1(\Omega)$ as $\gamma \to 0$. From (A.1) we obtain the weak convergence of $\bar{\vec{p}}_\gamma \to \bar{\vec{p}}$ in grad $H_0^1(\Omega) \subset \boldsymbol{L}^2(\Omega)$. This ends the proof. ∎

REFERENCES

[1] R. Acar and C. R. Vogel, *Analysis of bounded variation penalty methods for ill-posed problems*, Inverse Problems, 10 (1994), pp. 1217–1229.

[2] M. Bertalmio, V. Caselles, B. Rouge, and A. Sole, *TV-based image restoration with local constraints*, J. Sci. Comput., 19 (2003), pp. 95–122.

[3] P. Blomgren, T. F. Chan, and P. Mulet, *Extensions to total variation denoising*, in Proceedings Society of Photo-optical Instrumentation Engineers (SPIE)'97, San Diego, 1997.

[4] E. Casas, K. Kunisch, and C. Pola, *Regularization by functions of bounded variation and applications to image enhancement*, Appl. Math. Optim., 40 (1999), pp. 229–257.

[5] A. Chambolle and P.-L. Lions, *Image recovery via total variation minimization and related problems*, Numer. Math., 76 (1997), pp. 167–188.

[6] T. Chan, G. Golub, and P. Mulet, *A nonlinear primal-dual method for total variation-based image restoration*, SIAM J. Sci. Comput., 20 (1999), pp. 1964–1977.

[7] T. Chan, A. Marquina, and P. Mulet, *High-order total variation-based image restoration*, SIAM J. Sci. Comput., 22 (2000), pp. 503–516.

[8] F. H. Clarke, *Optimization and Nonsmooth Analysis*, Canadian Mathematical Society Series of Monographs and Advanced Texts, John Wiley & Sons, New York, 1983.

[9] D. C. Dobson and C. R. Vogel, *Convergence of an iterative method for total variation denoising*, SIAM J. Numer. Anal., 34 (1997), pp. 1779–1791.

[10] D. C. Dobson and F. Santosa, *An image-enhancement technique for electrical impedance tomography*, Inverse Problems, 10 (1994), pp. 317–334.

[11] I. Ekeland and R. Témam, *Convex Analysis and Variational Problems*, Classics Appl. Math. 28, SIAM, Philadelphia, 1999.

[12] E. Giusti, *Minimal Surfaces and Functions of Bounded Variation*, Monogr. Math. 80, Birkhäuser-Verlag, Basel, 1984.

[13] M. Hintermüller, K. Ito, and K. Kunisch, *The primal-dual active set strategy as a semismooth Newton method*, SIAM J. Optim., 13 (2003), pp. 865–888.

[14] M. Hintermüller and K. Kunisch, *Total bounded variation regularization as bilaterally constrained optimization problem*, SIAM J. Appl. Math., 64 (2004), pp. 1311–1333.

[15] K. Ito and K. Kunisch, *An active set strategy based on the augmented Lagrangian formulation for image restoration*, M2AN Math. Model. Numer. Anal., 33 (1999), pp. 1–21.

[16] K. Ito and K. Kunisch, *BV-type regularization methods for convoluted objects with edge, flat and grey scales*, Inverse Problems, 16 (2000), pp. 909–928.

[17] T. Kärkkäinen and K. Majava, *Nonmonotone and monotone active-set methods for image restoration.* I. *Convergence analysis*, J. Optim. Theory Appl., 106 (2000), pp. 61–80.

[18] T. Kärkkäinen and K. Majava, *Nonmonotone and monotone active-set methods for image restoration.* II. *Numerical results*, J. Optim. Theory Appl., 106 (2000), pp. 81–105.

[19] T. Kärkkäinen, K. Majava, and M. M. Mäkelä, *Comparison of formulations and solution methods for image restoration problems*, Inverse Problems, 17 (2001), pp. 1977–1995.

[20] S. L. Keeling, *Total variation based convex filters for medical imaging*, Appl. Math. Comput., 139 (2003), pp. 101–119.

[21] Y. Li and F. Santosa, *An Affine Scaling Algorithm for Minimizing Total Variation in Image Enhancement*, Report CTC94TR201, Cornell Theory Center, Cornell University, Ithaca, NY, 1994.

[22] R. Mifflin, *Semismooth and semiconvex functions in constrained optimization*, SIAM J. Control Optim., 15 (1977), pp. 959–972.

[23] S. Osher, A. Solé, and L. Vese, *Image Decomposition and Restoration Using Total Variation Minimization and the H^{-1}-Norm*, CAM-Report 02-57, UCLA, Los Angeles, CA, 2002.

[24] P. Perona and J. Malik, *Scale-space and edge detection using anisotropic diffusion*, IEEE Trans. Pattern Analysis and Machine Intelligence, 12 (1990), pp. 629–639.

[25] L. Q. Qi and J. Sun, *A nonsmooth version of Newton's method*, Math. Program., 58 (1993), pp. 353–367.

[26] L. I. Rudin, S. Osher, and E. Fatemi, *Nonlinear total variation based noise removal algorithms*, Phys. D, 60 (1992), pp. 259–268.

[27] Y. Saad, *Iterative Methods for Sparse Linear Systems*, 2nd ed., SIAM, Philadelphia, 2003.

[28] C. Schnörr, *Unique reconstruction of piecewise-smooth images by minimizing strictly convex nonquadratic functionals*, J. Math. Imaging Vision, 4 (1994), pp. 189–198.

[29] D. Strong and T. Chan, *Edge-preserving and scale-dependent properties of total variation regularization*, Inverse Problems, 19 (2003), pp. S165–S187.

[30] D. Sun and J. Han, *Newton and quasi-Newton methods for a class of nonsmooth equations and related problems*, SIAM J. Optim., 7 (1997), pp. 463–480.

[31] C. R. Vogel, *Nonsmooth regularization*, in Inverse Problems in Geophysical Applications (Yosemite, CA, 1995), H. W. Engl, A. K. Louis, and W. Rundell, eds., SIAM, 1997, pp. 1–11.

[32] C. R. Vogel, *Computational Methods for Inverse Problems*, Frontiers Appl. Math. 23, SIAM, Philadelphia, 2002.

[33] C. R. Vogel and M. E. Oman, *Iterative methods for total variation denoising*, SIAM J. Sci. Comput., 17 (1996), pp. 227–238.

[34] J. Y. Yuan, G. H. Golub, R. J. Plemmons, and W. A. G. Cecílio, *Semi-conjugate direction methods for real positive definite systems*, BIT, 44 (2004), pp. 189–207.

SIAM J. SCI. COMPUT.
Vol. 28, No. 1, pp. 24–46
© 2006 Society for Industrial and Applied Mathematics

A MULTILEVEL NONLINEAR METHOD*

IRAD YAVNEH† AND GREGORY DARDYK†

Abstract. A multilevel nonlinear method (MNM) for the numerical solution of nonlinear partial differential equations is developed. The MNM algorithm is motivated and analyzed using a simplified model which retains the essential features of the new approach. It is thereby shown to combine the advantages of the two classical multigrid approaches to nonlinear problems. The analysis is supported by numerical tests for nonlinear differential equations in one and two dimensions.

Key words. multigrid, nonlinear partial differential equations, numerical methods for partial differential equations

AMS subject classifications. 65N55, 65N22, 65H10

DOI. 10.1137/040613809

1. Introduction.

Many problems in the sciences and engineering are modeled by nonlinear partial differential equations (PDEs), usually requiring numerical solution. Most commonly, the PDE is approximated by a finite difference, element, or volume discretization on a grid, which results in a large but sparse nonlinear algebraic system of equations. Reliable and computationally efficient solvers for such systems are essential for achieving relevant and accurate simulations.

One of the most efficient techniques for solving such large systems is the multigrid iterative method (see, for example, the classical [1, 14, 15], the expository [5], and the comprehensive [25]). In this approach, a sequence of progressively coarser grids is used to accelerate the convergence of some basic iterative process on the target grid. The basic iterative method is referred to as the "relaxation" or "smoothing" operator, as its objective is to smooth the error in the current approximation relative to the computational grid. There are two general approaches for solving discretized PDEs by multigrid methods. One is to perform a global linearization (GL), usually by Newton's method or some inexact variant thereof, and solve the resulting linear system approximately by a linear multigrid algorithm; this is then repeated iteratively. The second approach, represented by the full approximation scheme (FAS) of [1, 2] and the closely related nonlinear multigrid method (NLMG) of [14], is to perform only local linearization (LL)—in the error-smoothing process. Convergence acceleration is then provided by nonlinear coarse-grid operators.

Over the years, variants of these two general categories have been proposed and developed, usually in an ad hoc framework of a particular problem. Some recent examples of developments and implementations of GL multigrid solvers appear in [16, 17, 23, 27, 28], while LL applications can be found in, e.g., [4, 7, 11, 12, 13, 18, 19, 20, 21]. For "friendly" problems, both approaches work well and the difference in efficiency is usually not large (e.g., [25, section 5.3]). However, for more difficult problems, the two approaches often exhibit distinct behavior, with GL holding an advantage in some situations and LL in others. We are therefore motivated to develop

*Received by the editors August 24, 2004; accepted for publication (in revised form) June 27, 2005; published electronically March 3, 2006. This research was supported by the Israeli Science Foundation under grant 48/02.

http://www.siam.org/journals/sisc/28-1/61380.html

†Department of Computer Science, Technion—Israel Institute of Technology, Haifa 32000, Israel (irad@cs.technion.ac.il, gregoryd@cs.technion.ac.il).

methods which will be at least as good as the more suitable of these two approaches, and often better than both.

The algorithm we propose is first motivated and analyzed in section 2, employing a simplified scalar analogy, and then presented in detail in section 3. The algorithm has been applied recently in [6] for the problem of two-dimensional phase unwrapping. However, this was an application-focused paper comparing different methods, with no analysis and just a very brief description of the algorithm.

Multigrid methods for [illegible] problems have been studied very extensively. For many problems, robust multigrid methods (such as the classical "algebraic" [3, 22], or "black-box" [8, 9] multigrid) are well known to be effective even in the presence of discontinuous coefficients and irregular domain boundaries. This represents an advantage for GL methods, where the problem solved per iteration is linear. LL approaches, on the other hand, generally require "direct" rediscretization of the nonlinear operator on the coarse grids, which is less robust. On the other hand, the [illegible] of efficient GL methods is usually small, which means that slow global search methods must often be applied before the fast GL method can be effective. This may represent a significant disadvantage.

1.1. Classical approaches. We first review two classical GL and LL methods. A two-grid framework is used, which is later generalized to a complete multigrid algorithm by recursion. Indices h and H denote fine- and coarse-grid operators and functions, respectively. The fine grid h is the target grid, while grid H is employed to accelerate convergence. We assume that, for [illegible] fine-grid operators, we can construct [illegible] coarse-grid approximations on grid H. By "robust" we mean that, given an appropriate smoother, the multigrid solver exhibits the typical fast and mesh-independent convergence.

Let the fine-grid problem be

$$N_h u_h = f_h, \tag{1}$$

where N_h is some discrete nonlinear matrix operator, u_h is the sought solution vector, and f_h is a given vector. Using iterative methods, it is assumed that we have some current approximation to the solution, denoted $\tilde{u}_h$, which we are trying to improve. It is convenient to subtract $N_h\tilde{u}_h$ from both sides of (1), obtaining the defect equation

$$N_h u_h - N_h\tilde{u}_h = f_h - N_h\tilde{u}_h \equiv r_h. \tag{2}$$

Let K_h denote a linearization of N_h. For example, using Newton's method we define K_h to be the Jacobian matrix of N_h, computed for the current approximation:

$$K_h = N_h'[\tilde{u}_h].$$

Then, Newton's method approximates (2) by

$$K_h(u_h - \tilde{u}_h) \approx r_h. \tag{3}$$

Let K_H denote the coarse-grid approximation to the linear operator K_h, which is obtained using a robust linear coarse-grid correction method. Indeed, in our simple analysis below, it is assumed that this coarse-grid operator provides a "perfect" approximation of the linear fine-grid operator (as would be the case if total reduction is used). That is, the two-grid algorithm, with suitable error-smoothing, solves the linear fine-grid problem exactly. Let I_h^H and I_H^h denote appropriate fine-to-coarse

(restriction) and coarse-to-fine (prolongation) operators, respectively. Then, after we smooth the error in (3) with a suitable relaxation, we solve the following coarse-grid problem:

$$K_H(u_H - \bar{I}_h^H \tilde{u}_h) = I_h^H r_h, \tag{4}$$

where $\bar{I}_h^H$ is some restriction operator[1] which may be different from I_h^H. Once u_H is computed, the correction is interpolated and added to $\tilde{u}_h$:

$$\tilde{u}_h^{new} = \tilde{u}_h + I_H^h(u_H - \bar{I}_h^H \tilde{u}_h). \tag{5}$$

We can optionally follow up with additional relaxation of (3), and this completes one Newton iteration.

Next, we review the classical FAS. First, the error in the fine-grid solution is smoothed using a suitable nonlinear relaxation method (which employs only local linearization). Let $\hat{N}_H$ denote a nonlinear coarse-grid approximation to N_h, obtained by rediscretizing the nonlinear problem on the coarse grid (normally using the same discretization as that of the fine grid; this is called direct discretization). The coarse-grid problem is then

$$\hat{N}_H u_H - \hat{N}_H \bar{I}_h^H \tilde{u}_h = I_h^H r_h. \tag{6}$$

The correction to the fine-grid approximation is added as in (5).

The two approaches are substantially distinct. In the GL method we approximate N_h by its linearization, which we then proceed to approximate on the coarse grid. Using suitable linear multigrid methods, the latter approximation is excellent, and the overall robustness of the method is therefore determined, loosely speaking, by the accuracy of the linearization, which depends in turn on the properties of the Jacobian and on the closeness of $\tilde{u}_h$ to u_h. In the LL approach, on the other hand, we use a nonlinear approximation to N_h on the coarse grid. If this approximation is sufficiently accurate, the LL algorithm can yield fast convergence even with a rather poor initial guess for the fine-grid solution, thus having an advantage over the GL approach. However, if $\hat{N}_H$ approximates N_h poorly, for example due to some nonsmoothness of the coefficients, then the method might not converge even if the initial error is very small and the nonlinearity very weak.

2. A scalar analogy. To motivate and analyze our approach, we present an analogy in the form of a simple scalar problem, which contains the essential elements we are studying. This analysis can be generalized to the multivariate vector-valued case, but this would increase the complication without yielding additional insight. Consider the problem of finding a real number u satisfying

$$N(u) = f, \tag{7}$$

where N is some twice continuously differentiable nonlinear real-valued function, and f is a constant which serves to help follow the analogy. For any given $\rho > 0$, denote $D_\rho = \{z : |u - z| < \rho\}$, and assume $|N'(z)| \geq K > 0$ for all $z \in D_\rho$.

[1] Since this is a linear problem, the coarse-grid correction is independent of this transfer operator; the usual choice for linear problems is $\bar{I}_h^H = 0$, which yields the so-called correction scheme.

2.1. FAS. Denote the current approximation to u by $u_n \in D_\rho$, and the error by $e_n = u_n - u$. Suppose that there exists some nonlinear function $\hat{N}$ which approximates N and which is easy to invert ($\hat{N}$ corresponds to $\hat{N}_H$ in our analogy). Then we can define an iteration process, which is analogous to FAS, by

$$\hat{N}(u_{n+1}) - \hat{N}(u_n) = f - N(u_n) \equiv r_n. \tag{8}$$

Suppose that $\hat{N}$ is an $O(\epsilon)$ approximation to N, where ϵ is a positive small parameter. That is,

$$\hat{N}(z) \equiv N(z) + \epsilon\phi(z), \tag{9}$$

for some twice continuously differentiable function ϕ. Substituting (9) into (8), we obtain after rearrangement

$$\epsilon[\phi(u_{n+1}) - \phi(u_n)] = f - N(u_{n+1}) \equiv r_{n+1}.$$

Using the mean-value theorem and the fact that $f = N(u)$, we get

$$\epsilon\phi'(\xi_1)(e_{n+1} - e_n) = -N'(\xi_2)e_{n+1},$$

for some $\xi_1 \in (u_n, u_{n+1})$ and $\xi_2 \in (u, u_{n+1})$. Assume $|\phi'| \le P < K/2\epsilon$ in D_ρ. Then

$$\left|\frac{e_{n+1}}{e_n}\right| = \left|\frac{\epsilon\phi'(\xi_1)}{N'(\xi_2) + \epsilon\phi'(\xi_1)}\right| \le \left|\frac{\epsilon P}{K - \epsilon P}\right| < 1. \tag{10}$$

Thus, the iteration converges to u, and the asymptotic convergence behavior is given by

$$\lim_{n\to\infty}\left|\frac{e_{n+1}}{e_n}\right| = \left|\frac{\epsilon\phi'(u)}{N'(u) + \epsilon\phi'(u)}\right| = \epsilon\left|\frac{\phi'(u)}{N'(u)}\right| + O\left(\epsilon^2\right) \quad \text{for } \epsilon \to 0. \tag{11}$$

2.2. Newton's method. The Newton iteration for this problem can be written as

$$N'(u_n)(u_{n+1} - u_n) = r_n. \tag{12}$$

For a sufficiently good initial approximation Newton's method converges. In particular, if $|N''| \le R$ in D_ρ, then

$$\left|\frac{e_{n+1}}{e_n}\right| \le \left|\frac{e_n R}{2K}\right|, \tag{13}$$

and the iteration converges to u if $|e_n| < 2K/R$. The well-known quadratic asymptotic convergence behavior is given by

$$\lim_{n\to\infty}\left|\frac{e_{n+1}}{e_n^2}\right| = \left|\frac{N''(u)}{2N'(u)}\right|. \tag{14}$$

2.3. Multilevel nonlinear method (MNM). Equations (10)–(11) and (13)–(14) demonstrate the advantage and disadvantage of each approach. Newton's method converges extremely fast when we have a good initial approximation ($|e_n|$ sufficiently small). FAS, on the other hand, will have a large attraction basin if $\hat{N}$ is a good

approximation to N, though it will not be useful when $\hat{N}$ approximates N poorly. It is generally far less dependent on a good initial approximation than Newton's method.

We now devise a method that has the advantages of both fast asymptotic convergence and a large attraction basin. First, we subtract $N(u_n)$ from both sides of (7) to obtain the defect equation,

$$N(u) - N(u_n) = r_n. \tag{15}$$

The key idea is to split the left side of (15) into a relatively large linear part and a small nonlinear part and use the $\hat{N}$ approximation only for the nonlinear part. Accordingly, we add and subtract $N'(u_n)(u - u_n)$ in the left side of (15). This yields

$$N'(u_n)(u - u_n) + F\,[\,N(u), N(u_n), N'(u_n), u, u_n\,] = r_n, \tag{16}$$

where

$$F\,[\,N(u), N(u_n), N'(u_n), u, u_n\,] = N(u) - N(u_n) - N'(u_n)(u - u_n). \tag{17}$$

Note that (16) is equivalent to the original problem (7), but the nonlinear term F (which is nothing but the remainder term of a truncated Taylor series) is only of size $O(|u - u_n|^2)$, while the first term and the right side are $O(|u - u_n|)$. We use the nonlinear approximation only for the small nonlinear part. That is, we define the iteration by

$$N'(u_n)(u_{n+1} - u_n) + F\big[\,\hat{N}(u_{n+1}), \hat{N}(u_n), \hat{N}'(u_n), u_{n+1}, u_n\,\big] = r_n. \tag{18}$$

Leaving only u_{n+1} terms on the left side we obtain

$$\begin{aligned} N'(u_n)u_{n+1} + \hat{N}(u_{n+1}) - \hat{N}'(u_n)u_{n+1} = r_n + N'(u_n)u_n \\ + \hat{N}(u_n) - \hat{N}'(u_n)u_n. \end{aligned} \tag{19}$$

Assume that we can easily invert the left side of (19) and solve for u_{n+1}. To analyze the convergence behavior of this iteration, we first use (9) to obtain, after rearrangement,

$$\epsilon\,\{\,[\phi(u_{n+1}) - \phi(u_n)] - \phi'(u_n)(u_{n+1} - u_n)\,\} = r_{n+1}. \tag{20}$$

Taylor series expansions now yield

$$\frac{\epsilon}{2}\phi''(\xi_1)(e_{n+1} - e_n)^2 = -e_{n+1}N'(\xi_2) \tag{21}$$

for some $\xi_1 \in (u_n, u_{n+1})$ and $\xi_2 \in (u_{n+1}, u)$. Denote

$$\mu = -\frac{N'(\xi_2)}{\epsilon e_n \phi''(\xi_1)}.$$

(If the denominator here is zero, then we have immediate convergence by (21), so we exclude this case.) Solving the quadratic equation (21) for the error reduction factor we obtain two solutions, the relevant one being

$$\frac{e_{n+1}}{e_n} = \mu + 1 - \mathrm{sign}(\mu)\,\sqrt{\mu^2 + 2\mu}. \tag{22}$$

TABLE 1
Comparison of convergence behavior for the scalar problem, under the assumptions $|N'| \geq K > 0$, $|N''| \leq R$, $|\phi'| \leq P$, $|\phi''| \leq Q$ *in* D_ρ.

Method	$\lim_{n\to\infty} \left\lvert \frac{e_{n+1}}{e_n^p} \right\rvert$	p	Sufficient for convergence
FAS	$\left\lvert \frac{\epsilon\phi'(u)}{N'(u)} \right\rvert$	1	$\left\lvert \frac{2\epsilon P}{K} \right\rvert < 1$
Newton	$\left\lvert \frac{N''(u)}{2N'(u)} \right\rvert$	2	$\left\lvert \frac{e_n R}{2K} \right\rvert < 1$
MNM	$\left\lvert \frac{\epsilon\phi''(u)}{2N'(u)} \right\rvert$	2	$\left\lvert \frac{2\epsilon e_n Q}{K} \right\rvert < 1$

(We assume here that we know how to choose the relevant solution when we solve (19) for u_{n+1}. In practice, this is not a problem when the iteration is relevant, since u_n provides a good initial approximation to the correct u_{n+1}.) A real solution to (22) exists if $\mu \notin (-2, 0)$. Furthermore, $|e_{n+1}/e_n| < 1$ for $\mu \notin [-2, 0]$. That is, the error is reduced by the iteration essentially whenever (19) has a real solution. In particular, if $|\phi''| \leq Q$ in D_ρ, then a sufficient condition for convergence of the new algorithm is

$$|\epsilon e_n| < \frac{K}{2Q}. \tag{23}$$

Furthermore, if the iteration converges to u, then the asymptotic convergence behavior is given by

$$\lim_{n\to\infty} \left\lvert \frac{e_{n+1}}{e_n^2} \right\rvert = \frac{\epsilon}{2} \left\lvert \frac{\phi''(u)}{N'(u)} \right\rvert + O\left(\epsilon^2\right) \quad \text{for } \epsilon \to 0. \tag{24}$$

We summarize the results of the scalar analysis in Table 1. These results are encouraging inasmuch as they show that the proposed method, which is analogous to the MNM described in detail below, enjoys both the quadratic convergence rate of Newton's method and the large attraction basin suggested by the ϵ coefficient. Indeed, the convergence rate of MNM is similar to the product of the convergence rates of FAS and Newton's method.

2.3.1. Backtracking. Although we expect to see a large attraction basin of the MNM, compared to Newton's method, there will be occasions where ϵ is not sufficiently small to yield convergence for a given initial guess. A well-known method for ensuring global convergence of Newton's method is backtracking [10]. While a Newton step does not ensure that the residual r_n will be reduced, the [illegible] of the Newton step is always one of descent. Thus, if we [illegible] the Newton correction by some sufficiently small positive scalar $0 < \lambda < 1$, we are assured that the residual will be reduced. Alternatively, instead of damping the correction step, we can damp the residual, by multiplying the right side of (12) by λ. Since this is a linear equation,

damping the right side is equivalent to damping the correction. We obtain from the damped-residual version of (12)

$$u_{n+1} - u_n = \frac{\lambda r_n}{N'(u_n)}. \tag{25}$$

Therefore we have, by the mean value theorem and (25),

$$\begin{aligned} r_{n+1} &= r_n - N'(\xi)(u_{n+1} - u_n) \\ &= r_n \left(1 - \lambda \frac{N'(\xi)}{N'(u_n)}\right) \end{aligned} \tag{26}$$

for some $\xi \in (u_n, u_{n+1})$. By (25), (26), and the continuity of N' (and the fact that it is bounded away from zero in D_ρ), there always exists a positive λ_{max} such that

$$\left|\frac{r_{n+1}}{r_n}\right| < 1 \quad \text{for every } \lambda \in (0, \lambda_{max}). \tag{27}$$

For MNM, damping the correction is no longer equivalent to damping the residual, since it is a nonlinear method. Indeed, as seen above, there might not even be a solution to the nonlinear iteration, hence no "direction" is defined. However, we show now that MNM also yields a "descent direction" if we backtrack using residual damping. This approach is in the spirit of the NLMG of [14], where residual damping is introduced in conjunction with a compensating overcorrection. Suppose that we multiply r_n in (16), (18), and (19) by $\lambda \in (0, 1)$. We then get from (19), denoting $\delta = u_{n+1} - u_n$ and using a Taylor series expansion,

$$\frac{1}{2}\hat{N}''(\xi_1)\delta^2 + N'(u_n)\delta = \lambda r_n \tag{28}$$

for some $\xi_1 \in (u_n, u_{n+1})$. If $\hat{N}''(\xi_1) = 0$, then (25)–(26) hold. Otherwise, δ is given by

$$\delta = \frac{N'(u_n)}{\hat{N}''(\xi_1)} \left(-1 \pm \sqrt{1 + 2\lambda r_n \hat{N}''(\xi_1)/N'(u_n)^2}\right). \tag{29}$$

Only the solution with the plus sign in front of the square root is relevant, and we assume that our nonlinear solver can find this solution. (In practice this is not a problem, as u_n provides an arbitrarily good initial guess for the correct u_{n+1} as λ is reduced.) We thus have

$$\delta = \frac{\lambda r_n}{N'(u_n)} + O\left(\lambda^2 r_n^2 \hat{N}''(\xi_1)/N'(u_n)^3\right), \quad \lambda r_n \to 0. \tag{30}$$

Now, similarly to (26),

$$\begin{aligned} r_{n+1} &= r_n - N'(\xi)(u_{n+1} - u_n) \\ &= r_n \left(1 - \lambda \frac{N'(\xi)}{N'(u_n)} + O\left(\lambda^2 r_n^2 \hat{N}''(\xi_1)/N'(u_n)^3\right)\right) \quad \text{for } \lambda r_n \to 0, \end{aligned} \tag{31}$$

so here too we can always choose some sufficiently small positive λ_{max} such that (27) holds. Indeed, the residual reduction factor of damped MNM tends to that of the damped Newton step as λ tends to zero. This is expected, as MNM can be viewed as a nonlinear correction to Newton's method, with the correction tending to zero with the damping parameter (cf. section 3.2).

3. MNM: A multilevel nonlinear method. We return now to the actual problem and the multigrid algorithm. Let $\hat{K}_H$ denote the linearization of $\hat{N}_H$. Then, analogously to (19), we define the coarse-grid problem as

$$N_H u_H = f_H, \tag{32}$$

with

$$N_H = K_H + \hat{N}_H - \hat{K}_H \tag{33}$$

and

$$f_H = I_h^H r_h + N_H \bar{I}_h^H \tilde{u}_h. \tag{34}$$

Note that K_H and $\hat{K}_H$ are analogous to $N'(u_n)$ and $\hat{N}'(u_n)$, respectively. That is, we are assuming that K_H is an excellent approximation to K_h, while $\hat{K}_H$ might not be as good. This makes sense because, as noted above, there are well-known methods that provide excellent coarse-grid approximations to linear fine-grid operators, whereas direct discretization often does not when coefficients are nonsmooth.

3.1. The MNM algorithm. The multigrid algorithm, which we describe next, is obtained from the two-grid algorithm by the usual recursion. Specific details, such as the relaxation method, intergrid transfers, and algorithmic parameters, can vary. In sections 4.1 and 4.2 we specify the particular choices used in our numerical tests.

A hierarchy of grids is defined and numbered 0 to M, where grid 0 is the target grid, which is the finest one, and grid M is the coarsest. On each level (grid) j we solve approximately a problem of the form

$$N_j u_j + L_j u_j = f_j, \tag{35}$$

where u_j is the vector of unknowns, f_j is a given vector, N_j—like $\hat{N}_H$ above—is a grid-j discretization of N (as in the classical FAS algorithm), and L_j is a linear correction operator (similar to $K_H - \hat{K}_H$ above). We specify the usual integer parameters, ν_1 and ν_2, which are the number of relaxation sweeps carried out before and after the coarse-grid correction, respectively, and the cycle index γ, which specifies the number of recursive calls (1 for a so-called V cycle, 2 for a W cycle, and so on.)

Initially, u_0 is set to some approximation (which is normally obtained using the so-called full multigrid (FMG) approach [1]), and L_0 is the zero matrix of appropriate size. Then, the MNM cycle, described below, is called repeatedly for $j = 0$, until some convergence criterion is met.

function $u_j = \mathrm{MNM}(u_j, N_j, L_j, f_j, j, \gamma, \nu_1, \nu_2, M)$

If $(j < M)$ ** On every grid but the coarsest **
{
Relax ν_1 times the equation,
$N_j u_j + L_j u_j = f_j$
Compute the residual,
$r_j = f_j - (N_j u_j + L_j u_j)$
Construct the linearized operator,
$K_j = L_j + N'_j[u_j]$
Initialize the coarse-grid solution,
$u_{j+1} = \bar{I}_j^{j+1} u_j$
Compute the robust coarsening of the linearized operator,
$K_{j+1} = I_j^{j+1} K_j I_{j+1}^j$
Compute L_{j+1} by subtracting off the linearization of N_{j+1},
$L_{j+1} = K_{j+1} - N'_{j+1}[u_{j+1}]$
Compute the coarse-grid right side,
$f_{j+1} = I_j^{j+1} r_j + N_{j+1} u_{j+1} + L_{j+1} u_{j+1}$
Call MNM recursively (repeat γ times),
$u_{j+1} = \mathrm{MNM}(u_{j+1}, N_{j+1}, L_{j+1}, f_{j+1}, j+1, \gamma, \nu_1, \nu_2, M)$
}
else ** On the coarsest grid **
{
Solve approximately the equation,
$N_M u_M + L_M u_M = f_M$
Return u_M
}
end if

Interpolate and add the correction,
$u_j = u_j + I_{j+1}^j (u_{j+1} - \bar{I}_j^{j+1} u_j)$
Relax ν_2 times the equation,
$N_j u_j + L_j u_j = f_j$
Return u_j

The restriction and prolongation operators, denoted here by I_j^{j+1} and I_{j+1}^j, respectively, are chosen so as to yield a robust coarse-grid approximation of the linear operator, K_j. These can be any of several matrix-dependent transfer operators that have been developed over the years. In our two-dimensional (2D) tests we use the black-box multigrid method of [8]. For the restriction of u_j we use the operator $\bar{I}_j^{j+1}$, which can be simple injection or some local averaging. Observe that the combined operator, $N_j + L_j$, is nothing but an approximation on level j of the fine-grid operator, N_0. Thus, the relaxation methods used in the classical FAS algorithm, such as Gauss–Newton, are suitable here as well. It is important to note that, since we are combining two different approximations to the fine-grid operator (for example, when computing the linear correction, L_j), they need to be scaled compatibly. We explain this issue and demonstrate the algorithm with an explicit simple example in the next subsection. (See also comments regarding scaling in section 4.1.)

3.1.1. A concrete example. By way of illustration, we now show an explicit example of the MNM cycle with two grids. Capital subscripts correspond to coarse-grid values and operators. Consider the one-dimensional (1D) nonlinear problem,

$$Nu = -u_{xx} - Ce^u = f, \tag{36}$$

with some appropriate boundary conditions, which are not important for this discussion. Here, C is a positive constant and $f(x)$ is given. (With $f = 0$ and zero Dirichlet boundary conditions, this is the classical Bratu problem.) Standard second-order finite-difference discretization on a grid of mesh size h produces a nonlinear system,

$$N_h u_h = f_h, \tag{37}$$

with the ith row of $N_h u_h$ given by

$$(N_h u_h)_i = h^{-2}\left[-u_{i-1} + 2u_i - h^2 Ce^{u_i} - u_{i+1}\right]. \tag{38}$$

Here and below, we omit the h subscript when using subscripts to denote row number (equivalently, mesh point). Let $\tilde{u}_h$ denote the current approximation to u_h. Employing Newton linearization, we obtain $K_h = N_h'(\tilde{u}_h)$, where the ith row of $K_h u_h$ can be written as

$$(K_h u_h)_i = h^{-2}\left[-u_{i-1} + u_i(2 - \sigma_i) - u_{i+1}\right], \tag{39}$$

with

$$\sigma_i = h^2 Ce^{\tilde{u}_i}.$$

Evidently, K_h approximates the linear operator,

$$K = -\partial_{xx} - Ce^{\tilde{u}}.$$

Assume standard coarsening, with coarse mesh point I coinciding with fine mesh point $i = 2I$. As in FAS, we need to transfer the current approximation, $\tilde{u}_h$, to the coarse grid and construct the nonlinear coarse-grid operator, $\hat{N}_H$, where $H = 2h$ denotes the coarse mesh interval. Assume we use injection for this transfer,

$$\tilde{U}_H = \bar{I}_h^H \tilde{u}_h,$$

with $\tilde{U}_I = \tilde{u}_i$. The operator $\hat{N}_H$ is obtained by rediscretization on the coarse grid. However, we introduce a multiplicative scaling, to be discussed below. The Ith row of $\hat{N}_H U_H$ is thus given by

$$\left(\hat{N}_H U_H\right)_I = B_I H^{-2}\left[-U_{I-1} + 2U_I - H^2 Ce^{U_I} - U_{I+1}\right], \tag{40}$$

where B_I is the scaling. The linearization of $\hat{N}_H$ is denoted by $\hat{K}_H$, with the Ith row of $\hat{K}_H U_H$ given by

$$\left(\hat{K}_H U_H\right)_I = B_I H^{-2}\left[-U_{I-1} + U_I(2 - \Sigma_I) - U_{I+1}\right], \tag{41}$$

with

$$\Sigma_I = H^2 Ce^{\tilde{U}_I} = 4\sigma_i.$$

Next, we turn to the construction of the robust coarse-grid approximation to K_h. The appropriate prolongation operator (see discussion in section 4) is such that the interpolated function, $v_h = I_H^h U_H$, satisfies the homogeneous equation, $(K_h v_h)_i = 0$, at odd i:

$$\left(I_H^h U_H\right)_i = \begin{cases} U_{i/2} & \text{if } i \text{ is even,} \\ \frac{U_{(i-1)/2}+U_{(i+1)/2}}{2-\sigma_i} & \text{if } i \text{ is odd.} \end{cases} \tag{42}$$

As the problem is symmetric, we choose the restriction to be the transpose of the prolongation. Therefore, the Galerkin coarsening rule yields the robust coarse-grid approximation to K_h:

$$K_H = \left(I_H^h\right)^T K_h I_H^h.$$

Observe that the restriction, I_H^h, appears as a factor both in K_H and in the right-hand side of the coarse-grid equation (since it is used to restrict the residuals). Hence, its scaling is unimportant with respect to K_H, since it cancels out. In contrast, when we construct $\hat{K}_H$ by direct discretization, we are implicitly assuming that the rows of the restriction sum up to one (hence, that a constant function remains invariant when restricted to the coarse grid). Since in MNM we combine the different operators, we must therefore scale K_H and N_H for consistency by multiplying them by the row-sums of the restriction operator. For $I = i/2$, this yields

$$B_I = 1 + \frac{1}{2-\sigma_{i-1}} + \frac{1}{2-\sigma_{i+1}} = \frac{2+\delta^- + \delta^+ - \sigma_i}{2-\sigma_i},$$

where

$$\delta^{\pm} = \frac{2-\sigma_i}{2-\sigma_{i\pm1}}.$$

For the Galerkin coarse-grid linear operator, an explicit computation yields

$$\begin{aligned}&(K_H U_H)_I \\ &\quad = B_I \left(\frac{4}{2+\delta^- + \delta^+ - \sigma_i}\right) H^{-2} \left\{-\delta^- U_{I-1} + \left[-\delta^- - \delta^+ + (2-\sigma_i)^2\right] U_I - \delta^+ U_{I+1}\right\}.\end{aligned} \tag{43}$$

The linear correction to $\hat{N}_H$ is given by $L_H = K_H - \hat{K}_H$, and the complete MNM coarse-grid operator is given by

$$N_H = \hat{N}_H + L_H = N_H + K_H - \hat{K}_H, \tag{44}$$

with $\hat{N}_H$, K_H, and $\hat{K}_H$ given by (40), (43), and (41), respectively.

To get an impression of the character of L_H, observe that if u_h is smooth (that is, undergoes a relative change of order h from one grid point to the next), then $\delta^{\pm} = 1 + O(h^3)$ (because σ_i itself is of order h^2). Substituting 1 for $\delta^{\pm}$, we obtain by a straightforward computation

$$(L_H U_H)_I = B_I H^{-2} \frac{\sigma_i}{4-\sigma_i} \left(-U_{I-1} + 2U_I - U_{I+1}\right).$$

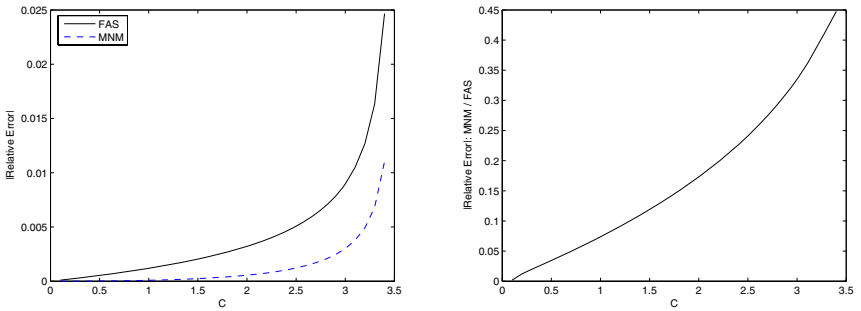


FIG. 1. *The maximal relative difference between the fine-grid solution to the* 1*D Bratu problem and two coarse-grid solutions, one using* $\hat{N}_H$ *(FAS) and the other using* N_H *(MNM). The relative differences appear on the left and their ratio on the right.*

Note that the correction L_H increases somewhat the relative size of the Laplacian in the nonlinear operator. This turns out to compensate for the slight extra "stiffness" of the coarse grid compared to the fine grid. Thus, it improves the nonlinear coarse-grid approximation to the fine-grid operator, as demonstrated below. Observe that the linear correction is normally very small, since K_H and $\hat{K}_H$ are both $O(h^2)$ approximations to the scaled K. It becomes large only when $\sigma_i \sim 1$. Furthermore, L_H is "very slightly" nonsymmetric due to the spatially varying scaling factor, B_I. This may become significant on coarse grids. We may choose to retain the symmetry by fixing the scaling factor to be some constant nominal value.

To demonstrate the effect of the linear correction, L_H, we perform a simple numerical test. We set f and the boundary conditions to zero, and the domain to $(0, 1)$, and solve the problem on a grid of 16 mesh intervals—our fine grid—for a range of positive C values. It is well known that solutions for the differential problem exist only for C that is below some critical value, and the problem becomes more and more difficult as C approaches this value. Next, we approximate and solve the same problem on a coarse grid of eight intervals, once using $\hat{N}_H$ (FAS) and once using N_H (MNM), employing the previously determined fine-grid solution to compute L_H. We compare in Figure 1 the maximal difference between the fine-grid solution and the two coarse-grid solutions. Evidently, the MNM coarse-grid operator provides a much better approximation. The relative gain in accuracy is very large when the parameter C is small (because the linearizations then yield excellent approximations), and it becomes smaller and smaller, but still quite significant, as the critical turning point is approached.

3.1.2. Computational cost. The number of operations per V cycle is proportional to the number of unknowns in all three algorithms—FAS, Newton, and MNM. The actual cost per cycle for each of the methods depends on the problem and on the implementation. FAS requires evaluating the nonlinear operator for every relaxation sweep and for residual computation. Newton's method requires only one linearization per cycle, which is comparable in cost to an evaluation of the nonlinear operator, and the resulting linear operator can then be stored. On the other hand, after each such linearization, a setup of the robust linear solver with its matrix-dependent intergrid transfers and (Petrov–)Galerkin coarse-grid operators needs to be applied. The MNM algorithm is essentially a combination of the two, starting from the second-finest grid. An MNM cycle is therefore more expensive than either an FAS or a Newton iteration,

but it is less expensive than the sum of the two. In particular, on the finest grid, where normally most of the work is performed, the MNM and FAS algorithms are identical because there is no linear correction term.

3.1.3. Backtracking. We introduce an optional backtracking procedure into the basic MNM algorithm, both for the local (scalar, as we use nonlinear pointwise Gauss–Seidel relaxation) nonlinear problems encountered in the relaxation, and for the coarse-grid correction. In the local problems we damp the Newton-step corrections in the classical manner, while for the coarse-grid corrections we backtrack via residual damping as described in subsection 2.3.1. We basically use the classical backtracking procedure of [10] with the following changes: (1) We do not compute the condition number and redefine the problem for poorly conditioned problems; (2) We limit the number of attempts to reduce the residual by backtracking; if unsuccessful, the correction is simply discarded. The reason for these changes is that we do not want to spend much effort to reduce the residual of coarse-grid problems when this is unlikely to contribute much to the convergence of the fine-grid problem. The greatest difficulties occur on the coarser grids (because, in our analogy, ϵ is larger there), and it is often not cost-effective to insist on residual reduction on the very coarse grids, where there might not even be a solution in some cases. As the fine-grid residual is reduced, the convergence on all the grids normally becomes much easier, consistent with our analysis.

The backtracking procedure naturally adds to the cost of the algorithm, but this increase is rarely substantial. In the relaxation, the main increase normally comes from the need to recompute the residuals of the local problems to check at which of the equations (if any) backtracking is required. Usually, there are few such locations, if any, and the backtracking procedure is implemented only for those equations. For the coarse-grid correction process (which backtracks by residual damping), there is a potential for a greater increase in cost, because if the coarse-grid correction fails to reduce the residual norm then we discard it and recompute the coarse-grid correction with a damped residual, etc. As we shall show, the actual increase in the operation count in our tests was small.

The measure we use to describe the actual complexity of the MNM algorithm with backtracking is the [illegible], $\bar{\gamma}$. Recall that the cycle index γ is the number of recursive calls to function MNM in the standard algorithm (with no backtracking). Hence, the number of visits to grid j is γ^j. Due to the backtracking, the number of visits may be larger, resulting in an effective cycle index that is larger than the nominal one. Suppose that the actual total number of relaxation sweeps performed on grid j in a particular computation, divided by the number of sweeps carried out on the finest grid, is k_j, and that the number of variables on grid j is n_j. Then the effective cycle index is naturally defined implicitly as the only positive solution of

$$\sum_{j=0}^{M} n_j \bar{\gamma}^j = \sum_{j=0}^{M} k_j n_j. \tag{45}$$

In particular, if $n_j/n_{j+1} \approx \tau > 1$, where the coarsening ratio τ is independent of j, then

$$\bar{\gamma} \approx \frac{\sum_{j=0}^{M-1} k_{j+1} \tau^{-j}}{\sum_{j=0}^{M} k_j \tau^{-j}}. \tag{46}$$

In this case, the bound (2.4.14) of [25, p. 51] for the total computational work per cycle holds approximately: if the fine-grid work is Cn_0, then the total work is approximately

$$W \approx \frac{\tau C n_0}{\tau - \bar{\gamma}} \quad \text{for } \bar{\gamma} < \tau. \tag{47}$$

In a parallel computing environment, the standard backtracking procedure is relatively more troubling, since it is done sequentially. In this case we would prefer to compute in parallel several coarse-grid solutions—with different residual-damping parameters ($\lambda = 1, 1/2, 1/4, \ldots$, for example)—and choose the correction with the largest damping parameter which still yields a reduction of the residual norm, discarding the rest.

3.2. Adaptive MNM. An interesting alternative (or addition) to backtracking, is a more general adaptive procedure. Consider the MNM coarse-grid operator N_H of (33). Two points of view suggest themselves: (a) The first term, K_H, is the main term, while the second and third terms comprise a nonlinear correction for the defect which results from approximating N_h by the usual Newton linearization; (b) The second term, $\hat{N}_H$, is the main term, while the first and third terms form a linearized correction for the defect resulting from the nonrobustness of the FAS approximation. Normally, these corrections improve the coarse-grid approximation. Suppose, however, that there is somewhere a discontinuity in a coefficient. This may show up as a large linear correction, $K_H - \hat{K}_H$, because $\hat{K}_H$ is no longer a good approximation to K_h. This indicates that $\hat{N}_H$ itself is a poor approximation to N_h, and therefore we might be better off not using the FAS contribution. We should then damp (perhaps completely) the [illegible] correction, reducing to Newton's method. On the other hand, suppose that due to some nonlinear near-singularity the linearization provides a poor approximation. This would show up as a large nonlinear correction. We might then be better off using FAS in this case, and we should damp the [illegible] correction. Accordingly, we introduce two nonnegative parameters, a and b, and redefine N_H as

$$N_H = aK_H + b\hat{N}_H + (1 - a - b)\hat{K}_H. \tag{48}$$

We thus obtain MNM for $a = b = 1$, Newton's method for $a = 1$, $b = 0$, and FAS for $a = 0$, $b = 1$. In fact, a and b can be diagonal matrices instead of scalars, so that the damping is chosen locally.

Although the details are quite different, this use of parameters to obtain favorable "compromises" between different types of coarse-grid operators is in the spirit of the nonlinear multigrid approach of [14]. There, the coarse-grid operator varies from that of FAS—when the (multiplicative) parameter is equal to one, to an operator that is linearized around the initial coarse-grid approximate solution—when it is equal to zero. This effect is somewhat similar to what we obtain in the present approach as we vary (a, b) from $(0, 1)$ to $(1, 0)$ if we use a fixed-point linearization.

We applied a version of adaptive MNM in [6], fixing $b = 1$ and optimizing $a \in [0, 1]$ over the convergence factor at very coarse grids. This approach proved quite useful, but we expect that a more refined analysis will allow us to yield still greater dividends from the adaptive approach. We do not pursue this further here; however, in section 4.2 we demonstrate the potential of this approach by manually selecting appropriate values of a and b for problems which prove to be difficult for the standard MNM, even with backtracking.

4. Numerical tests.

4.1. 1D tests. We first test our algorithm on the following nonlinear 1D diffusion problem:

$$
\begin{aligned}
Nu \equiv -\left[\, g(x,u,u_x)u_x \,\right]_x &= f(x), \quad x \in (0,1),\\
u(0) &= u_\ell,\\
u(1) &= u_r,
\end{aligned}
\tag{49}
$$

where u is the unknown function, and g, f, u_ℓ, u_r are given. Discretizing by second-order accurate finite differences on a uniform grid $x_h = \{x_i\}_{i=0}^n$, with mesh size $= 1/n$, we obtain the nonlinear system

$$
N_h u_h = f_h, \tag{50}
$$

where the ith equation in the system is given by

$$
-\frac{g_{i+1/2}(x_h, u_{i+1}, u_i)[u_{i+1} - u_i] - g_{i-1/2}(x_h, u_i, u_{i-1})[u_i - u_{i-1}]}{h^2} = f_i, \quad i = 1, \ldots, n-1. \tag{51}
$$

Here, u_i and f_i are the ith components of u_h and f_h, respectively, and $g_{i\pm1/2}$ is some discrete approximation to g midway between grid points i and $i \pm 1$. The boundary conditions yield $u_0 = u_\ell$ and $u_n = u_r$. In our test problems we define g to correspond to the so-called van Genuchten curve modeling hydraulic conductivity of unsaturated soils [26]. This problem, in two dimensions, has been studied extensively by Starke in [23, 24] employing a least-squares finite elements approach with, in the latter paper, a GL multilevel approach, and by Korsawe and Starke in [18], where three LL methods (variants of FAS) are compared.

Given constant parameters k_s, α, and p, we have

$$
g = g(u) = \begin{cases} 1, & u \geq 0,\\ k_s \psi(|u|), & u < 0, \end{cases} \tag{52}
$$

where

$$
\psi(\theta) = [1 + (\alpha\theta)^p]^{-\frac{q}{2}} \left(1 - \frac{(\alpha\theta)^{p-1}}{[1 + (\alpha\theta)^p]^q}\right)^2, \tag{53}
$$

with $q = 1 - 1/p$. To represent $g_{i\pm1/2}$ we compute $g(u_i)$ and $g(u_{i\pm1})$ and use simple averaging. For a given domain and boundary conditions, this problem generally becomes more difficult for smaller p and larger α. For $p < 2$, g' fails to satisfy a Lipschitz condition at $u = 0$, and $g(u)$ actually tends to a step function for $p \to 1^+$. For α that is not small compared to one, the solution exhibits a thin boundary layer near $x = 0$ (in our example), which narrows quickly as α grows. The boundary conditions are $u_\ell = -2$, $u_r = 1$, and we set $f = 0$. The initial guess for the fine-grid solution is the linear profile which satisfies the boundary conditions. We perform V cycles until the ℓ_2 residual norm is reduced to at most 10^{-8} times its initial value. We backtrack if the coarse-grid correction plus the (single) postrelaxation sweep has failed to reduce the ℓ_2 norm of the residual. In this case, we discard the coarse-grid correction, damp the residual according to the strategy of [10], and compute a new coarse-grid correction.

For error smoothing we use Gauss–Seidel relaxation in red-black ordering, whereby we first relax at the even-indexed points and then at the odd-indexed points. To

describe the prolongation operator, let the n_j by n_j matrix, K_j, denote the linearized operator on grid j. Then the prolongation operator is a matrix P (denoted I_{j+1}^j above) of size n_j by n_{j+1}, with

$$P_{i,k} = \begin{cases} 1 & \text{if } i = 2k, \\ -\frac{K_{i,i\mp1}}{K_{i,i}} & \text{if } i = 2k \pm 1, \\ 0 & \text{otherwise.} \end{cases} \tag{54}$$

The restriction operator (denoted I_j^{j+1} above) is a matrix R of size n_{j+1} by n_j, whose (j,i)th element is given by (54), with K_j replaced by its transpose. With these choices and the construction of the coarse-grid operator by Petrov–Galerkin coarsening, K_j remains a tridiagonal matrix on all the grids. Furthermore, for linear problems, the multigrid cycle is an exact solver (equivalent to total reduction), provided that ν_1 or ν_2 is nonzero and the problem on the coarsest grid is solved exactly. This is a well-known result of the fact that, after we relax on the fine grid at the odd-indexed points—leaving there zero residuals—the remaining error is in the range of the prolongation defined by (54).

As explained in subsection 3.1.1, it is important to scale the different coarse-grid approximations which are added/subtracted to be compatible. The usual FAS algorithm assumes that rows of the restriction matrix sum up to 1. Therefore, in the MNM algorithm of section 3.1, the correct scaling of N_{j+1} (hence also N'_{j+1}) is by left-multiplication by a diagonal matrix whose elements are the row-sums of the restriction matrix, I_j^{j+1}. Such a rescaling can result in the loss of symmetry. We choose to simply scale by the constant 2, which is correct for the standard full-weighting, and is an excellent approximation on all the grids for the actual restriction.

In all our tests the finest grid is 256 intervals. For the nonlinear algorithms (MNM and FAS) we use seven multigrid levels, coarsening down to a grid of three interior variables. On the coarsest grid we relax 10 times. We use $\gamma = 1$ (V cycle) and $\nu_1 = \nu_2 = 1$. The FAS algorithm is the standard one (except that we also allow backtracking), with injection for $\bar{I}_h^H$ (local averaging is sometimes better for FAS, but not dramatically so in our tests).

We show results for physical parameters $\alpha = 0.5, 1.0$, and $p = 2.5, 1.8$, in Figures 2, 3, 4, and 5, with NWT denoting Newton's method. In Figure 2, it is evident that MNM indeed behaves as predicted by the analysis. Its asymptotic convergence rate is similar to that of NWT, but it gets there much sooner. In the moderate case (top left), $\alpha = 0.5$, $p = 2.5$, all three methods perform well. For $\alpha = 1.0$, $p = 2.5$ (bottom left) there is a rather thin boundary layer near $x = 0$, as seen in Figure 3. This nonsmooth solution causes FAS convergence to slow down significantly, while the other two methods remain effective. For $\alpha = 0.5$, $p = 1.8$ (top right), on the other hand, the solution is rather smooth but there is a strong singularity in the derivative of g at $u = 0$ (see Figure 5). As a result, the linearization yields a very poor approximation at that point, and NWT stagnates, but FAS is hardly affected. MNM has a slower start, but soon regains its fast convergence. Here, it is evident that MNM could gain much from an adaptive procedure as outlined in section 3.2, and implemented in [6]. For $\alpha = 1.0$, $p = 1.8$ (bottom right), there is both a boundary layer in u and a singularity of g', hence both FAS and NWT suffer, yet MNM remains very effective. In more extremely singular cases (smaller p), FAS becomes better than MNM due to the extreme poorness of the linear approximation. In such cases, adaptive MNM would be highly advantageous.

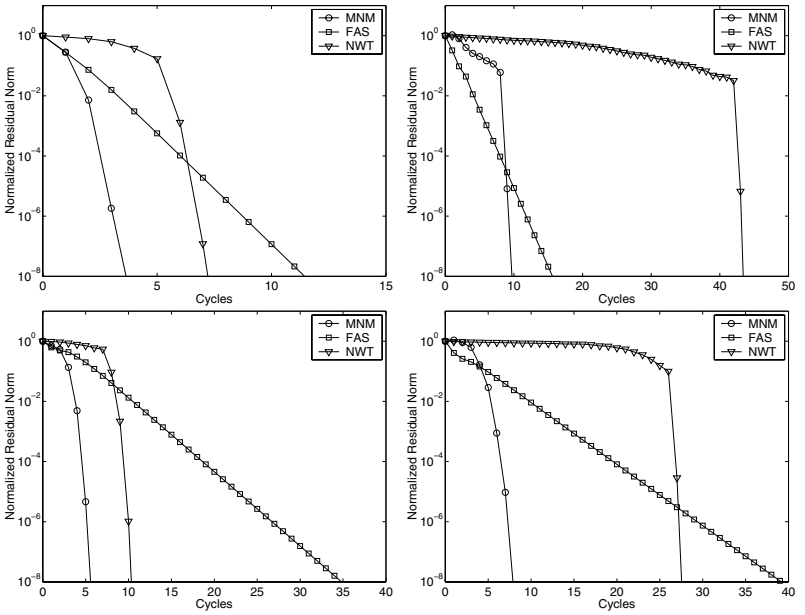


FIG. 2. *Residual norm convergence histories, normalized by the initial value, are shown for* $\alpha = 0.5$, $p = 2.5$ *(top left);* $\alpha = 0.5$, $p = 1.8$ *(top right);* $\alpha = 1.0$, $p = 2.5$ *(bottom left);* $\alpha = 1.0$, $p = 1.8$ *(bottom right).*

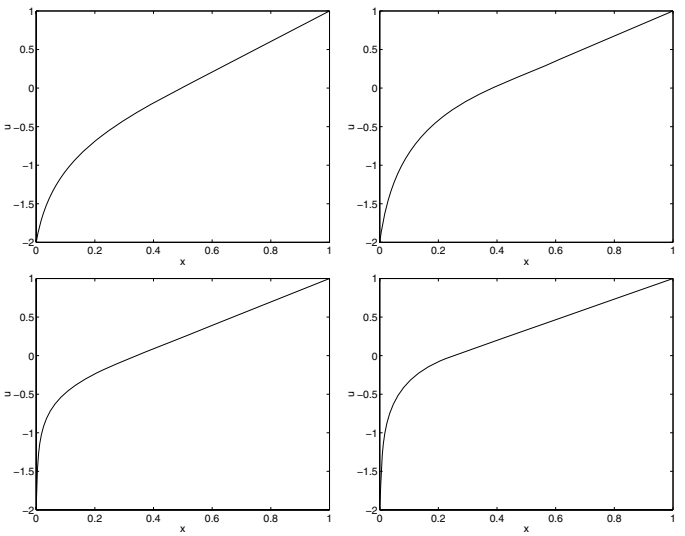


FIG. 3. *The numerical solution is shown for* $\alpha = 0.5$, $p = 2.5$ *(top left);* $\alpha = 0.5$, $p = 1.8$ *(top right);* $\alpha = 1.0$, $p = 2.5$ *(bottom left);* $\alpha = 1.0$, $p = 1.8$ *(bottom right).*

In the examples where $p = 2.5$, MNM used no backtracking, so $\bar{\gamma} = \gamma = 1$. For $\alpha = 1.0$, $p = 1.8$, we obtained $\bar{\gamma} = 1.11$, indicating an average increase of about 12% in the number of floating-point operations, compared to V cycles, by (47). For $\alpha = 0.5$, $p = 1.8$, we observed $\bar{\gamma} = 1.19$, implying a 23% average increase in computational cost, but there were also two cases of backtracking on the finest grid, so the overall work for MNM is about 35% more than is suggested by Figure 2.

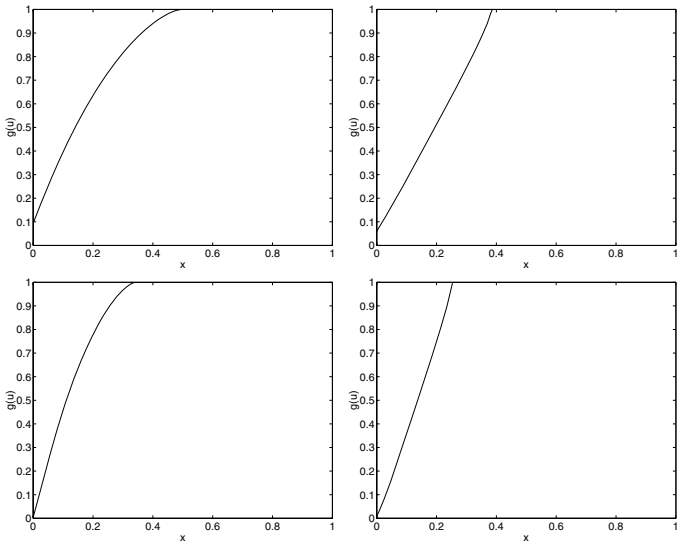


FIG. 4. *The diffusion coefficient g, computed for the numerical solutions at gridpoints, is shown for $\alpha = 0.5$, $p = 2.5$ (top left); $\alpha = 0.5$, $p = 1.8$ (top right); $\alpha = 1.0$, $p = 2.5$ (bottom left); $\alpha = 1.0$, $p = 1.8$ (bottom right).*

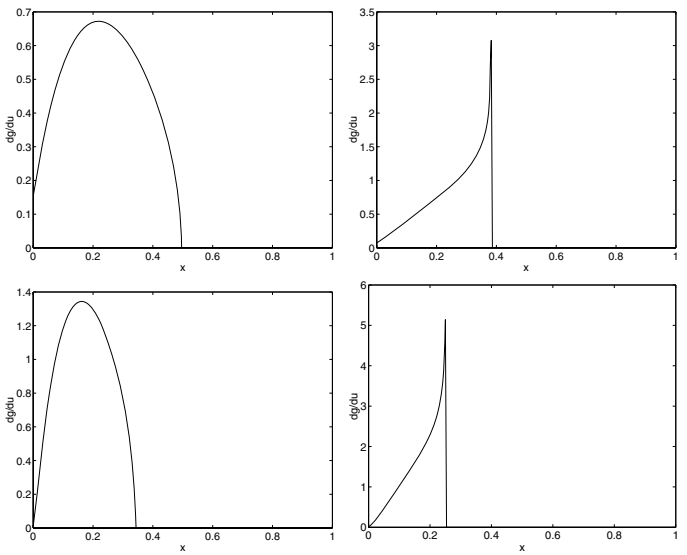


FIG. 5. *The derivative $g'(u)$, computed for the numerical solutions at gridpoints, is shown for $\alpha = 0.5$, $p = 2.5$ (top left); $\alpha = 0.5$, $p = 1.8$ (top right); $\alpha = 1.0$, $p = 2.5$ (bottom left); $\alpha = 1.0$, $p = 1.8$ (bottom right).*

4.2. 2D tests. We test the 2D version of the problem of section 4.1,[2]

$$N(u) \equiv -\nabla \cdot (g(u)\nabla u) = f(x,y), \quad x,y \in (0,1)\times(0,1), \tag{55}$$

again with Dirichlet boundary conditions, where u is the unknown function, g is given by (52), and f is a given function (zero in our tests). Discretizing by second-order

[2]See more 2D tests, including discontinuous coefficients, in [6].

accurate finite volumes on a uniform grid with mesh size h, we obtain the nonlinear system

$$N_h u_h = f_h, \tag{56}$$

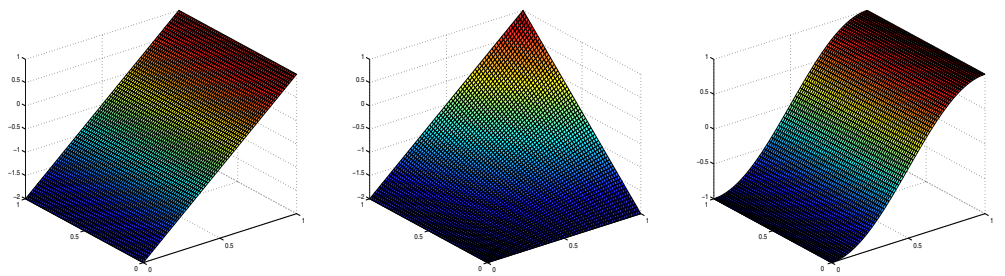
where the ith equation in the system is given by

$$\begin{aligned} &-\frac{g_{i+1/2,j}(u_{i+1,j},u_{i,j})[u_{i+1,j}-u_{i,j}]-g_{i-1/2,j}(u_{i,j},u_{i-1,j})[u_{i,j}-u_{i-1,j}]}{h^2} \\ &-\frac{g_{i,j+1/2}(u_{i,j+1},u_{i,j})[u_{i,j+1}-u_{i,j}]-g_{i,j-1/2}(u_{i,j},u_{i,j-1})[u_{i,j}-u_{i,j-1}]}{h^2} \\ &= f_{i,j} i = 1,\dots,n-1; j = 1,\dots,n-1. \end{aligned} \tag{57}$$

As in the 1D problem, $u_{i,j}$ and $f_{i,j}$ are the components of u_h and f_h, respectively, while $g_{i\pm1/2,j}$ and $g_{i,j\pm1/2}$ are averaged approximations to g midway between neighboring grid points: (i,j) and $(i\pm1,j)$ or (i,j) and $(i,j\pm1)$, respectively. The boundary conditions are described below.

The tests were performed with three cases of boundary conditions. The initial guesses for the three cases are shown in Figure 6. In the first case the boundary conditions were defined to be the boundaries of an inclined linear plane, with its upper side at $u=1$ and its lower side at $u=-2$. In the second case the boundary conditions were set to $u=-2$ along two adjacent sides, while the boundaries along the other two sides were set to linear profiles with $u=-2$ at their lower points and $u=1$ at their common top point. In the third case the boundary values were set to $u=-1$ and $u=1$ along two opposite sides, while along the other sides the values had the shape of a sine function, varying from -1 to 1. The right-hand side function was set to $f=0$ in all the examples.

We perform V cycles until the ℓ_2 residual norm is reduced by a factor of 10^{-8}. We backtrack if the coarse-grid correction plus the (single) postrelaxation sweep fails to reduce the ℓ_2 norm of the residual. In this case, we discard the coarse-grid correction, damp the residual according to the strategy of [10], and compute a new coarse-grid correction.



(a) Case 1: Inclined plane. (b) Case 2. (c) Case 3: Sine function.

FIG. 6. *Boundary conditions and initial guesses.*

TABLE 2

Test results for Case 1. *The* $*$ *superscript denotes that backtracking was required. For this case, setting* $a = 0.3$, $b = 0.5$ *yielded for MNM an average convergence factor of* 0.44 *with no backtracking.*

$P\backslash\alpha$	.5	.75	1
1.5	MNM = 0.18 FAS = 0.26 NLMG = 0.25 FP = 0.50	MNM = 0.26 FAS = 0.35 NLMG = 0.35 FP = 0.64	MNM = 0.97* FAS = 0.64 NLMG = 0.64 FP = 0.77
2	MNM = 0.13 FAS = 0.23 NLMG = 0.20 FP = 0.39	MNM = 0.25 FAS = 0.33 NLMG = 0.31 FP = 0.58	MNM = 0.38 FAS = 0.41 NLMG = 0.40 FP = 0.73
2.5	MNM = 0.12 FAS = 0.21 NLMG = 0.16 FP = 0.35	MNM = 0.25 FAS = 0.33 NLMG = 0.26 FP = 0.58	MNM = 0.41 FAS = 0.42 NLMG = 0.40 FP = 0.76

For error-smoothing we use a nonlinear Gauss–Seidel relaxation in red-black ordering. The local nonlinear problems of the relaxation are solved approximately by one Newton iteration. We use Dendy's classical black-box multigrid [8] to construct prolongation and restriction operators. Alternatives can of course be found in the literature, all essentially approximate generalizations of (54) to two dimensions.

In all our tests the finest grid has 64 intervals in each direction, while the coarsest grid has 4 intervals in each direction, yielding 5 multigrid levels, coarsening down to a grid of 3 by 3 interior variables. On the coarsest grid we relax 5 times. We use $\gamma = 1$ (V cycle) and $\nu_1 = \nu_2 = 1$. For the linearization in the GL and MNM algorithms, we use a fixed-point (FP) approach (sometimes called "lagged diffusion" or "principal-part linearization"), that is, the operator is linearized by freezing the diffusion coefficient, g. This approach was chosen rather than NWT, because the linear operators obtained using NWT are nonsymmetric and therefore require a robust solver for nonsymmetric linear problems. We will consider such solvers in the future.

In these tests we also compute results using the NLMG of Hackbusch [14], with the parameter optimized manually. These must of course be at least as good as those of FAS, since the latter is a special case where the parameter is set to 1. With a few exceptions, the results are not significantly superior to FAS, even though the parameter is optimized.

The results are presented in Tables 2, 3, and 4 for the three cases of boundary conditions. We show average convergence factors for physical parameters $\alpha = 0.5, 1.0, 1.5$, and $p = 1.5, 2.0, 2.5$, with FP denoting fixed-point GL. As in the 1D case, for a given domain and boundary conditions, the problem generally becomes more difficult for smaller p and larger α. The MNM algorithm yielded the fastest convergence in every case but two (the upper-right terms in Tables 2 and 4). In these cases (only), backtracking was required, and the results were still quite poor. (Recall that the analysis for backtracking was performed for Newton's linearization, not FP.) For these two cases where MNM failed, we recomputed the results using the "adaptive" approach

TABLE 3
Test results for Case 2.

$P\backslash\alpha$	.5	.75	1
1.5	MNM = 0.12 FAS = 0.18 NLMG = 0.18 FP = 0.42	MNM = 0.16 FAS = 0.22 NLMG = 0.22 FP = 0.53	MNM = 0.19 FAS = 0.30 NLMG = 0.30 FP = 0.64
2	MNM = 0.11 FAS = 0.18 NLMG = 0.15 FP = 0.34	MNM = 0.16 FAS = 0.26 NLMG = 0.26 FP = 0.51	MNM = 0.27 FAS = 0.35 NLMG = 0.35 FP = 0.66
2.5	MNM = 0.11 FAS = 0.18 NLMG = 0.16 FP = 0.33	MNM = 0.20 FAS = 0.30 NLMG = 0.29 FP = 0.55	MNM = 0.34 FAS = 0.40 NLMG = 0.40 FP = 0.74

TABLE 4
Test results for Case 3. *The* $*$ *superscript denotes that backtracking was required. For this case, setting* $a = 0.2$, $b = 0.4$ *yielded for MNM an average convergence factor of* 0.37 *with no backtracking.*

$P\backslash\alpha$	.5	.75	1
1.5	MNM = 0.15 FAS = 0.21 NLMG = 0.17 FP = 0.39	MNM = 0.24 FAS = 0.31 NLMG = 0.22 FP = 0.50	MNM = 0.94* FAS = 0.94* NLMG = 0.44 FP = 0.58
2	MNM = 0.10 FAS = 0.15 NLMG = 0.13 FP = 0.20	MNM = 0.11 FAS = 0.19 NLMG = 0.15 FP = 0.31	MNM = 0.13 FAS = 0.25 NLMG = 0.22 FP = 0.43
2.5	MNM = 0.10 FAS = 0.14 NLMG = 0.12 FP = 0.13	MNM = 0.10 FAS = 0.16 NLMG = 0.15 FP = 0.24	MNM = 0.11 FAS = 0.21 NLMG = 0.17 FP = 0.38

of section 3.2, with manually chosen parameters: $a = 0.3$, $b = 0.5$ yielded an average convergence factor of 0.44 for Case 1 with $p = 1.5$, $\alpha = 1$, while $a = 0.2$, $b = 0.4$ yielded an average convergence factor of 0.37 for Case 3 with the same values for p and α. This demonstrates the potential efficiency of adaptive MNM.

5. Discussion. MNM eliminates the compromise between global and local linearization, as we approximate the full nonlinear operator on the coarse grid, but we use a robust approximation for a large part of it. The new method exhibits better behavior than both its predecessors, consistent with the prediction of the scalar

analysis. The development of a two- (or higher-) dimensional solver employing Newton linearization is expected to yield better results, provided that a robust linear solver for nonsymmetric linear problems is used. The (partial) [illegible] MNM approach was used in [6], where it proved to be effective, but it can no doubt be improved significantly by deriving appropriate criteria to replace the generic optimization. In the tests above, we demonstrate that optimizing both parameters of the adaptive MNM may be quite efficient, while ensuring that MNM will be at least as good as the better of the two competitors, as they are each a special case of adaptive MNM.

REFERENCES

[1] A. Brandt, *Multi-level adaptive solutions to boundary-value problems*, Math. Comp., 31 (1977), pp. 333–390.

[2] A. Brandt, *Guide to multigrid development*, in Multigrid Methods, W. Hackbusch and U. Trottenberg, eds., Springer-Verlag, Berlin, 1982, pp. 220–312.

[3] A. Brandt, S. F. McCormick, and J. W. Ruge, *Algebraic multigrid (AMG) for sparse matrix equations*, in Sparsity and Its Applications, D. J. Evans, ed., Cambridge University Press, Cambridge, UK, 1984, pp. 257–284.

[4] A. Brandt and I. Yavneh, *On multigrid solution of high-Reynolds incompressible entering flows*, J. Comput. Phys., 101 (1992), pp. 151–164.

[5] W. L. Briggs, V. E. Henson, and S. F. McCormick, *A Multigrid Tutorial*, 2nd ed., SIAM, Philadelphia, 2000.

[6] G. Dardyk and I. Yavneh, *A multigrid approach to two-dimensional phase unwrapping*, Numer. Linear Algebra Appl., 11 (2004), pp. 241–259.

[7] P. M. de Zeeuw, *Nonlinear multigrid applied to a one-dimensional stationary semiconductor model*, SIAM J. Sci. Statist. Comput., 13 (1992), pp. 512–530.

[8] J. E. Dendy, Jr., *Black box multigrid*, J. Comput. Phys., 48 (1982), pp. 366–386.

[9] J. E. Dendy, Jr., *Black box multigrid for nonsymmetric problems*, Appl. Math. Comput., 13 (1983), pp. 261–284.

[10] J. E. Dennis, Jr., and R. B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice Hall Series in Computational Mathematics, Prentice–Hall, Englewood Cliffs, NJ, 1983.

[11] D. Drikakis, O. P. Iliev, and D. P. Vassileva, *A nonlinear multigrid method for the three-dimensional incompressible Navier-Stokes equations*, J. Comput. Phys., 146 (1998), pp. 301–321.

[12] B. Epstein, A. Averbuch, and I. Yavneh, *An accurate ENO driven multigrid method applied to 3D turbulent transonic flows*, J. Comput. Phys., 168 (2001), pp. 316–338.

[13] P. Gerlinger, H. Möbus, and D. Brüggeman, *An implicit multigrid method for turbulent combustion*, J. Comput. Phys., 167 (2001), pp. 247–276.

[14] W. Hackbusch, *Multi-grid Methods and Applications*, Springer-Verlag, Berlin, 1985.

[15] W. Hackbusch and U. Trottenberg, eds., *Multigrid Methods*, Springer-Verlag, Berlin, 1982.

[16] M. J. Holst, R. E. Kozack, F. Saied, and S. Subramaniam, *Treatment of electrostatic effects in proteins – multigrid-based Newton iterative method for solution of the full nonlinear Poisson-Boltzmann equation*, Proteins: Structure Function and Genetics, 18 (1994), pp. 231–245.

[17] M. J. Holst and F. Saied, *Numerical solution of the nonlinear Poisson-Boltzmann equation – developing more robust and efficient methods*, J. Comput. Chem., 16 (1995), pp. 337–364.

[18] J. Korsawe and G. Starke, *Multilevel projection methods for nonlinear least-squares finite element computations*, Electron. Trans. Numer. Anal., 21 (2000), pp. 1869–1885.

[19] M. H. Lallemand and B. Koren, *Iterative defect correction and multigrid accelerated explicit time stepping schemes for the steady Euler equations*, SIAM J. Sci. Comput., 14 (1993), pp. 953–970.

[20] J. Molenaar, *Adaptive multigrid applied to a bipolar-transistor problem*, Appl. Numer. Math., 17 (1995), pp. 61–83.

[21] J. W. Ruge, Y. Li, S. McCormick, A. Brandt, and J. R. Bates, *A nonlinear multigrid solver for a semi-Lagrangian potential vorticity-based shallow-water model on the sphere*, SIAM J. Sci. Comput., 21 (2000), pp. 2381–2395.

[22] J. W. Ruge and K. Stüben, *Algebraic multigrid (AMG)*, in Multigrid Methods, Frontiers Appl. Math. 3, S. F. McCormick, ed., SIAM, Philadelphia, 1987, pp. 73–130.

[23] G. Starke, *Gauss-Newton multilevel methods for least-squares finite element computations of variably saturated subsurface flow*, Computing, 64 (2000), pp. 323–338.
[24] G. Starke, *Least-squares mixed finite element solution of variably saturated subsurface flow problems*, SIAM J. Sci. Comput., 21 (2000), pp. 1869–1885.
[25] U. Trottenberg, C. Oosterlee, and A. Schüller, *Multigrid*, Academic Press, London, San Diego, 2001.
[26] M. Th. Van Genuchten, *A closed-form equation for predicting the hydraulic conductivity of unsaturated soils*, Soil Sci. Soc. Am. J., 44 (1980), pp. 892–898.
[27] C. R. Vogel and M. E. Oman, *Iterative methods for total variation denoising*, SIAM J. Sci. Comput., 17 (1996), pp. 227–238.
[28] S. Zeng and P. Wesseling, *Multigrid solution of the incompressible Navier–Stokes equations in general coordinates*, SIAM J. Numer. Anal., 31 (1994), pp. 1764–1784.

SIAM J. SCI. COMPUT.
Vol. 28, No. 1, pp. 47–74
© 2006 Society for Industrial and Applied Mathematics

INCORPORATING DIFFUSION IN COMPLEX GEOMETRIES INTO STOCHASTIC CHEMICAL KINETICS SIMULATIONS*

SAMUEL A. ISAACSON† AND CHARLES S. PESKIN‡

Abstract. A method is developed for incorporating diffusion of chemicals in complex geometries into stochastic chemical kinetics simulations. Systems are modeled using the reaction-diffusion master equation, with jump rates for diffusive motion between mesh cells calculated from the discretization weights of an embedded boundary method. Since diffusive jumps between cells are treated as first order reactions, individual realizations of the stochastic process can be created by the Gillespie method. Numerical convergence results for the underlying embedded boundary method, and for the stochastic reaction-diffusion method, are presented in two dimensions. A two-dimensional model of transcription, translation, and nuclear membrane transport in eukaryotic cells is presented to demonstrate the feasibility of the method in studying cell-wide biological processes.

Key words. reaction-diffusion, stochastic chemical kinetics, complex geometry, embedded boundary

AMS subject classifications. 65C20, 92C40, 65M06

DOI. 10.1137/040605060

1. Introduction.

Spatially homogeneous, deterministic mass-action kinetics is a standard model for the interactions of proteins, genes, and mRNAs within cellular networks. It assumes that the evolution of any chemical species can be represented by a continuously varying concentration representing the number of that species within the cell divided by the cell volume. Fundamentally, however, the numbers of protein and mRNA molecules within cells are discrete integer variables. The notion of a continuously varying concentration is only well defined for a sufficiently large number of the molecules. Moreover, for systems in which spatial effects are important the population number must not only be large within the cell but also locally within areas of interest. If the population number is sufficiently small, stochastic effects can have a noticeable impact on the behavior of the biological system.

The issue of stochasticity in gene expression was discussed in [3]. The authors subsequently demonstrated numerically that the variation of cell fates within a λ-phage infected population of bacterial cells can be explained as arising from a stochastically driven gene expression switch [4]. The presence of noise in prokaryotic transcription and translation was demonstrated experimentally in both [9] and [23]. More recently, experimental results demonstrating transcriptional noise in eukaryotic cells were presented in [6].

Spatial effects are present in many biological systems, and hence the spatially uniform, or well-mixed, assumption will not always hold. Systems in which spatial effects are known to play a role include Ash1 mRNA localization in budding yeast [1],

*Received by the editors March 11, 2004; accepted for publication (in revised form) August 11, 2005; published electronically March 3, 2006. This work was partially supported by the National Computational Science Alliance under DMS030001N and utilized the NCSA IA–32 Linux Cluster. Partial support was also provided by Defense Advanced Research Projects Agency–Air Force Research Laboratory grant F30602-02-0554.

http://www.siam.org/journals/sisc/28-1/60506.html

†Department of Mathematics, University of Utah, 155 S. 1400 E., Room 233, Salt Lake City, UT 84112 (isaacsas@cims.nyu.edu).

‡Courant Institute of Mathematical Sciences, 251 Mercer St., New York, NY 10012 (peskin@cims.nyu.edu).

morphogen gradients across egg-polarity genes in Drosophila oocyte [1], and the synapse-specificity of long-term facilitation in Aplysia [20]. The exact processes by which proteins and mRNAs move between specific locations, and become localized, are not yet known definitively. Movement is believed to be composed of a mixture of diffusion and various forms of active transport. For example, in [10], mRNA movement within the cytoplasm was found to exhibit diffusive motion, "corralled" diffusive motion, and active, directionally specific, transport. A probabilistic switching between these three types of movement was observed in most cases. Moreover, the fraction of time spent in each of the types of motion was found to differ depending on whether the mRNA was destined to be localized or not.

Diffusion can be incorporated into deterministic mass-action kinetics through the use of reaction-diffusion partial differential equations. Molecular noise can be accounted for through the use of stochastic chemical kinetics, but the usual formulation of stochastic chemical kinetics assumes a spatially homogeneous cellular space. Methods for incorporating diffusion into stochastic chemical kinetics are presented in [25] and [12]. In both methods, space is divided into a collection of equally sized mesh cells. A reaction-diffusion master equation is then derived to give the probability of the system being in a given state. In [12] all transition rates between mesh cells are identical, and formulas for the transition rates are calculated in this case. In [25] a discussion of the continuum-limit for the first several moments of the model is given, but no method is given for calculating the cell-based transition rates.

We present a stochastic reaction-diffusion model for mesh cells of arbitrary volume and varying diffusive transition rates. In section 3, a method is developed for calculating diffusive transition rates between mesh cells within a domain containing multiple complex boundaries. The development of a convergent diffusive approximation within complex geometries is critical for use in individual (biological) cell models where the large number of organelles and membranes could have a significant effect on the overall dynamics of chemical systems. Transition rates are systematically calculated from terms in an embedded boundary discretization of the Fokker–Planck equation for a classically diffusing Brownian particle. We expect the reaction-diffusion master equation to be such a discretization in the case of no reactions and only one particle. The notion of the master equation as an approximation to a continuum equation was used in [26] to obtain approximations of the Fokker–Planck equation for a single particle moving in a one-dimensional potential. The master equation itself has a special structure, requiring a constraint on the weights of the discretization. Our method is found to conserve probability and satisfy detailed balance. The underlying embedded boundary discretization converges between first and second order in practice. As an application of the method, a model of transcription, translation, and nuclear transport within a two-dimensional eukaryotic cell is developed and simulated.

We note that the method we develop is an Eulerian method, using an underlying Cartesian mesh. Other models have been developed using Lagrangian methods that track the movement of each individual molecule within the system. For example, in [2] a Lagrangian method was developed for simulating stochastic reaction-diffusion systems within biological cells. These methods have the benefit of allowing exact local treatments of particle diffusion; however, they introduce other challenges in how to handle reactions between particles, detection of particle collisions, and resolving particle-geometry interactions. Lagrangian methods have been found very useful in the turbulent reactive flow literature. We point the reader interested in numerical methods for convection dominated reactive flows to [22] and [19].

2. Master equation formulation of reaction-diffusion. In section 2.1 the general abstract definition of the master equation is presented. Several properties of the transition rates, $W_{ij}(t)$, are pointed out. These properties constrain the form of the discretization of the Fokker–Planck equation used in generating the diffusive transition rates for the reaction-diffusion master equation. Sections 2.2 and 2.3 review the chemical master equation for a spatially homogeneous system, and the Gillespie method for creating realizations of it. In section 2.4 the reaction-diffusion master equation is presented, along with the motivation for thinking of its diffusive jump rates as weights in a spatial-only discretization of the Fokker–Planck equation.

2.1. General master equation. Let $X(t)$ denote an integer valued, continuous time Markov process. Let $Q_i(t) = \text{Prob}\{X(t) = i | X(0) = x_0\}$, and $W_{ij}(t)dt = \text{Prob}\{X(t + dt) = i | X(t) = j\}$. By definition, $W_{ij}(t)$ is the transition rate at time t to go into state i from state j, for $i \neq j$. Then

$$Q_i(t + dt) = \left(1 - \sum_{j \neq i} W_{ji}(t)dt\right) Q_i(t) + \sum_{j \neq i} W_{ij}(t)Q_j(t)dt + o(dt). \tag{2.1}$$

This says that the probability of being in state i a small time after time t is the probability of being there at time t and staying there added to the probability of being in another state at time t and transitioning to i. The master equation is the coupled set of ODEs obtained by rearranging and taking the limit $dt \to 0$ of (2.1):

$$\begin{aligned} \frac{dQ_i}{dt} &= \sum_{j:j\neq i} \left(W_{ij}(t)Q_j(t) - W_{ji}(t)Q_i(t)\right) \\ &= \left(\sum_{j:j\neq i} W_{ij}(t)Q_j(t)\right) - W_i(t)Q_i(t), \quad W_i(t) = \sum_{j\neq i} W_{ji}(t) \geqslant 0. \end{aligned} \tag{2.2}$$

There are several properties of the structure of (2.2) that we subsequently use. First, $W_{ij}(t) \geq 0$. Second, the term $W_{ij}(t)Q_j(t)$ appears twice with opposite sign: once in the equation for $\partial_t Q_i$ and once in the equation for $\partial_t Q_j$. This implies that the amount of probability that flows per unit time into state i due to transitions from state j to i is exactly the amount of probability that flows per unit time out of state j due to these same transitions. Finally, the only negative term in the equation for $Q_i(t)$ is given by $W_i(t)Q_i(t)$.

2.2. Review of stochastic chemical kinetics. Consider a well-mixed chemical system in a finite closed volume V; that is, the probability of a given particle of the system being in an arbitrary subregion of volume dV is dV/V. Further, let each individual particle's velocity be a Maxwell–Boltzmann distributed random variable. With these assumptions Gillespie showed in [16] that a chemical system can be represented as an integer valued, continuous time Markov process satisfying a master equation.

Let $\boldsymbol{M}(t) = (M^1(t), \ldots, M^L(t))$ denote the state vector of the chemical system. $M^l(t)$ will be the random variable representing the number of molecules of chemical species l at time t. Define $\boldsymbol{m} = (m^1, \ldots, m^L)$ to be a given value of $\boldsymbol{M}(t)$. Denote by S^l the [illegible] of the lth species. Assume there are K possible reactions, with the function $a^k(\boldsymbol{m})$ giving the probability per unit time of reaction k occurring when $\boldsymbol{M}(t) = \boldsymbol{m}$. For example, letting k label the unimolecular (first order) reaction

$S^i \to S^j$, then $a^k(\boldsymbol{m}) = \alpha\, m^i$, where α is the rate constant in units of number of occurrences of the reaction per molecule of S^i per unit time. Letting k' denote the index of the bimolecular reaction $S^i + S^j \to S^n$, where $i \neq j$, then $a^{k'}(\boldsymbol{m}) = \beta\, m^i m^j$. Here β is the rate constant in units of number of occurrences of the reaction per molecule of S^i and per molecule of S^j, per unit time. Note that if species i and j are the same, then $a^{k'}(\boldsymbol{m}) = (\beta/2)\, m^i(m^i - 1)$. Let $\boldsymbol{\nu}_k = (\nu_k^1, \ldots, \nu_k^L)$ be the change in $\boldsymbol{M}$ that results from one occurrence of reaction k (i.e., $\boldsymbol{M}(t) \to \boldsymbol{M}(t) + \boldsymbol{\nu}_k$). The integers $(\nu_k^1, \ldots, \nu_k^L)$ define the stoichiometry of the kth reaction. In the notation of section 2.1, $a^k(\boldsymbol{m}) = W_{\boldsymbol{m}+\boldsymbol{\nu}_k, \boldsymbol{m}}$, assuming $\boldsymbol{\nu}_i \neq \boldsymbol{\nu}_k$ for all $i \neq k$. (If two or more reactions have identical stoichiometry, their rates are added to obtain the corresponding transition rate W. Also, if there are no reactions leading from state $\boldsymbol{m}'$ to $\boldsymbol{m}$, then $W_{\boldsymbol{m},\boldsymbol{m}'}$ is zero.) Note that the ν_k^l may be positive or negative (or zero). Then if

$$P(\boldsymbol{m}, t) \equiv P(\boldsymbol{m}) \equiv \text{Prob}\{\boldsymbol{M}(t) = \boldsymbol{m} | \boldsymbol{M}(0) = \boldsymbol{m}_0\},$$

the master equation for the time evolution of the probability of the system will be

$$\frac{dP(\boldsymbol{m})}{dt} = \sum_{k=1}^{K} \left(a^k(\boldsymbol{m} - \boldsymbol{\nu}_k) P(\boldsymbol{m} - \boldsymbol{\nu}_k) - a^k(\boldsymbol{m}) P(\boldsymbol{m}) \right). \tag{2.3}$$

This is a coupled set of ODEs over all possible integer values of the components of the vector $\boldsymbol{m}$. We shall subsequently refer to (2.3) as the chemical master equation. Assuming all the $\boldsymbol{\nu}_k$ are distinct, in the notation of section 2.1 the master equation is

$$\frac{dP(\boldsymbol{m})}{dt} = \sum_{k=1}^{K} \left(W_{\boldsymbol{m}, \boldsymbol{m}-\boldsymbol{\nu}_k} P(\boldsymbol{m} - \boldsymbol{\nu}_k) - W_{\boldsymbol{m}+\boldsymbol{\nu}_k, \boldsymbol{m}} P(\boldsymbol{m}) \right). \tag{2.4}$$

In both (2.3) and (2.4) the first term in the sum corresponds to all possible transitions into state $\boldsymbol{m}$ from some prestate, $\boldsymbol{m} - \boldsymbol{\nu}_k$, while the second term corresponds to all possible transitions out of state $\boldsymbol{m}$ to $\boldsymbol{m} + \boldsymbol{\nu}_k$.

We now consider the relationship between the traditional macroscopic chemical kinetic equations to the master equation derived above. Define $\boldsymbol{C} \equiv \boldsymbol{M}/V$ to be the vector of random variables for the concentration of each species. Let $\tilde{a}^k$ denote the concentration dependent form of the rate a^k. $\tilde{a}^k$ and a^k are related by $\tilde{a}^k(\boldsymbol{c}) = a^k(V\boldsymbol{c})/V$, and vice versa $a^k(\boldsymbol{m}) = \tilde{a}^k(\boldsymbol{m}/V)V$. Note that $\tilde{a}^k(\boldsymbol{c})$ is the expected number of occurrences of reaction k [illegible] per unit time when the vector of concentrations is $\boldsymbol{c}$. From the master equation the mean of C^l, $\langle C^l \rangle$, satisfies

$$\frac{d\langle C^l \rangle}{dt} = \sum_{k=1}^{K} \nu_k^l \langle \tilde{a}^k(\boldsymbol{C}) \rangle.$$

If the averaging operation commutes with the functions $\tilde{a}^k$, then one obtains the equations for classical deterministic chemical kinetics,

$$\frac{d\langle C^l \rangle}{dt} = \sum_{k=1}^{K} \nu_k^l\, \tilde{a}^k(\langle \boldsymbol{C} \rangle);$$

however, in general $\langle \tilde{a}^k(\boldsymbol{C}) \rangle \neq \tilde{a}^k(\langle \boldsymbol{C} \rangle)$, unless $\tilde{a}^k$ is linear. One does expect that asymptotically, as the population sizes become very large, $\langle \tilde{a}^k(\boldsymbol{C}) \rangle \approx \tilde{a}^k(\langle \boldsymbol{C} \rangle)$. For a discussion of the validity of the deterministic approximation see [17].

2.3. Gillespie method. The number of the ODEs comprising the master equation are very large even for simple chemical systems. Direct solution methods become impractical, and thus Monte Carlo methods are used instead. These methods create realizations of the underlying stochastic process, $\boldsymbol{M}(t)$, governed by the master equation. Statistics from many realizations can be used to calculate moments of $\boldsymbol{M}(t)$ and the distribution, $P(\boldsymbol{m}, t)$. In [15] Gillespie presented two equivalent methods that produce exact realizations of $\boldsymbol{M}(t)$.

One of the two methods, the first reaction method, is based on calculating the time of occurrence of each reaction independently, as if no other reactions are present in the system. The reaction with the minimal time to its next occurrence is then executed, and the system state updated. This process is then repeated until the simulation reaches the desired final time. In detail, let

$$p\left(\tau, k|\boldsymbol{m}, t\right) d\tau \equiv \text{ Prob}\{\text{reaction } k \text{ occurs in the interval } [t+\tau, t+\tau+d\tau), \\ \text{ignoring that another reaction could occur first}\}.$$

Then in [15] it is shown that

$$p\left(\tau, k|\boldsymbol{m}, t\right) = a^k\left(\boldsymbol{m}\right) e^{-a^k(\boldsymbol{m})\tau}. \tag{2.5}$$

Knowing $\boldsymbol{m}$ at time t, for each reaction k, a time τ_k at which reaction k would occur can be generated by inverting the probability distribution corresponding to probability density (2.5). The time at which reaction k would occur is given by $t + \tau_k$, where

$$\tau_k = \frac{1}{a^k(\boldsymbol{m})} \ln \frac{1}{r_k},$$

r_k being a uniformly distributed random number in $[0, 1]$. The overall time-evolution algorithm is then the following:

1. Initialize $t = 0$, set initial molecule numbers $\boldsymbol{m} = \boldsymbol{m}_0$
2. Calculate $a^k(\boldsymbol{m})$ for all k
3. For each k, generate $\tau_k = \left(\ln\left(1/r_k\right)\right)/a^k(\boldsymbol{m})$
4. Execute the reaction, k', with the smallest τ_k, update $\boldsymbol{m} := \boldsymbol{m} + \boldsymbol{\nu}_{k'}$
5. Set the time to $t := t + \tau_{k'}$
6. Return to 2.

Optimized versions of Gillespie's algorithms that substantially decrease the computational work were developed by Gibson and Bruck [14]. Their next reaction method is equivalent to the first reaction method but requires only one random number per reaction event. By using a special data structure they reduce finding the reaction with the minimum τ_k to a constant work operation, and updating following a reaction to logarithmic work in the number of reactions, K. For all subsequent Monte Carlo simulations discussed we use the next reaction method.

2.4. Reaction-diffusion master equation. Let the domain of interest be a closed volume V. Divide the domain into a series of computational cells indexed by $i = 1, \ldots, N$. Assume that the size of each cell can be chosen such that within each cell, independently, the master equation formulation of chemical kinetics holds (see section 2.2). Transitions of particles [illegible] cells are then modeled as first order reactions.

Let $M_i^l(t)$ denote the random variable for the number of particles of species l in cell i, $l = 1, \ldots, L$. Define $\boldsymbol{M}^l(t) = (M_1^l, \ldots, M_N^l)^T$ to be the spatial vector of

species l, and $\boldsymbol{M}(t) = (\boldsymbol{M}^1, \ldots, \boldsymbol{M}^L)$ to be the state matrix of the system. As for the well-mixed case, let

$$P(\boldsymbol{m}, t) \equiv P(\boldsymbol{m}) \equiv \text{Prob}\{\boldsymbol{M}(t) = \boldsymbol{m} | \boldsymbol{M}(0) = \boldsymbol{m}_0\}.$$

Let $\boldsymbol{M}_i(t) = (M_i^1, \ldots, M_i^L)$, $\boldsymbol{e}_i^l$ be the matrix all of whose elements are zero except for the element (i, l), which is one, and let $\boldsymbol{e}_i$ be the column vector which has a 1 in its ith entry and is 0 elsewhere. $\boldsymbol{e}_i \boldsymbol{\nu}_k$ will denote the matrix formed by the product of the column vector $\boldsymbol{e}_i$ with the row vector $\boldsymbol{\nu}_k$. This matrix will be zero everywhere except for the ith row, which will be equal to $\boldsymbol{\nu}_k$. It corresponds to the change in $\boldsymbol{M}(t)$ due to one occurrence of reaction k at location i (i.e., $\boldsymbol{M}(t) \to \boldsymbol{M}(t) + \boldsymbol{e}_i \boldsymbol{\nu}_k$). Let k_{ij}^l denote the jump rate for each individual molecule of the lth chemical species into cell i from cell j, for $i \neq j$. Since diffusion is treated as a first order reaction, and since the molecules are assumed to diffuse independently, the total probability per unit time at time t for one molecule of species l to jump from cell j to cell i is $k_{ij}^l M_j^l(t)$.

The master equation for the time evolution of $P(\boldsymbol{m}, t)$ is then

$$\begin{aligned} \frac{dP(\boldsymbol{m})}{dt} =& \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{l=1}^{L} \left(k_{ij}^l \left(m_j^l + 1 \right) P(\boldsymbol{m} + \boldsymbol{e}_j^l - \boldsymbol{e}_i^l) - k_{ji}^l m_i^l P(\boldsymbol{m}) \right) \\ &+ \sum_{i=1}^{N} \sum_{k=1}^{K} \left(a_i^k(\boldsymbol{m}_i - \boldsymbol{\nu}_k) P(\boldsymbol{m} - \boldsymbol{e}_i \boldsymbol{\nu}_k) - a_i^k(\boldsymbol{m}_i) P(\boldsymbol{m}) \right). \end{aligned} \tag{2.6}$$

This is a coupled set of ODEs over all possible integer values of the matrix $\boldsymbol{m}$. Notice the important point that the reaction probabilities per unit time, $a_i^k(\boldsymbol{m})$, may now depend on spatial location. For example, in an eukaryotic cell some reactions may occur only in the nucleus and others may occur only in the cytoplasm.

Equation (2.6) is separated into two sums. The first term corresponds to diffusive motion between cells i and j of a given species, l. The second is just the components of the chemical master equation, but applied at each individual cell. At this point no specification has been made as to where the rates k_{ij}^l come from, or what their values should be.

For deterministic reaction-diffusion, let $C^l(\boldsymbol{x}, t)$ denote the concentration and D^l the diffusion constant of the lth species. Let $\tilde{a}^k(\boldsymbol{C}(\boldsymbol{x}, t), \boldsymbol{x})$ denote the spatially varying concentration dependent form of a_i^k (see section 2.2). Then

$$\frac{\partial C^l(\boldsymbol{x}, t)}{\partial t} = D^l \Delta C^l + \sum_{k=1}^{K} \nu_k^l \, \tilde{a}^k(\boldsymbol{C}(\boldsymbol{x}, t), \boldsymbol{x}).$$

To see the relation to the stochastic formulation let $C_i^l(t) = M_i^l(t)/V_i$ be the random variable for the chemical concentration of species l, in cell i. Let $\tilde{a}_i^k$ denote the concentration dependent form of a_i^k. $\tilde{a}_i^k$ and a_i^k are related by $\tilde{a}_i^k(\boldsymbol{c}) = a_i^k(V_i \boldsymbol{c})/V_i$, and vice versa $a_i^k(\boldsymbol{m}) = \tilde{a}_i^k(\boldsymbol{m}/V_i) V_i$. From the reaction-diffusion master equation the average of C_i^l then satisfies

$$\frac{d\langle C_i^l \rangle}{dt} = \sum_{j=1}^{N} \left(\frac{V_j}{V_i} k_{ij}^l \langle C_j^l \rangle - k_{ji}^l \langle C_i^l \rangle \right) + \sum_{k=1}^{K} \nu_k^l \langle \tilde{a}_i^k \left(C_i \right) \rangle.$$

Suppose the rates k_{ij}^l could be chosen so that the first term is a discretization of $D^l \Delta C^l$. The only difference between the deterministic and the means of the stochastic formulation in the continuum limit would then be the noncommutativity of the reaction terms with averaging; diffusion would be identical.

Returning to the reaction-diffusion master equation (2.6), we expect that the diffusive jump rates should be independent of the chemical reactions present within a given chemical system. As such, we now consider a chemical system consisting of no reactions. In this case, we expect each particle of each chemical species to independently undergo Brownian motion in the continuum limit as the computational cell size approaches zero. Since, by construction, the diffusive motion of each particle of each chemical species is independent in the reaction-diffusion master equation formulation, it is sufficient to further restrict ourselves to a system consisting of only one chemical species containing just one particle. With these choices, the reaction-diffusion master equation reduces to

$$\frac{dP(\boldsymbol{m})}{dt} = \sum_{i=1}^{N} \sum_{j=1}^{N} k_{ij} \, (m_j + 1) \, P(\boldsymbol{m} + \boldsymbol{e}_j - \boldsymbol{e}_i) - k_{ji} m_i P(\boldsymbol{m}). \tag{2.7}$$

Since $\boldsymbol{M}(t)$ is describing one particle, $M_i(t)$ will be zero for all i except the location the particle is currently at. Let $\boldsymbol{X}(t)$ be the Brownian motion process of the particle. Then define

$$Q_i(t) \equiv P(\boldsymbol{e}_i, t) \approx \text{Prob}\{\boldsymbol{X}(t) \in V_i | \boldsymbol{X}(0) = \boldsymbol{x}_0\},$$

where $\boldsymbol{e}_i$ denotes, as before, the unit vector along the ith Cartesian coordinate axis of $\mathbb{R}^N$. With this definition, $Q_i(t)$ is the probability that the particle is in the ith computational cell. Letting $\boldsymbol{m} = \boldsymbol{e}_i$, equation (2.7) implies

$$\frac{dQ_i}{dt} = \sum_{j=1}^{N} k_{ij} \, Q_j - k_{ji} \, Q_i. \tag{2.8}$$

Define a probability density for the particle's position by $p_i(t) \equiv Q_i(t)/V_i$. Then p_i satisfies

$$\frac{dp_i}{dt} = \sum_{j=1}^{N} \frac{V_j}{V_i} k_{ij} \, p_j - k_{ji} \, p_i. \tag{2.9}$$

Again, if the rates form a discretization of $D\Delta p$, for p the continuum single particle density, then the continuum limit as cell size goes to zero would be

$$\frac{\partial p(\boldsymbol{x}, t)}{\partial t} = D \Delta p. \tag{2.10}$$

This is the Fokker–Planck equation for a single classical Brownian particle.

With these continuum limits in mind, constructing an operator, L_h, of the form

$$(L_h p)_i = \sum_{j=1}^{N} \frac{V_j}{V_i} k_{ij} \, p_j - k_{ji} \, p_i,$$

where $L_h \to \Delta$ as $h \to 0$, will determine the k_{ij}. (Here h represents the maximum length scale associated with a computational cell.) The diffusion jump rates will then be k_{ij}, provided $k_{ij} \geq 0$ (see section 2.1).

3. Determining the jump rates. From the results of section 2.4, an appropriate spatial discretization of either the classical deterministic formulation of reaction-diffusion, or the Fokker–Planck equation for a single Brownian particle, would determine the jump rates in (2.6). We note that in choosing the jump rates to recover either one of these two formulations, the macroscopic diffusion of the mean concentration from the reaction-diffusion master equation, or the microscopic Brownian motion of independent particles, we will also recover the other formulation. Previous authors [12], [25] have used the requirement of obtaining macroscopic diffusion of the mean concentrations given by the reaction-diffusion master equation to derive diffusive jump rates in the special case of a uniform Cartesian mesh. In this case, it has been shown that the jump rates, with mesh width h, can be given by

$$\frac{D}{h^2}$$

for jumps between neighboring mesh cells, and are zero for jumps between non-neighboring cells. This result has been known from as far back as [11]. In this section, instead of focusing on recovering macroscopic diffusion of the mean concentrations, we focus on the mathematically equivalent requirement of reproducing single particle Brownian motion. We note that if we assume molecules are only moving by diffusion, then the requirement of recovering single particle Brownian motion in the continuum limit is a physical necessity for the reaction-diffusion master equation. With this choice we can derive valid jump rates for diffusion within complex geometries, allowing us to extend the existing methodology to handle arbitrary boundaries. This result is important since many problems of practical interest contain geometrically complex domains, such as the movement of molecules within biological cells where there are many barriers due to organelles and membranes.

To discretize the Laplacian, and hence obtain the diffusive jump rates, boundary conditions must be specified for the domain of interest; this is discussed in section 3.1. Section 3.2 derives the embedded boundary method to discretize the Fokker–Planck equation, and calculates the corresponding diffusive jump rates for the reaction-diffusion master equation. Some consequences of the specific discretization chosen are also presented. In section 3.3 the complete algorithm for solving the reaction-diffusion master equation is summarized. Convergence of the embedded boundary discretization is demonstrated for several examples in section 3.4, while section 3.5 discusses the convergence of the overall algorithm of section 3.3.

3.1. Boundary conditions. Since we are applying the overall method to simulations of biochemical networks in eukaryotic cells, which are cells with a nuclear membrane, we formulate our boundary conditions for this type of domain. Denote the entirety of the cell by Ω and the cell membrane (i.e., exterior boundary) as $\partial\Omega$. A eukaryotic cell will also have a nucleus, $\Omega_n \subset \Omega$, a closed volume contained completely within the interior of the cell. The nuclear membrane, $\partial\Omega_n$, encloses the nucleus, partitioning the cell's interior into two distinct compartments.

Passage through the cellular membrane will not be allowed, so that the exterior boundary will have a no-flux Neumann condition. This can, of course, be changed to a more general flux condition if necessary. The nuclear membrane will be impermeable to certain species, requiring again a no-flux Neumann condition, while allowing a passive flux for others. This second condition will be modeled in this paper by having the nuclear membrane flux proportional to the jump in probability density, or concentration, across the membrane. Note that this boundary condition assumes molecules

cross the nuclear membrane independently of each other. The boundary condition may be derived from a more detailed model of the nuclear membrane as containing channels, called nuclear pores, through which certain species may diffuse. The nuclear membrane is thin relative to the size of a eukaryotic cell, and hence the diffusive motion of molecules within the membrane will equilibrate rapidly. This implies that the diffusive flux within the membrane can be assumed spatially constant. In addition, the number of nuclear pores is large enough that we approximate them as a fixed background density within the membrane. Combining these two approximations, and idealizing the membrane as infinitely thin, implies that the flux through the nuclear membrane is proportional to the difference in probability density, or concentration, across the membrane. Note that in certain situations this boundary condition would need to be modified to a more physically detailed model. One such example would be if the number of molecules crossing the nuclear membrane can become large enough to saturate the nuclear pores.

Consider a single particle moving by classical Brownian motion within the cell. Let $p(\boldsymbol{x},t)$ denote the probability density for the position of the particle. Then within both the nucleus, Ω_n, and the cytoplasm, $\Omega \setminus \Omega_n$, $p(x,t)$ will satisfy (2.10). (Here $\Omega \setminus \Omega_n$ denotes the set of points within Ω that are not also contained in Ω_n.) Let $[p]_n$ denote the jump in $p(\boldsymbol{x},t)$ across the nuclear membrane. Define by ρ the nuclear membrane permeability, and by D the diffusion constant of the particle. Let $\boldsymbol{\eta}$ denote the outward pointing normal to a given surface. Then, incorporating the assumed boundary conditions, $p(\boldsymbol{x},t)$ satisfies

$$\begin{aligned} \frac{\partial p}{\partial t} &= D\Delta p \qquad && \text{in } \Omega, \\ \frac{\partial p}{\partial \boldsymbol{\eta}} &= 0 && \text{on } \partial\Omega, \\ -D\frac{\partial p}{\partial \boldsymbol{\eta}} &= -\rho\,[p]_n && \text{on } \partial\Omega_n, \end{aligned} \tag{3.1}$$

with the initial condition $p(\boldsymbol{x},0) = \delta(\boldsymbol{x}-\boldsymbol{x}_0)$, where $\boldsymbol{x}_0$ is the known initial position of the particle. A discretization of these equations that has the general form of (2.9) will determine the jump rates in the reaction-diffusion master equation.

The corresponding multiple species, deterministic reaction-diffusion model for the assumed boundary conditions follows. Let $[C^l]_n$ denote the jump in C^l across the nuclear membrane, ρ^l the nuclear membrane permeability, and D^l the diffusion constant of the lth species. Then

$$\begin{aligned} \frac{\partial C^l(\boldsymbol{x},t)}{\partial t} &= D^l\Delta C^l + \sum_{k=1}^{K} \nu_k^l\, \tilde{a}^k(\boldsymbol{C}(\boldsymbol{x},t),\boldsymbol{x}) && \text{in } \Omega, \\ \frac{\partial C^l(\boldsymbol{x},t)}{\partial \boldsymbol{\eta}} &= 0 && \text{on } \partial\Omega, \\ -D^l\frac{\partial C^l(\boldsymbol{x},t)}{\partial \boldsymbol{\eta}} &= -\rho^l\left[C^l\right]_n && \text{on } \partial\Omega_n, \end{aligned} \tag{3.2}$$

with the initial condition, $C^l(\boldsymbol{x},0) = C_0(\boldsymbol{x})$.

3.2. Numerical discretization. For simplicity, we discretize (3.1) instead of the deterministic reaction-diffusion formulation (3.2). We may do this because our purpose is not to obtain a numerical scheme for the deterministic equations but instead

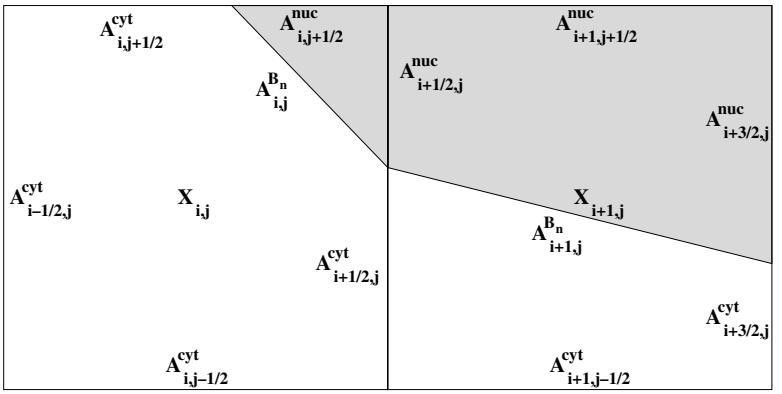


FIG. 1. *Two-dimensional Cartesian mesh cells cut by nuclear membrane. The darker region represents the portion of Ω_n, the nuclear space, within the mesh cells. The lighter region represents the portion of $\Omega \setminus \Omega_n$, the cytoplasm, within the mesh cells. $A^{\mathrm{cyt}}_{i+1/2,j}$ gives the length of the piece of the face connecting cells (i,j) and $(i+1,j)$ that is within the cytoplasmic domain. The remaining $A^{\alpha}_{i,j}$ values, for α a given domain, are defined similarly. $A^{B_n}_{i,j}$ is the length of the piece of the nuclear membrane within the cell (i,j). $X_{i,j}$ gives the center of the Cartesian cell at location (i,j). Also, $V^{\alpha}_{i,j}$ (not shown) gives the area of the portion of cell (i,j) within the domain α.*

to obtain the jump rates for use in our stochastic simulation, and, as was shown in section 2.4, the jump rates are the same whether we use (3.1) or the full deterministic reaction-diffusion system (3.2). For the remainder of this section, we use the word "cell" to refer to a cell of the Cartesian computational mesh, and we specify "biological cell" when that is what we mean. We embed Ω in a Cartesian mesh with cell centers $\boldsymbol{x_i}$ and mesh width h. There are three different domains contained in this description: the nuclear space, Ω_n, the cytoplasm, $\Omega \setminus \Omega_n$, and the space exterior to the biological cell. The total computational domain within which the biological cell is embedded is taken to be a square.

Define three separate solutions to (3.1), corresponding to each of the three domains: p^{nuc}, the value of $p(x,t)$ within Ω_n, p^{cyt}, the value of $p(x,t)$ within $\Omega \setminus \Omega_n$, and p^{ext}, the value of p outside the biological cell. Since the particle is assumed to never leave Ω, p^{ext} will be zero. Denote by p^{α} the value of the solution for domain α, where $\alpha \in \{\mathrm{nuc}, \mathrm{cyt}, \mathrm{ext}\}$. Each of these solutions is assumed to be smoothly extendable across the boundaries of its region of definition. Note that the actual solution $p(x,t)$ will not be smooth across either the nuclear or cellular membrane, but its value on either side of each of these membranes can be smoothly extended to the other side.

For spatial dimension d, let $\boldsymbol{i} \in \mathbb{Z}^d$ denote the index vector for mesh cells. With the smooth extension assumption, a cell-centered solution value, $p^{\alpha}_{\boldsymbol{i}}$, can always be defined within cells containing some portion of the domain α. For a cell completely within this domain only one solution value will be considered, $p^{\alpha}_{\boldsymbol{i}}$. If a cell is split by one of the membranes, then two cell-centered solution values will be stored for the cell. It is assumed that at most one boundary intersects a given cell. For example, in Figure 1, which shows the two-dimensional case for simplicity, cell $\boldsymbol{i} = (i,j)$ is split by the nuclear membrane; hence, there would be two solution values stored for this cell, $p^{\mathrm{nuc}}_{\boldsymbol{i}}$ and $p^{\mathrm{cyt}}_{\boldsymbol{i}}$. These two solution values are subsequently used to approximate the boundary condition on the membrane. They also allow, in the master equation formulation, separate probabilities of the particle being in either portion of the cell.

Again, as the particle cannot cross $\partial\Omega$, $p_{\boldsymbol{i}}^{\text{ext}} = 0$ for all $\boldsymbol{i}$. As we know $p_{\boldsymbol{i}}^{\text{ext}}$, multiple values need not be stored for cells cut by $\partial\Omega$. If the particle could cross the cellular membrane, then $p_{\boldsymbol{i}}^{\text{ext}}$ would not necessarily be zero, and multiple values would be needed in cells split by $\partial\Omega$.

Figure 1 defines several geometric quantities associated with cells that are used in the numerical discretization. Let $\boldsymbol{e}_k$ denote the unit vector along the kth coordinate axis of $\mathbb{R}^d$. Define by

$$\boldsymbol{F}(x,t) = -D\nabla p \tag{3.3}$$

the probability flux vector per unit area. Let $\boldsymbol{F}^\alpha$ be the flux vector of the domain α solution and $(F_k)^\alpha$ the kth component of $\boldsymbol{F}^\alpha$. $\boldsymbol{\eta}^\alpha$ will denote the outward pointing, with respect to domain α, normal to a given surface. Following the conservative discretization method in [21], for all cells, $\boldsymbol{i}$, in which $p_{\boldsymbol{i}}^\alpha$ is defined,

(3.4)

$$\begin{aligned}
\frac{dp_{\boldsymbol{i}}^\alpha}{dt} &= -(\nabla\cdot\boldsymbol{F}^{\boldsymbol{\alpha}})_i \\
&\approx -\frac{1}{V_{\boldsymbol{i}}^\alpha}\int_{V_{\boldsymbol{i}}^\alpha}\nabla\cdot\boldsymbol{F}^{\boldsymbol{\alpha}}\,dV = -\frac{1}{V_{\boldsymbol{i}}^\alpha}\int_{\partial V_{\boldsymbol{i}}^\alpha}\boldsymbol{F}^{\boldsymbol{\alpha}}\cdot\boldsymbol{\eta}^{\boldsymbol{\alpha}}\,dS \\
&= -\frac{1}{V_{\boldsymbol{i}}^\alpha}\left(\sum_{\pm}\sum_{k=1}^{d}\int_{A^\alpha_{\boldsymbol{i}\pm\boldsymbol{e}_k}}\boldsymbol{F}^{\boldsymbol{\alpha}}\cdot\boldsymbol{\eta}^{\boldsymbol{\alpha}},\ dS + \int_{B_n}\boldsymbol{F}^{\boldsymbol{\alpha}}\cdot\boldsymbol{\eta}^{\boldsymbol{\alpha}}\,dS + \int_{B_e}\boldsymbol{F}^{\boldsymbol{\alpha}}\cdot\boldsymbol{\eta}^{\boldsymbol{\alpha}}\,dS\right) \\
&\approx -\frac{1}{V_{\boldsymbol{i}}^\alpha}\left(\sum_{k=1}^{d}\left(A^\alpha_{\boldsymbol{i}+\frac{1}{2}\boldsymbol{e}_k}(F_k)^\alpha_{\boldsymbol{i}+\frac{1}{2}\boldsymbol{e}_k} - A^\alpha_{\boldsymbol{i}-\frac{1}{2}\boldsymbol{e}_k}(F_k)^\alpha_{\boldsymbol{i}-\frac{1}{2}\boldsymbol{e}_k}\right) + A_{\boldsymbol{i}}^{B_n}\boldsymbol{F}_{\boldsymbol{i}}^{B_n}\cdot\boldsymbol{\eta}_{\boldsymbol{i}}^{B_n}\right).
\end{aligned}$$

Here, $B_n = \partial\Omega_n \cap \partial V_{\boldsymbol{i}}^\alpha$ and $B_e = \partial\Omega \cap \partial V_{\boldsymbol{i}}^\alpha$. The surface integral over B_e is identically zero due to the no-flux boundary condition at $\partial\Omega$. If a nonzero flux boundary condition is specified instead, then the integral would contribute another term of the form

$$-\frac{1}{V_{\boldsymbol{i}}^\alpha}A_{\boldsymbol{i}}^{B_e}\boldsymbol{F}_{\boldsymbol{i}}^{B_e}\cdot\boldsymbol{\eta}_{\boldsymbol{i}}^{B_e}$$

in the last line of (3.4).

To give a valid reaction-diffusion master equation the fluxes are chosen as the standard full-face centered difference

$$(F_k)^\alpha_{\boldsymbol{i}\pm\frac{1}{2}\boldsymbol{e}_k} = \mp D\,\frac{p^\alpha_{\boldsymbol{i}\pm\boldsymbol{e}_k} - p^\alpha_{\boldsymbol{i}}}{h}.$$

From the nuclear membrane boundary condition

$$\boldsymbol{F}_{\boldsymbol{i}}^{B_n}\cdot\boldsymbol{\eta}_{\boldsymbol{i}}^{B_n} = -\rho\,[p]_{\boldsymbol{i}}^{B_n}.$$

To enforce this jump condition the cell-centered solution value for the domain on the other side of the nuclear membrane is used. For example, if $\alpha = \text{nuc}$, then $p_{\boldsymbol{i}}^{\alpha'}$ is used, where $\alpha' = \text{cyt}$. Vice versa, if $\alpha = \text{cyt}$, then $\alpha' = \text{nuc}$. Using this definition, the boundary condition is approximated by choosing

$$\rho\,[p]_{\boldsymbol{i}}^{B_n} \equiv \rho(p_{\boldsymbol{i}}^{\alpha'} - p_{\boldsymbol{i}}^\alpha).$$

With all the flux terms specified, the final discretization is then

$$\frac{dp_{\boldsymbol{i}}^{\alpha}}{dt} = \frac{D}{V_{\boldsymbol{i}}^{\alpha} h}\left(\sum_{\pm}\sum_{k=1}^{d} A_{\boldsymbol{i}\pm\frac{1}{2}\boldsymbol{e}_k}^{\alpha}\left(p_{\boldsymbol{i}\pm\boldsymbol{e}_k}^{\alpha} - p_{\boldsymbol{i}}^{\alpha}\right)\right) + \frac{A_{\boldsymbol{i}}^{B_n}\rho}{V_{\boldsymbol{i}}^{\alpha}}(p_{\boldsymbol{i}}^{\alpha'} - p_{\boldsymbol{i}}^{\alpha}). \tag{3.5}$$

For cells that do not intersect any boundary the discretization reduces to the standard five- (seven-) point discrete Laplacian in two dimensions (three dimensions).

Defining the probability of the particle being in the domain α portion of the $\boldsymbol{i}$th cell as

$$Q_{\boldsymbol{i}}^{\alpha}(t) = p_{\boldsymbol{i}}^{\alpha} V_{\boldsymbol{i}}^{\alpha} \approx \text{Prob}\{\boldsymbol{X}(t) \in V_{\boldsymbol{i}}^{\alpha} | \boldsymbol{X}(0) = \boldsymbol{x}_0\},$$

the discretization can be rewritten as

$$\frac{dQ_{\boldsymbol{i}}^{\alpha}}{dt} = \frac{D}{h}\left(\sum_{\pm}\sum_{k=1}^{d} \frac{A_{\boldsymbol{i}\pm\frac{1}{2}\boldsymbol{e}_k}^{\alpha}}{V_{\boldsymbol{i}\pm\boldsymbol{e}_k}^{\alpha}} Q_{\boldsymbol{i}\pm\boldsymbol{e}_k}^{\alpha} - \frac{A_{\boldsymbol{i}\pm\boldsymbol{e}_k}^{\alpha}}{V_{\boldsymbol{i}}^{\alpha}} Q_{\boldsymbol{i}}^{\alpha}\right) + A_{\boldsymbol{i}}^{B_n}\rho\left(\frac{Q_{\boldsymbol{i}}^{\alpha'}}{V_{\boldsymbol{i}}^{\alpha'}} - \frac{Q_{\boldsymbol{i}}^{\alpha}}{V_{\boldsymbol{i}}^{\alpha}}\right). \tag{3.6}$$

The first term in (3.6) represents diffusive motion between computational cells, while the second represents motion across the nuclear membrane within one computational cell. Let the two components of a split computational cell be represented as separate computational cells. In this form, all movement between computational cells within the same domain is by diffusive jumps, while movement between computational cells of different domains is by jumps determined by the membrane fluxes. Despite the appearance of the membrane flux terms, this equation, with an appropriate index relabeling to account for the multiple components of a split computational cell, still has the same form as (2.8). Hence, all motion is still represented as first order reactions, with jump rates given by the coefficients of the $Q_{\{\cdot\}}^{\{\cdot\}}$ terms.

Taking $A_{\boldsymbol{ij}}^{\alpha}$ to be the area of the domain α face shared by cells $\boldsymbol{i}$ and $\boldsymbol{j}$, the jump rate from the domain α component of cell $\boldsymbol{j}$ to the domain α component of cell $\boldsymbol{i}$ is

$$k_{\boldsymbol{ij}}^{\alpha} = \frac{D A_{\boldsymbol{ij}}^{\alpha}}{h V_{\boldsymbol{j}}^{\alpha}}. \tag{3.7}$$

For strictly interior cells this reduces to

$$k_{\boldsymbol{ij}}^{\alpha} = \frac{D}{h^2}.$$

The jump rate across the nuclear membrane in cell $\boldsymbol{i}$ from domain α' to domain α is

$$k_{\boldsymbol{i}}^{\alpha\alpha'} = \frac{A_{\boldsymbol{i}}^{B_n}\rho}{V_{\boldsymbol{i}}^{\alpha'}}. \tag{3.8}$$

A similar equation holds for the jump from domain α to domain α'. Note that $k_{\boldsymbol{ij}}^{\alpha}$ and $k_{\boldsymbol{ji}}^{\alpha}$ are unequal in general because $V_{\boldsymbol{i}}^{\alpha}$ may not be equal to $V_{\boldsymbol{j}}^{\alpha}$. Similarly, $k_{\boldsymbol{i}}^{\alpha\alpha'}$ and $k_{\boldsymbol{i}}^{\alpha'\alpha}$ will also differ as $V_{\boldsymbol{i}}^{\alpha}$ may not be equal to $V_{\boldsymbol{i}}^{\alpha'}$.

There is a fundamental difference in the scaling between (3.7), which scales like $1/h^2$, and (3.8), which scales like $1/h$. This difference arises because the flux for diffusive motion is proportional to the [illegible] of the density, while the flux across the

nuclear membrane is proportional to the $\boldsymbol{\prime\prime}$ $\ldots$ $\boldsymbol{\prime}$ in the density across the membrane.

A consequence of the conservative discretization form is that total probability is conserved:

$$\sum_{\alpha}\sum_{\boldsymbol{i}\in\alpha}\frac{dQ_{\boldsymbol{i}}^{\alpha}}{dt}=\sum_{\alpha}\sum_{\boldsymbol{i}\in\alpha}\frac{dp_{\boldsymbol{i}}^{\alpha}}{dt}V_{\boldsymbol{i}}^{\alpha}=0.$$

The master equation approximation will also satisfy the principle of detailed balance—the statement that at thermodynamic equilibrium the unidirectional probability flux from cell $\boldsymbol{i}$ to cell $\boldsymbol{j}$ is equal to the unidirectional probability flux from cell $\boldsymbol{j}$ to cell $\boldsymbol{i}$. We expect detailed balance to hold, as (3.1) forms a closed and isolated system due to the no-flux cellular membrane condition and passive nuclear membrane flux (no active transport). Letting $p^{eq}(x,t)=p^{eq}$ be the constant equilibrium solution to the continuous problem, $(Q^{eq})_{\boldsymbol{i}}^{\alpha}=p^{eq}V_{\boldsymbol{i}}^{\alpha}$ will be the equilibrium probability of the Brownian particle being in the domain α component of cell $\boldsymbol{i}$. For $A_{\boldsymbol{ij}}^{\alpha}$, the area of the domain α face between cells $\boldsymbol{i}$ and $\boldsymbol{j}$, the detailed balance condition is

$$\begin{aligned}
&k_{\boldsymbol{ji}}^{\alpha}(Q^{eq})_{\boldsymbol{i}}^{\alpha}=k_{\boldsymbol{ij}}^{\alpha}(Q^{eq})_{\boldsymbol{j}}^{\alpha}\\
\iff&\frac{DA_{\boldsymbol{ij}}^{\alpha}}{hV_{\boldsymbol{i}}^{\alpha}}(Q^{eq})_{\boldsymbol{i}}^{\alpha}=\frac{DA_{\boldsymbol{ij}}^{\alpha}}{hV_{\boldsymbol{j}}^{\alpha}}(Q^{eq})_{\boldsymbol{j}}^{\alpha}\\
\iff&\frac{(Q^{eq})_{\boldsymbol{i}}^{\alpha}}{V_{\boldsymbol{i}}^{\alpha}}=\frac{(Q^{eq})_{\boldsymbol{j}}^{\alpha}}{V_{\boldsymbol{j}}^{\alpha}},
\end{aligned}$$

which holds by the definitions of $(Q^{eq})_{\boldsymbol{i}}^{\alpha}$ and $(Q^{eq})_{\boldsymbol{j}}^{\alpha}$. For jumps across a membrane between domains α and α',

$$\begin{aligned}
&k_{\boldsymbol{i}}^{\alpha'\alpha}(Q^{eq})_{\boldsymbol{i}}^{\alpha}=k_{\boldsymbol{i}}^{\alpha\alpha'}(Q^{eq})_{\boldsymbol{i}}^{\alpha'}\\
\iff&\frac{A_{\boldsymbol{i}}^{B_n}\rho}{V_{\boldsymbol{i}}^{\alpha}}(Q^{eq})_{\boldsymbol{i}}^{\alpha}=\frac{A_{\boldsymbol{i}}^{B_n}\rho}{V_{\boldsymbol{i}}^{\alpha'}}(Q^{eq})_{\boldsymbol{i}}^{\alpha'}\\
\iff&\frac{(Q^{eq})_{\boldsymbol{i}}^{\alpha}}{V_{\boldsymbol{i}}^{\alpha}}=\frac{(Q^{eq})_{\boldsymbol{i}}^{\alpha'}}{V_{\boldsymbol{i}}^{\alpha'}},
\end{aligned}$$

which again holds by the definitions of $(Q^{eq})_{\boldsymbol{i}}^{\alpha}$ and $(Q^{eq})_{\boldsymbol{i}}^{\alpha'}$. Note that an active transport boundary condition at the cellular membrane could make the system open, in which case detailed balance would no longer be expected to hold. Moreover, an active transport mechanism at the nuclear membrane would also prevent the system from coming to thermodynamic equilibrium and would result in a steady state in which the above relationships across the nuclear membrane would be violated. This would not affect the rate constants for other (passive) processes, however, which would therefore still satisfy the conditions derived from the principle of detailed balance.

For comparison to our discretization, note that in [21] $(F_k)_{\boldsymbol{i}\pm\frac{1}{2}\boldsymbol{e}_k}$ is chosen to be the standard centered difference approximation when the kth face does not intersect a boundary. This leads to the five- (seven-) point Laplacian as the discretization for strictly interior cells. For faces cut by the boundary, [21] interpolates between the centered difference flux in neighboring cells to approximate the flux at the midpoint of the cut face. This method leads to a second order accurate approximation to the Laplacian, but does not give a valid master equation for the continuous time–discrete space discretization. To see why, note that the general form of the master equation

for the probability, Q_i, of being at site i with transition probability per unit time $W_{ji} \geqslant 0$ is given by (2.2). In general interpolation will introduce negative weights W_{ji} or W_{ij} into the ith equation. Furthermore, interpolation violates the condition that if the ith equation contains a term $W_{ij}Q_j$, then the jth equation should contain a term $-W_{ij}Q_j$.

An alternative discretization method is presented in [13] that accounts for the boundary by using the standard five- (seven-) point Laplacian at all locations and adding an extra forcing term. As the master equation has no forcing terms this method would not give an equation that could be realized by the Gillespie method.

3.3. Overall method. Using the discretization from the previous section, realizations of the stochastic process described by the reaction-diffusion master equation can be created using the Gillespie method. The overall simulation algorithm is as follows:

1. Initialization:
   (a) Given the membrane locations, calculate $A^{\alpha}_{\boldsymbol{i}\pm\boldsymbol{e}_k}$, $V^{\alpha}_{\boldsymbol{i}}$, and $A^{B_n}_{\boldsymbol{i}}$ for each location, $\boldsymbol{i}$, each direction, $\boldsymbol{e}_k$, and each domain, α.
   (b) From (3.7) and (3.8) calculate the jump rates for all species, within all Cartesian cells containing some part of Ω.
   (c) For each piece of a Cartesian cell, calculate the rates of all chemical reactions that can occur there. For reactions with volume dependent rates use $V^{\alpha}_{\boldsymbol{i}}$ to change the rate constants to units of reciprocal time.
2. Time Evolution:
   (a) Simulate individual realizations of the stochastic process described by the reaction-diffusion master equation using the Gillespie method. Diffusive and transmembrane solute motion are represented as first order reactions, using the jump rates calculated in step 1(b). Within each component of a cell, the rates from step 1(c) are used to simulate chemical reactions.
3. Output:
   (a) To estimate moments or distributions, use statistics from many simulations.

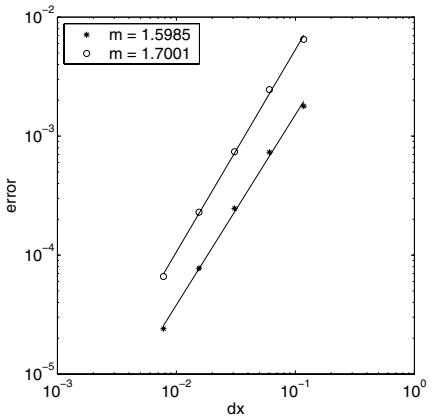
3.4. Convergence of the numerical discretization. The convergence of the discretization provided by (3.5) is illustrated in two dimensions for both the Poisson equation and diffusion equation with circular boundaries. The method was found to converge between first and second order spatially. Let $(\boldsymbol{u}_h)^{\alpha}_{\boldsymbol{i}}$ denote the numerical solution to the Poisson problem, and $\boldsymbol{u}_h$ the vector whose components are given by $(\boldsymbol{u}_h)^{\alpha}_{\boldsymbol{i}}$. Define the maximum norm, $\|\boldsymbol{u}_h\|_{\infty}$, as

$$\|\boldsymbol{u}_h\|_{\infty} = \max_{\alpha,\boldsymbol{i}\in\alpha} |(\boldsymbol{u}_h)^{\alpha}_{\boldsymbol{i}}|.$$

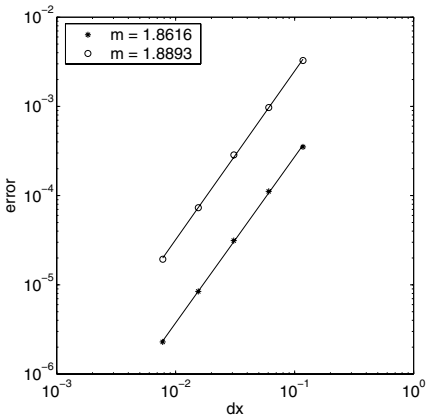
A volume weighted two norm, $\|\boldsymbol{u}_h\|^w_2$, is defined as

$$\|\boldsymbol{u}_h\|^w_2 = \left(\sum_{\alpha} \sum_{\boldsymbol{i}\in\alpha} \left((\boldsymbol{u}_h)^{\alpha}_{\boldsymbol{i}}\right)^2 V^{\alpha}_{\boldsymbol{i}} \right)^{\frac{1}{2}}.$$

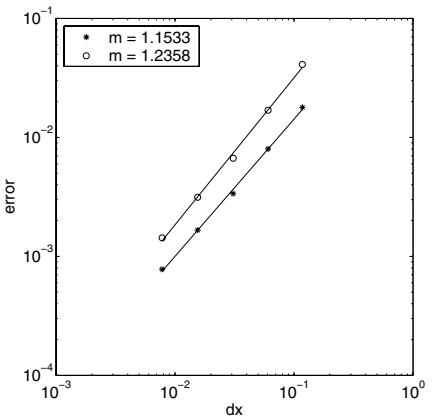
Let $(\boldsymbol{u}^n_h)^{\alpha}_{\boldsymbol{i}}$ denote the numerical solution to the diffusion equation at time t_n, $\boldsymbol{u}^n_h$ the solution vector at time t_n, and $\boldsymbol{u}_h$ the solution over all spatial locations, domains, and times. The maximum norm over all time is defined as $\max_n \|\boldsymbol{u}^n_h\|_{\infty}$. As in section 3.1,



(a) Error for (3.9), with exact solution (3.10).



(b) Error for (3.9), with exact solution (3.11).



(c) Error for (3.12), with exact solution (3.13).

FIG. 2. *Convergence results for Poisson equation solutions; star is two norm error and circle is maximum norm error. These errors are plotted against the mesh width dx on log-log plots. The empirical order of accuracy is m, which is the slope of the best fit straight line in each case.*

$\partial\Omega$ will denote the cell membrane, or outer boundary, while $\partial\Omega_n$ will denote the nuclear membrane, or inner boundary. In all of the examples considered in this section, the two boundaries are concentric circles with radii 1 and 1/2, respectively. The domain into which the circles are embedded is taken, for a given mesh width h, to be the square centered at the origin with sides of length $2 + 4h$. This provides a separation between the boundary of the computational domain and the embedded boundaries.

Figure 2(a) shows the convergence of the discretization for the Poisson equation

with just the outer boundary. The equations are

$$\begin{aligned} \Delta u &= f(\boldsymbol{x}) \quad \text{in } \Omega, \\ \frac{\partial u}{\partial \boldsymbol{\eta}} &= 0 \qquad \text{on } \partial\Omega, \end{aligned} \tag{3.9}$$

where $f(\boldsymbol{x})$ was chosen so that the exact solution was

$$u = \frac{r^4}{4} - \frac{r^3}{3}, \quad r \le 1, \tag{3.10}$$

with $u = 0$ for $r > 1$. The two norm error converged at order 1.6, while the maximum norm error converged at order 1.7.

An angular dependence was added to $f(\boldsymbol{x})$ to give the exact solution

$$u = \left(\frac{r^4}{4} - \frac{r^3}{3}\right)\cos\theta, \quad r \le 1, \tag{3.11}$$

with $u = 0$ for $r > 1$. Figure 2(b) shows the convergence of the errors for this problem. The θ dependence actually improves the convergence, so that the two norm error is order 1.86, while the maximum norm error now converges at order 1.89.

Figure 2(c) shows the convergence of the discretization for the Poisson equation with both boundaries. The equations are

$$\begin{aligned} \Delta u &= f(\boldsymbol{x}) \qquad \text{in } \Omega, \\ \frac{\partial u}{\partial \boldsymbol{\eta}} &= 0 \qquad \text{on } \partial\Omega, \\ -\frac{\partial u}{\partial \boldsymbol{\eta}} &= -\pi\,[u]_n \quad \text{on } \partial\Omega_n, \end{aligned} \tag{3.12}$$

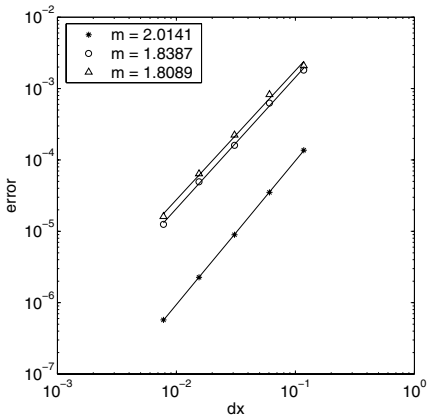
where $f(\boldsymbol{x})$ was chosen so that the exact solution was

$$u = \begin{cases} \cos \pi r + \sin \pi r & \text{in } \Omega_n, \\ \cos \pi r & \text{in } \Omega \setminus \Omega_n, \\ 0 & \text{outside } \Omega. \end{cases} \tag{3.13}$$

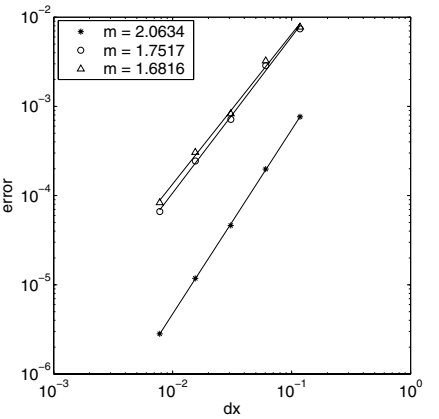
The jump condition at the interior boundary reduces the order of convergence to 1.15 in the two norm and 1.23 in the maximum norm.

Figure 3(a) shows the convergence of the discretization for the diffusion equation with just the exterior boundary. The final time, at which the two norm and maximum norm errors were measured, was taken to be $t_f = .025$. At this time the solution was still far from equilibrium. $\Delta t/h$ was fixed at about t_f/L, where L was the diameter of the outer boundary. For the examples in Figure 3, $\Delta t/h \approx .013$. The time integration was performed using the second order, L_0 stable, implicit Runge–Kutta method mentioned in [21]. Similar results were obtained for a Crank–Nicolson discretization. The equations are

$$\begin{aligned} \frac{\partial u}{\partial t} &= \Delta u + f(\boldsymbol{x}, t) \quad \text{in } \Omega, \\ \frac{\partial u}{\partial \boldsymbol{\eta}} &= 0 \qquad\qquad\quad \text{on } \partial\Omega, \end{aligned} \tag{3.14}$$



(a) Error for (3.14), with exact solution (3.15).



(b) Error for (3.16), with exact solution (3.17).

FIG. 3. *Convergence results for diffusion equation solutions; star is two norm error at t_f, circle is the maximum norm error at t_f, and diamond is the maximum of the maximum norm error over all time. These errors are plotted against the mesh width dx on log-log plots. The empirical order of accuracy is m, which is the slope of the best fit straight line in each case.*

where $f(\boldsymbol{x}, t)$ was chosen so that the exact solution was

$$u = \left(\frac{r^4}{4} - \frac{r^3}{3}\right) e^{-20t}, \quad r \leq 1, \tag{3.15}$$

with $u = 0$ for $r > 1$. At t_f, the two norm error converged at about order 2.01, and the maximum norm error converged at about order 1.84. The maximum over all time points of the maximum norm converged at about order 1.81.

Figure 3(b) shows the convergence of the discretization for the diffusion equation with both boundaries. Note that the same time discretization was used as above, again with $\Delta t/h \approx .013$. The equations are

$$\begin{aligned} \frac{\partial u}{\partial t} &= \Delta u + f(\boldsymbol{x}, t) && \text{in } \Omega, \\ \frac{\partial u}{\partial \boldsymbol{\eta}} &= 0 && \text{on } \partial\Omega, \\ -\frac{\partial u}{\partial \boldsymbol{\eta}} &= -\left[u\right]_n && \text{on } \partial\Omega_n, \end{aligned} \tag{3.16}$$

where $f(\boldsymbol{x}, t)$ was chosen so that the exact solution was

$$u = \begin{cases} \left(\frac{br^4}{4} + \frac{ar^3}{3} + \frac{ar^5}{5}\right) e^{-20t} & \text{in } \Omega_n, \\ \left(\frac{r^3}{3} - \frac{r^4}{4}\right) e^{-20t} & \text{in } \Omega \setminus \Omega_n, \\ 0 & \text{outside } \Omega. \end{cases} \tag{3.17}$$

a and b were chosen to satisfy the jump boundary condition at $r = 1/2$. At t_f the two norm error converged at about order 2.06, and the maximum norm error converged at

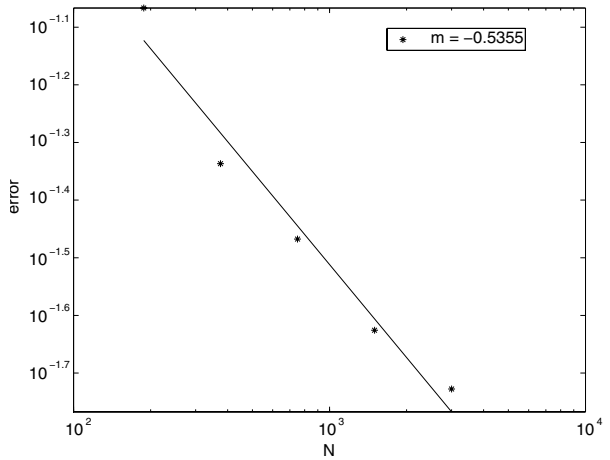


FIG. 4. *Convergence of the mean of Gillespie simulations to the numerical solution to* (3.1). *Here the error is given by the maximum absolute difference over all times and spatial locations between the Gillespie simulation estimate and the numerical solution. A* 37 *by* 37 *mesh was used with mesh width* 2/33, $D = 1$, *and* $\rho = 1$. *The particle was started at the center of the domain for all simulations. m gives the slope of the best fit line to the data.*

about order 1.75. The maximum over all time points of the maximum norm converged at about order 1.68.

3.5. Convergence of the overall method. The overall method presented in section 3.3 has two sources of error associated with it: sampling error and error due to the spatial discretization. The Gillespie method provides exact realizations of the stochastic process defined by the reaction-diffusion master equation, hence the error in any individual realization is due solely to the spatial discretization. From the central limit theorem, for a fixed mesh size, we expect the error between the mean population levels from N simulations and the means of the reaction-diffusion master equation to be approximately $O(1/\sqrt{N})$. Note that the constant in the $O(1/\sqrt{N})$ term will depend on the mesh size. With a sufficient number of samples the joint probability distribution, for a fixed mesh size, can be estimated to any desired accuracy, but the required number of samples might be very large.

As an example, consider a single particle moving within a two-dimensional biological cell. Assume the cell has a circular cellular membrane of radius 1, and a circular nuclear membrane of radius 1/2. The particle's dynamics are assumed to be given by (3.1), with $D = 1$ and $\rho = 1$. Since there is only one particle, the mean number of the particle in a given region is just the probability of the particle being in that region. As the reaction-diffusion master equation is in terms of the population levels of the particle at different locations, its mean should converge to the solution of (3.1) as h goes to zero. Figure 4 shows the maximum error over all times between the mean of N Gillespie simulations and the numerical solution to (3.1), as the number of samples is increased. Here h is fixed at 2/33, for a 37 by 37 mesh. Notice that the error decreases like $1/\sqrt{N}$.

Having obtained the distribution given by the reaction-diffusion master equation, there is still the question of how accurate a representation of the dynamics of the system it gives. It is not clear that the reaction-diffusion master equation has a well-defined continuum limit as h goes to zero. Instead, it is generally considered valid

only for a range of h values, and there are several simultaneous physical conditions which h should satisfy. First, h should be significantly larger than the mean free path, λ, between the elastic collisions driving diffusion. This condition ensures that the system can be considered in local equilibrium within a computational cell because of many nonreactive collisions. Note, however, that to ensure that the system is accurately resolving local behavior, and also that the approximation of diffusion as jumps between cells is reasonable, h should be significantly smaller than the length scale of the entire system, L. Summarizing these constraints,

$$L \gg h \gg \lambda.$$

As we now show, h also needs to be chosen large enough (!) that the system can be considered in local equilibrium and well mixed within each mesh cell on the time scale of the fastest bimolecular reactions. This assumption underlies the use of stochastic chemical kinetics independently within each mesh cell. For this to hold, the time scale for the particle to diffuse throughout a mesh cell should be much faster than the time scale for the fastest bimolecular chemical reaction. The time scale to diffuse throughout a region of length scale h is approximately h^2/D. For a bimolecular reaction, the time scale with rate constant k is h^3/k for a mesh cell of volume h^3. (Recall that the units of the rate constant of a bimolecular reaction are volume per unit time.) Hence, it is necessary that

$$\frac{h^3}{k} \gg \frac{h^2}{D},$$

which implies that

$$h \gg \frac{k}{D}.$$

Thus, the rate constant for a bimolecular reaction places a fundamental lower limit on the spatial size of mesh cells. Fortunately, for cellular processes and the molecules involved in them, k/D is generally not large enough to impose a significant restriction on h. In summary, letting k denote the rate constant of the fastest bimolecular reaction,

$$L \gg h \gg \max\left(\lambda, \frac{k}{D}\right).$$

In these inequalities D is the diffusion coefficient of a particular molecular species and k refers to the fastest bimolecular reaction in which that species participates. Such inequalities must hold simultaneously for all species under consideration. Note that a first order reaction does not restrict the mesh size in any manner since it generally represents an internal molecular event, and is not dependent on the system being well mixed locally.

As of yet, the reaction-diffusion master equation has not been derived from more microscopic physical models. Its validity has been verified numerically in [5] through comparison with microscopic hard sphere models that track individual particle positions and momenta. One might also hope that, for sufficiently fast diffusion, the results from the reaction-diffusion master equation applied to an initially well-mixed system would agree over a range of h values with the spatially homogeneous chemical master equation. For example, consider the simple chemical reaction $A + B \to C$,

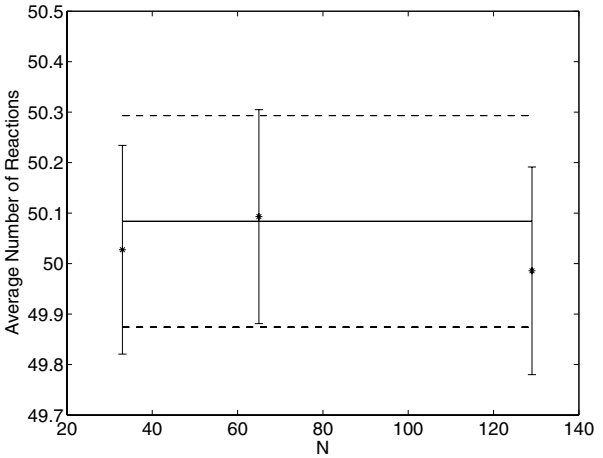


FIG. 5. *Average number of reactions in stochastic diffusion simulations, given by stars, compared to the exact solution for the expected number of reactions of the spatially homogeneous chemical master equation, given by the solid straight line. Error bars give* 99.7 *percent confidence intervals about the data points. The error bars are calculated using the sampled variance. N gives the number of mesh points in each direction. For each data point,* 3000 *sample realizations were used to calculate the average number of reactions. The dashed lines give the predicted* 99.7 *percent confidence interval, for* 3000 *samples, about the exact mean using the exact variance of the well-mixed master equation. Data is from* $t = .1$*, a time at which approximately half the A and half the B chemicals have been converted to C. Note the drastically expanded vertical scale.*

within a 1 unit by 1 unit square. The exact solution, at any time, for the average number of reactions in the spatially homogeneous chemical master equation is given in [18]. This reduces the statistical error in the problem to that in estimating the average number of reactions from the reaction-diffusion master equation. 100 molecules of A and 100 of B are each started randomly distributed throughout the domain. No-flux boundary conditions were assumed for the diffusion of chemicals, but periodic boundary conditions were also tested with no significant impact on the results. With bimolecular rate constant, .0001, and a diffusion constant of 100 for each species, $k/D = .000001$. Hence, by the preceding arguments h should only need to be greater than .000001 for the reaction-diffusion master equation to be valid. Figure 5 shows that as the number of mesh points, N, is increased, the average number of reactions from spatial simulations agrees with the exact number of reactions given by the spatially homogeneous chemical master equation to statistical error.

4. Transcription, translation, transport model. As an application of the method presented in section 3.3 we present a model of transcription, nuclear export, translation, and nuclear import in eukaryotes. While the model is simplified, and uses transcription and translation models that are more appropriate for a prokaryotes, it demonstrates the feasibility of the method in simulating biological networks. Section 4.1 describes the basic molecular cell biology necessary to understand the biochemical reaction model of section 4.2. Section 4.3 shows the time evolution of one realization of the model.

4.1. Background biology. The process by which a functional protein is produced from a segment of DNA that codes for it, called a gene, is a complex series of chemical events. In a coarsest description, the gene is first [illegible] to produce

a copy of itself called an mRNA. The mRNA is a linear sequence of nucleotides, every three of which comprise an element of a code representing an amino acid. The mRNA is then translated by special cellular machinery whereby the linear sequence of nucleotides is read, and the corresponding sequence of amino acids that the mRNA codes for is assembled. The polypeptide chain of amino acids typically undergoes further chemical and structural modifications to then form a functional protein.

For eukaryotes this situation is more complicated: transcription occurs in the nucleus and translation occurs in the cytoplasm of the cell. Between transcription and translation there is then an additional step that is required involving the movement of the mRNA out of the nucleus. This nuclear export process requires a series of chemical events; it cannot occur simply through diffusion of the mRNA. Similarly, those proteins that influence gene expression must be imported into the nucleus to be effective.

Large protein complexes form pores in the nuclear membrane through which all transmembrane traffic flows. These nuclear pore complexes are normally only wide enough for small molecules of less than 9 nanometers to pass through by free diffusion. Large molecules, such as mRNAs and most proteins, require the assistance of nuclear export and import receptors, which allow the pores effectively to dilate so that molecules of up to 26 nanometers can pass [1].

We shall be interested in the export of newly transcribed mRNA from the nucleus, and the import of the resulting protein product back into the nucleus. These two pathways are believed to be distinct in the nuclear receptors used, and in the biasing factors that confer directionality to the process (so that, for example, newly exported mRNA does not get reimported into the nucleus). Since the general mRNA export pathway is still unsettled, we shall use the same pathway for modeling mRNA export and protein import. This pathway is driven by a nuclear membrane RanGTP gradient that confers directionality to the movement of cargo across the membrane. A subset of the full RanGTP cycle model presented in [24] is used to account for the movement of receptors and their cargo. The RanGTP nuclear transport process is sometimes referred to as "active" because there is an energy expenditure in maintaining the gradient of RanGTP across the nuclear membrane. In contrast, the actual movement of cargo across the nuclear membrane is believed to be passive. Note that this pathway is used for the export of some mRNAs, for example, the export of incompletely spliced HIV mRNA encoding viral structural proteins [7].

The basic model presented in the next section is built on the pathway of transcription of one gene to produce mRNA, nuclear export of the mRNA, translation of the mRNA within the cytoplasm, and import of the resulting protein product back into the nucleus. Protein import plays an important role in many transcriptional networks, allowing the protein product of one gene to bind and regulate the expression of another. Several important steps in the transcription-translation cycle for eukaryotes are not modeled, including splicing, the opening of the chromatin to allow physical access to the gene, and the assembly of the transcription initiation complex.

4.2. Chemical model. The eukaryotic cell is modeled in two dimensions as two concentric circles, representing the plasma and nuclear membranes. The plasma membrane radius is taken to be 11.81 μm, while the nuclear membrane radius is 5 μm. The transcription and translation models used are based on those presented in [8]. Specifically, the gene of interest has several states. DNA denotes that the gene is free of RNA polymerase II, hereafter RNAP, the enzyme which reads the nucleotides comprising the gene and then incorporates the corresponding nucleoside triphospate

into the growing mRNA transcript. DNA^0 will denote that the RNAP is bound to the gene's promoter and ready to begin transcription. DNA^l will denote that the first l of the nucleotides forming the gene have been read and incorporated into the mRNA, with M giving the total number of nucleotides in the gene. Finally, the number of mRNA molecules within spatial computational cell i is given by $mRNA_i$. For simplicity, the gene is assumed to be localized in the center of the nucleus, which will have index i_{center}. The concentration of both RNAP and the nucleoside triphospates that are added to the mRNA transcript are assumed fixed so that transitions between the different DNA states are first order reactions. With these definitions, the transcriptional reactions, all defined solely at the cell center, are then

$$\begin{aligned} DNA &\to DNA^0, \\ DNA^{l-1} &\to DNA^l, \quad l = 1, \ldots, M-1, \\ DNA^{L-1} &\to mRNA_{i_{\text{center}}} + DNA. \end{aligned}$$

The rates for these reactions are, respectively, $.05\,\text{s}^{-1}$, $30\,\text{s}^{-1}$, and $30\,\text{s}^{-1}$. The number of nucleotides in the gene, M, is chosen to be 999.

Newly transcribed mRNA is assumed to diffuse freely throughout the nucleus until entering the export pathway. The mRNA export model is based on the RanGTP export system. Denote by NR the nuclear export receptor to which mRNA can bind; by Rt, RanGTP; by Rd, RanGDP; by Rb, RanBP1; and by NR–Rt the nuclear receptor complexed with RanGTP. All five are assumed to be at steady state concentrations, and uniformly distributed throughout the nucleus and cytoplasm. Note that the steady state concentrations are not assumed to be the same within the nucleus and cytoplasm, simply within each individually. Steady state concentrations and subsequent reaction rates were based on the data in [24]. The export process consists of three reactions, beginning with the binding of the export receptor-RanGTP complex to nuclear mRNA. Once this has occurred, the nuclear membrane is assumed to be permeable to the complex. At all locations within the cytoplasm the complex can then bind RanBP1, which subsequently induces the release of the mRNA from the complex. The reactions are

$$\begin{aligned} mRNA_i + NR\text{–}Rt_i &\to mRNA\text{–}NR\text{–}Rt_i && \text{within nucleus,} \\ mRNA\text{–}NR\text{–}Rt_i + Rb_i &\rightleftarrows mRNA\text{–}NR\text{–}Rt\text{–}Rb_i && \text{within cytoplasm,} \\ mRNA\text{–}NR\text{–}Rt\text{–}Rb_i &\to mRNA_i + NR_i + Rb_i + Rd_i && \text{within cytoplasm.} \end{aligned}$$

Using the assumption of constant concentrations, we can reduce these reactions to the following four:

$$\begin{aligned} mRNA_i &\to mRNA\text{–}NR\text{–}Rt_i && \text{within nucleus,} \\ mRNA\text{–}NR\text{–}Rt_i &\rightleftarrows mRNA\text{–}NR\text{–}Rt\text{–}Rb_i && \text{within cytoplasm,} \\ mRNA\text{–}NR\text{–}Rt\text{–}Rb_i &\to mRNA_i && \text{within cytoplasm.} \end{aligned}$$

The rates for these reactions are, respectively, $1184.5\,\text{s}^{-1}$, $298.93\,\text{s}^{-1}$ for the forward, $.5\,\text{s}^{-1}$ for the reverse, and $51.19\,\text{s}^{-1}$.

Once the mRNA is free within the cytoplasm, it can then associate with ribosomes, the molecular machines on which translation occurs. The ribosomes are assumed to be uniformly distributed throughout the cytoplasm as a constant background concentration. The translation process begins with the binding of an mRNA

to a ribosome to form a complex that is ready to translate, denoted by $mRNA_i^0$. As the ribosomes are very large, it is assumed that the translation process is immobile. Translation then consists of a series of elongation steps by which the mRNA is moved through the ribosome and read, and each set of three nucleotides is translated into the appropriate amino acid. This amino acid is then added to the growing polypeptide that will become the protein. Let $mRNA^l$ denote that l amino acids have been incorporated into the polypeptide chain, $l = 1, \ldots, N$, where $N = M/3$. Denote by P protein. Assuming all amino acid concentrations are spatially homogeneous and at steady state, the translation process reduces to the following set of reactions:

$$\begin{aligned} mRNA_i &\to mRNA_i^0 && \text{within cytoplasm,} \\ mRNA_i^{l-1} &\to mRNA_i^l, \quad l = 1, \ldots, N-1, && \text{within cytoplasm,} \\ mRNA_i^{N-1} &\to P_i + mRNA_i && \text{within cytoplasm.} \end{aligned}$$

The rates for these reactions are, respectively, $.5\,\mathrm{s}^{-1}$, $33\,\mathrm{s}^{-1}$, and $33\,\mathrm{s}^{-1}$.

Protein may freely diffuse within both the nucleus and the cytoplasm; however, it may not freely cross the nuclear membrane. Instead, it uses the RanGTP nuclear import process to enter the nucleus. This process consists of two reactions: protein binding to nuclear import receptor within the cytoplasm, and unbinding of protein from the receptor upon binding of RanGTP within the nucleus. The reactions are then

$$\begin{aligned} P_i + NR_i &\to P\text{–}NR_i && \text{within cytoplasm,} \\ P\text{–}NR_i + Rt_i &\to P_i + NR\text{–}Rt_i && \text{within nucleus.} \end{aligned}$$

Using the constant concentration assumption, these reactions can be reduced to the following first order reactions:

$$\begin{aligned} P_i &\to P\text{–}NR_i && \text{within cytoplasm,} \\ P\text{–}NR_i &\to P_i && \text{within nucleus.} \end{aligned}$$

The rates for these reactions are, respectively, $1218.25\,\mathrm{s}^{-1}$ and $1.23\,\mathrm{s}^{-1}$.

A feedback mechanism is incorporated into the model by allowing the protein to repress the expression of its own gene. Denoting by DNA_{rep} the repressed state of the gene, the reactions are

$$DNA + P_{i_{center}} \rightleftarrows DNA_{\mathrm{rep}}.$$

Here the forward reaction rate is $.01\,\mu\mathrm{m}^3\mathrm{s}^{-1}$ and the reverse rate is $.01\,\mathrm{s}^{-1}$.

Finally, mRNA is assumed to be degradable within the cytoplasm with rate $.2\,\mathrm{s}^{-1}$, and protein anywhere with rate $.0025\,\mathrm{s}^{-1}$. Table 4.1 gives the diffusion and nuclear membrane permeability rates for the different species in the model. Note that because of the finite diffusion coefficients, this model will give different statistics than the corresponding well-mixed model. For example, in the nuclear export model the time for an mRNA-nuclear export receptor complex to diffuse throughout the nucleus is approximately

$$\frac{r_{\mathrm{nuc}}^2}{D} = 250 \text{ s}.$$

TABLE 4.1
Diffusion constant, D, and nuclear membrane permeability, ρ, for each species.

Species	D $\mu\text{m}^2\,\text{s}^{-1}$	ρ $\mu\text{m}\,\text{s}^{-1}$
$mRNA$	.1	0
$mRNA^l$	0	0
$mRNA$–NR–Rt	.1	.17
$mRNA$–NR–Rt–Rb	.1	0
P	20.0	0
P–NR	20.0	1.87

In contrast, the time scale for that complex to leave the nucleus in the well-mixed model would be approximately

$$\frac{\frac{4}{3}\pi r_{\text{nuc}}^3}{4\pi r_{\text{nuc}}^2\,\rho} = 9.8 \text{ s}.$$

With the assumed biological constants the two step export process, finding the nuclear membrane and then passing through it, is strongly diffusion limited. Therefore, a well-mixed model with infinitely fast diffusion would ignore the long time scale for an export complex to find the nuclear membrane, much less to get through it. This would result in a substantially faster export process than biologically occurs.

4.3. Numerical results. Figures 6 and 7 show the time evolution of one realization of the model described in the previous section, using an underlying 37 by 37 Cartesian mesh with a mesh width of approximately .72 microns. Within the simulations we calculate the number of molecules of each chemical species within each mesh cell. We assume that each individual mesh cell particles are well mixed, and as such the marker for every molecule in each figure is placed randomly within the mesh cell containing that molecule. Initially there are no mRNAs or proteins within the system, and the DNA is in the unbound state. Figure 6 shows the evolution of the system over several minutes. By Figure 6(a) the first mRNA has been fully transcribed and, because of the fast binding rate, is bound to a nuclear export receptor. The DNA is unbound, waiting for the next transcription cycle to begin. After 160 seconds, a second mRNA has been transcribed and bound with a nuclear receptor. The DNA is again unbound. By 310 seconds several proteins have been translated. One is diffusing within the cytoplasm bound to a nuclear receptor, while the other has already undergone the nuclear import process and is diffusing freely within the nucleus. Notice there is only one visible mRNA, as the second is undergoing translation within the cytoplasm and hence not displayed. The DNA is also not visible as it is undergoing transcription. Finally, after 380 seconds, several proteins have accumulated within the nucleus, with one binding to and inhibiting the DNA. A free mRNA is diffusing within the cytoplasm, along with a nuclear receptor coupled protein.

By allowing the protein to feed back and inhibit the transcription of its gene, we create a system in which protein production occurs in bursts. Figure 7 shows that a large amount of protein is built up in the nucleus by 1160 seconds, and that the DNA is in the repressed state. The higher protein levels increase the probability of the gene becoming repressed, and because the on rate for binding is sufficiently fast, and the off rate sufficiently slow, by 1670 seconds the overall protein population has shrunk from 10 to just 2. This is due in part to the fast decay rates of the mRNA. As the DNA is repressed, no mRNAs are transcribed to replace those that quickly decay, and hence the protein population simply decays away over time. Once the protein

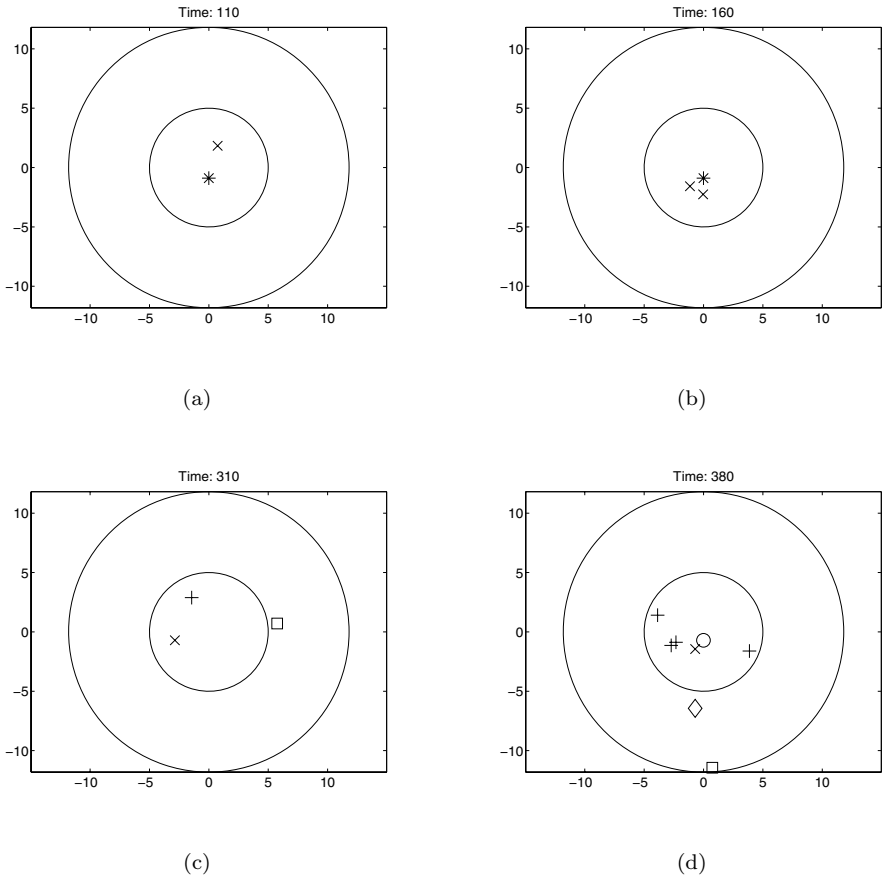


FIG. 6. *Evolution of one realization of the model from section* 4.2 *over several minutes. A star denotes the unbound DNA and a circle that the DNA is repressed. During transcriptional states the DNA is not displayed. A diamond denotes mRNA, and "×'s" denote mRNA bound to nuclear receptor and RanGTP. mRNAs coupled to nuclear receptors, RanGTP, and RanBP*1 *are not present in the images shown. During translation mRNAs are not displayed. A plus denotes a protein, and a box represents protein bound to nuclear receptor. Time is in seconds.*

population becomes sufficiently low, as in Figure 7(b), it becomes more unlikely that protein will find the DNA to bind and repress it, and hence the DNA is able to enter the transcription cycle. By 2010 seconds at least one mRNA has been transcribed and subsequently translated several times, as the nuclear protein level has increased. The production of mRNA followed by its subsequent translation produces new proteins, and consequently the nuclear protein population rises thereafter. By 2080 seconds, 10 protein molecules are within the nucleus.

Figure 8 shows the total number of nuclear proteins in one realization, over 30,000 seconds. The total number is given by the amount of free, receptor-bound, and DNA-bound proteins within the nucleus at a fixed time. Note that the number of proteins tends to quickly jump up to a given amount, and then more slowly decay to either one or zero proteins. When only one protein is present for large periods of time, it is usually bound to the DNA, thereby repressing the gene (data not shown).

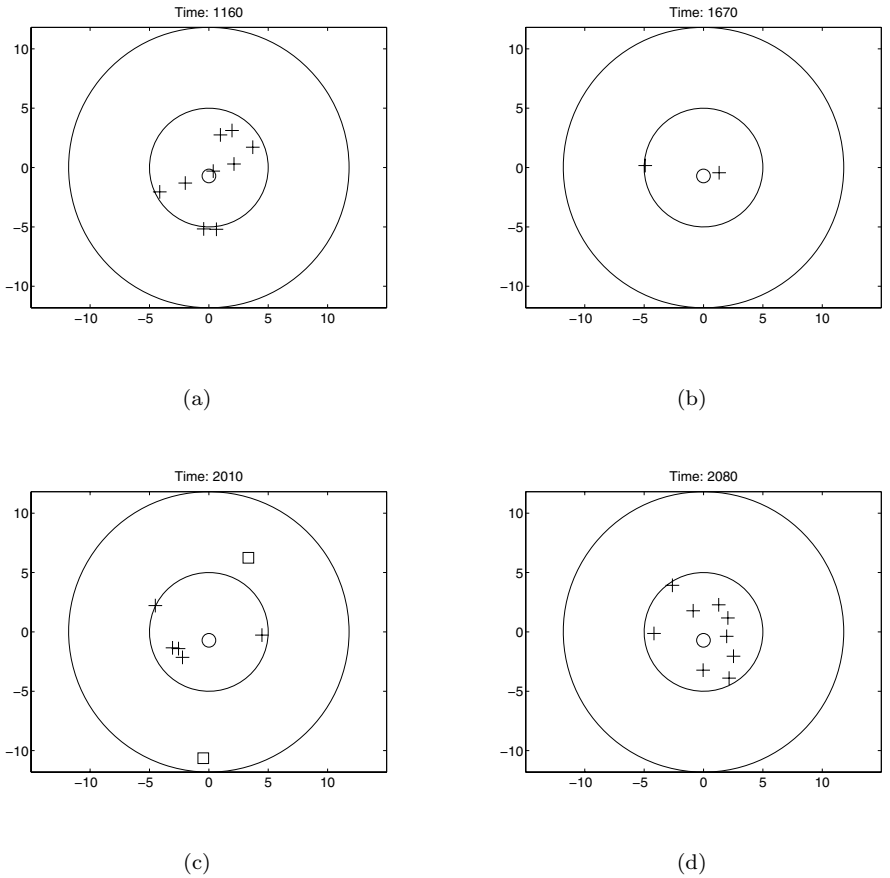


FIG. 7. *Evolution of one realization of the model from section* 4.2 *over a half hour. Symbols have the same codes as in Figure* 6.

5. Conclusions and future work. We have presented a method for calculating diffusive jump rates for the reaction-diffusion master equation within complex geometries. The method allows for multiple domains, and easily handles both jump and Neumann boundary conditions. The underlying discretization on which the method is based is found to converge between first and second order accuracy on both the Poisson equation and the diffusion equation. The discretization also conserves probability, and gives jump rates that satisfy the principle of detailed balance for diffusive motion and for passive boundary fluxes. With only one particle and no chemical reactions the method converges, as the mesh width goes to zero, to the solution to the Fokker–Planck equation for a classical Brownian particle. Further, when the system contains only first order reactions, the mean concentrations given by the method converge to the solution of the reaction-diffusion equations associated with the deterministic model of the chemical system. If the system contains bimolecular or higher order reactions, however, no such agreement with the solution of deterministic reaction-diffusion equations would be expected (except in the limit of large numbers of molecules), since averaging does not commute with multiplication. In such cases,

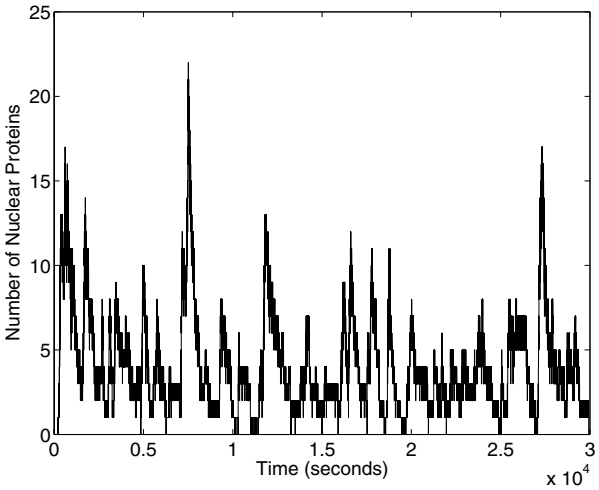


FIG. 8. *Total number of nuclear proteins in one realization of the model from section* 4.2. *Simulation occurred over* 30,000 *seconds, with data points at one second intervals.*

our method should give a more accurate representation of the stochastic dynamics of the spatially distributed chemical system.

As an illustrative application, we developed a model for eukaryotic transcription, nuclear export, translation, nuclear import, and gene regulation, and we simulated this model in a simplified two-dimensional representation of a eukaryotic cell. Spontaneous, but noisy, oscillations in nuclear protein levels were observed in this simple genetic network.

For future work, we plan to extend the current software implementation of the method to three dimensions, with the eventual goal of simulating cellular networks with realistic nuclear membrane and plasma membrane geometry. One interesting question is the interplay between reaction rates and diffusion rates in producing oscillatory nuclear protein levels. For different parameter regimes well-mixed stochastic, well-mixed deterministic, deterministic reaction-diffusion, and stochastic reaction-diffusion models could all show different qualitative behavior. We are also working on developing a more accurate biochemical model for the transcription and translation processes within eukaryotic cells.

Acknowledgments. The authors thank Ian G. Macara for helpful discussions regarding the model in [24]. We would also like to thank the referees for their careful reading of the paper and helpful comments and suggestions.

REFERENCES

[1] B. ALBERTS, A. JOHNSON, J. LEWIS, M. RAFF, K. ROBERTS, AND P. WALTER, *Molecular Biology of the Cell*, 4th ed., Garland Science, New York, 2002.

[2] S. S. ANDREWS AND D. BRAY, *Stochastic simulation of chemical reactions with spatial resolution and single molecule detail*, Physical Biology, 1 (2004), pp. 137–151.

[3] A. ARKIN AND H. H. MCADAMS, *Stochastic mechanisms in gene expression*, Proc. Natl. Acad. Sci. USA, 94 (1997), pp. 814–819.

[4] A. ARKIN, J. ROSS, AND H. H. MCADAMS, *Stochastic kinetic analysis of developmental pathway bifurcation in phage λ–infected escherichia coli cells*, Genetics, 149 (1998), pp. 1633–1648.

[5] F. Baras and M. M. Mansour, *Reaction-diffusion master equation: A comparison with microscopic simulations*, Phys. Rev. E, 54 (1996), pp. 6139–6147.
[6] W. J. Blake, M. Kaern, C. R. Cantor, and J. J. Collins, *Noise in eukaryotic gene expression*, Nature, 422 (2003), pp. 633–637.
[7] B. R. Cullen, *Nuclear mRNA export: Insights from virology*, TRENDS Biochem. Sci., 28 (2003), pp. 419–424.
[8] D. A. Drew, *A mathematical model for prokaryotic protein synthesis*, Bull. Math. Biol., 63 (2001), pp. 329–351.
[9] M. B. Elowitz, A. J. Levine, E. D. Siggia, and P. S. Swain, *Stochastic gene expression in a single cell*, Science, 297 (2002), pp. 1183–1186.
[10] D. Fusco, N. Accornero, B. Lavoie, S. Shenoy, J. Blanchard, R. Singer, and E. Bertrand, *Single mRNA molecules demonstrate probabilistic movement in living mammalian cells*, Curr. Biol., 13 (2003), pp. 161–167.
[11] C. W. Gardiner, K. J. McNeil, D. F. Walls, and I. S. Matheson, *Correlations in stochastic models of chemical reactions*, J. Statist. Phys., 14 (1976), p. 307–331.
[12] C. W. Gardiner, *Handbook of Stochastic Methods: For Physics, Chemistry, and the Natural Sciences*, 2nd ed., Springer Ser. Synergetics 13, Springer-Verlag, New York, 1996.
[13] F. Gibou, R. P. Fedkiw, L. Cheng, and M. Kang, *A second-order-accurate symmetric discretization of the Poisson equation on irregular domains*, J. Comput. Phys., 176 (2002), pp. 205–227.
[14] M. A. Gibson and J. Bruck, *Efficient exact stochastic simulation of chemical systems with many species and many channels*, J. Phys. Chem. A, 104 (2000), pp. 1876–1899.
[15] D. T. Gillespie, *Exact stochastic simulation of coupled chemical reactions*, J. Phys. Chem., 81 (1977), pp. 2340–2361.
[16] D. T. Gillespie, *A rigorous derivation of the chemical master equation*, Phys. A, 188 (1992), pp. 404–425.
[17] D. T. Gillespie, *The chemical Langevin equation*, J. Chem. Phys., 113 (2000), pp. 297–306.
[18] K. Ishida, *Stochastic model for bimolecular reaction*, J. Chem. Phys., 41 (1964), pp. 2472–2478.
[19] P. Jenny, S. B. Pope, M. Muradoglu, and D. A. Caughey, *A hybrid algorithm for the joint pdf equation of turbulent reactive flows*, J. Comput. Phys., 166 (2001), pp. 218–252.
[20] E. R. Kandel, *The molecular biology of memory storage: A dialogue between genes and synapses*, Science, 294 (2001), pp. 1030–1038.
[21] P. McCorquodale, P. Colella, and H. Johansen, *A cartesian grid embedded boundary method for the heat equation on irregular domains*, J. Comput. Phys., 173 (2001), pp. 620–635.
[22] M. Muradoglu, P. Jenny, S. B. Pope, and D. A. Caughey, *A consistent hybrid finite-volume/particle method for the pdf equations of turbulent reactive flows*, J. Comput. Phys., 154 (1999), pp. 342–370.
[23] E. M. Ozbudak, M. Thattai, I. Kurtser, A. D. Grossman, and A. van Oudenaarden, *Regulation of noise in the expression of a single gene*, Nat. Genet., 31 (2002), pp. 69–73.
[24] A. E. Smith, B. M. Slepchenko, J. C. Schaff, L. M. Loew, and I. G. Macara, *Systems analysis of ran transport*, Science, 295 (2002), pp. 488–491.
[25] N. G. Van Kampen, *Stochastic Processes in Physics and Chemistry*, North–Holland, Amsterdam, 2001.
[26] H. Wang, C. S. Peskin, and T. C. Elston, *A robust numerical algorithm for studying biomolecular transport processes*, J. Theoret. Biol., 221 (2003), pp. 491–511.

SIAM J. SCI. COMPUT.
Vol. 28, No. 1, pp. 75–101
© 2006 Society for Industrial and Applied Mathematics

ON CONSTRAINT PRESERVATION IN NUMERICAL SIMULATIONS OF YANG–MILLS EQUATIONS*

SNORRE H. CHRISTIANSEN† AND RAGNAR WINTHER†

Abstract. The problem of constraint preservation for discretizations of nonlinear PDEs is addressed in the example of the hyperbolic Yang–Mills equations in temporal gauge. These equations preserve a nonlinear divergence field analogous to the electric charge for Maxwell's equations. We introduce and discuss several discretizations of these equations on finite element spaces of Lie algebra valued differential forms. Numerical experiments indicate that simply restricting the variational formulation to the Galerkin spaces yields substantial drift in the charge, contrary to the linear Maxwell case. We then propose a fully discrete method constrained with Lagrange multipliers, for which we prove discrete charge conservation and observe excellent energy conservation.

Key words. constraint preservation, finite elements, Yang–Mills equations

AMS subject classifications. 65M60, 65P10, 35L70

DOI. 10.1137/040616887

1. Introduction.

Many physical phenomena can be accurately described in terms of fields subject to a constrained evolution equation. For instance, Maxwell's equations describe an electric field in the vacuum as a divergence-free vector field satisfying a wave equation. The Navier–Stokes equations describe the velocity field of an incompressible fluid as a divergence-free vector field subject to a nonlinear evolution equation, involving the pressure which is a Lagrange multiplier for the divergence constraint. For both of these examples, discretization is today relatively well understood, leading to powerful simulation tools.

However, for other examples the situation is less satisfactory. In particular, in certain gauges, Einstein's equations of general relativity (GR) reduce to a hyperbolic evolution equation for a Riemannian metric on a space-slice, subject to nonlinear differential constraints (see, e.g., Wald [39]). The construction of good discretizations of these equations is an active research topic where much remains to be done. Lehner [26] mentions in particular the need to develop tools to help interpret data from gravitational wave detection experiments in the new field of "gravitational wave astronomy." It has been suggested that the problems encountered for simulating Einstein's equations are linked to difficulties for discretizations to preserve nonlinear differential constraints, since hyperbolicity might be lost outside the constraint manifold.

To gain insight into the generic problem of constraint preservation for nonlinear PDEs, we study in this paper the Yang–Mills equations in their hyperbolic form. These are evolution PDEs preserving a nonlinear constraint on the Cauchy data (initial values). However, in contrast to the initial value formulation of Einstein's equations, the Yang–Mills system remains stable even if the constraint is violated. The Yang–Mills equations appear relatively ripe for numerical analysis and could therefore serve as a stepping stone toward the successful simulation of more complicated equations. There are several reasons for this belief. First, concerning space discretization, when the Lie algebra is trivial, Yang–Mills equations reduce to Maxwell's equations

*Received by the editors October 13, 2004; accepted for publication (in revised form) August 9, 2005; published electronically March 3, 2006.
http://www.siam.org/journals/sisc/28-1/61688.html
†Centre of Mathematics for Applications, Department of Mathematics, University of Oslo, P.O. Box 1053 Blindern, NO-0316 Oslo, Norway (snorrec@math.uio.no, ragnar.winther@cma.uio.no).

and the Galerkin discretization of these on edge element spaces is now relatively well understood (see section 3.1 for references). Second, concerning time discretization, recent years have seen a deeper understanding of geometric properties of time-stepping methods (see also section 3.1). These two points are of course not completely distinct as they can be seen as facets of space-time variational principles. A third reason for optimism is that contrary to many other nonlinear equations one has analysis tools for studying relatively rough solutions to Yang–Mills equations, perhaps enabling norm estimates on the nonsmooth functions that appear in finite element discretizations (see section 2.2).

Briefly put, Yang–Mills equations are Euler–Lagrange equations for a time dependent gauge potential, which can be deduced from a Lagrangian which is invariant under a certain group of transformations called gauge transformations. This invariance provides a conservation property similar to (electric) charge conservation for Maxwell's equations. However, the natural discretization space for gauge potentials appears not to be invariant under the natural candidate for a subgroup of gauge transformations, contrary to the linear case of Maxwell's equations. We study the effect of this noninvariance on charge conservation and observe numerically that indeed, by discretizing , we lose , -conservation of some significant quantities (even before time is discretized). We then discuss ways to impose charge conservation with Lagrange multipliers. Once a satisfactory way of doing this has been found, we turn to the time discretization.

The paper is organized as follows. A first part is devoted to introducing the equations and their setting. In section 2.1 we define Lie algebra valued differential forms and describe the basic operations on them that we will need. Then, in section 2.2, we introduce the Yang–Mills equations from a variational principle and relate these equations to Maxwell's equations. In section 2.3 we prove the charge conservation property we are interested in. In a second part we turn to discretizations and study first, in section 3.1, the (semi)discretizations obtained by restricting the Lagrangian to a finite dimensional space of gauge potentials. We show numerical results indicating that charge conservation is bad even though other parameters appear to be good. Motivated by this failure, we then, in section 3.2, discuss ways to impose charge conservation actively and show excellent numerical results for one of them. We have also included some remarks pertaining to the implementation of the proposed schemes in section 3.3. Finally, we conclude with some indications of possible directions in which to pursue the investigations.

2. Yang–Mills equations.

2.1. Lie algebra valued differential forms. In this section we introduce notation and collect some facts about Lie algebra valued differential forms. Short introductions to this topic can be found in Taylor [37, appendix C] and Donaldson and Kronheimer [13, chapter 2]. While we allow arbitrary base-manifolds, we restrict our attention to what are called trivial principal bundles, so in fact we shall not need the definition of a principal bundle at all. This leads to many simplifications in the presentation. Some of the global topological properties of the theory are thus lost, but we hope to convince the reader that even in this simplified setting many interesting problems remain. The discretizations we will introduce later will allow for general base-manifolds but are also restricted to trivial principal bundles.

We will define many objects in terms of tensor products. This is convenient not only for the exposition but also for implementing algorithms since tensor products of operators correspond to Kronecker products of matrices, providing a systematic way of

assembling complex objects from simpler constituents. A handy introduction to tensor products of vector spaces can be found in Ryan [33, chapter 1]. We will come back to issues of implementation later. Many tensor products are canonically isomorphic to spaces of (multi)linear maps. In particular, if E denotes a finite dimensional vector space, the space of p-linear forms on E can be identified with the p-fold tensor product $(\otimes^p E^\star)$. If $u : E^p \to \mathbb{R}$ is a p-linear form and $v : E^q \to \mathbb{R}$ is a q-linear form, their tensor product $u \otimes v$ is the $(p+q)$-linear form $E^{p+q} \to \mathbb{R}$ defined by

$$(u \otimes v)(\xi_1, \ldots, \xi_{p+q}) = u(\xi_1, \ldots, \xi_p) v(\xi_{p+1}, \ldots, \xi_{p+q}). \tag{2.1}$$

The antisymmetrization operator on the space of p-linear forms is the projector α_p defined by

$$(\alpha_p u)(\xi_1, \ldots, \xi_p) = \frac{1}{p!} \sum_{\sigma \in \mathfrak{S}(p)} \epsilon(\sigma) u(\xi_{\sigma(1)}, \ldots, \xi_{\sigma(p)}), \tag{2.2}$$

where $\mathfrak{S}(p)$ denotes the group of permutations of the set $\{1,\ 2, \ldots, p\}$ and ϵ is the signature morphism. Antisymmetric p-linear forms are also said to be alternating.

We will use freely the language of (real-valued) differential forms, as exposed, for instance, in Lang [25], but recall some basic facts. Let M be a finite dimensional real and smooth manifold. The tangent space of M at a point x is denoted $\mathrm{T}_x M$. For any integer p, a p-form on M is an object u which to any point $x \in M$ associates a p-linear alternating map $u_x : (\mathrm{T}_x M)^p \to \mathbb{R}$. We denote by Ω^p the space of (smooth) p-forms on M. When $p > \dim M$, Ω^p is reduced to 0. Ω^0 is the space of real-valued functions on M. The direct sum of the spaces Ω^p is denoted Ω. The wedge product is a bilinear associative map $\wedge : \Omega \times \Omega \to \Omega$ such that for each integer p and q

$$\wedge : \Omega^p \times \Omega^q \to \Omega^{p+q}. \tag{2.3}$$

For each $u \in \Omega^p$ and $v \in \Omega^q$, $u \wedge v$ is defined at $x \in M$ by

$$(u \wedge v)_x = \frac{(p+q)!}{p!\ q!} \alpha_p(u_x \otimes v_x). \tag{2.4}$$

We then have

$$v \wedge u = (-1)^{pq} u \wedge v. \tag{2.5}$$

The exterior derivative is a linear map $\mathrm{d} : \Omega \to \Omega$ such that for each integer p

$$\mathrm{d} : \Omega^p \to \Omega^{p+1}. \tag{2.6}$$

In charts it can be constructed by antisymmetrizing the full differential. It is also characterized by the following two properties: If $u \in \Omega^0$, then for each $x \in M$, $(\mathrm{d}u)_x = \mathrm{D}u(x) : \mathrm{T}_x M \to \mathbb{R}$ is the usual differential of u at x (also called derivative or tangent map). Moreover, for each $u \in \Omega^p$ and $v \in \Omega^q$ we have

$$\mathrm{d}(u \wedge v) = (\mathrm{d}u) \wedge v + (-1)^p u \wedge \mathrm{d}v. \tag{2.7}$$

Essentially because the full second order differential (say, in some chart) is symmetric (on $\mathbb{R}^n$ the partial derivatives ∂_i commute: $\partial_i \partial_j = \partial_j \partial_i$), the operator d satisfies

$\mathrm{d}^2 = 0$. This makes the sequence of spaces (Ω^p) what is called a complex (see, e.g., Gelfand and Manin [17, p. 8]):

$$0 \xrightarrow{\mathrm{d}} \Omega^0 \xrightarrow{\mathrm{d}} \Omega^1 \xrightarrow{\mathrm{d}} \Omega^2 \xrightarrow{\mathrm{d}} \cdots \tag{2.8}$$

We now fix a finite dimensional real Lie algebra $\mathfrak{g}$, that is, a finite dimensional real vector space equipped with a bilinear map $[\cdot,\cdot] : \mathfrak{g} \times \mathfrak{g} \to \mathfrak{g}$ called a Lie bracket, which is antisymmetric and satisfies the Jacobi identity: for all $a, b, c \in \mathfrak{g}$ we have

$$[a, b] + [b, a] = 0, \tag{2.9}$$

$$[a, [b, c]] + [b, [c, a]] + [c, [a, b]] = 0. \tag{2.10}$$

Under the condition that the other properties of the bracket are kept, the Jacobi identity is equivalent to the property that, for all $a, b, c \in \mathfrak{g}$,

$$[a, [b, c]] = [[a, b], c] + [b, [a, c]], \tag{2.11}$$

which expresses that for each $a \in \mathfrak{g}$, $[a, \cdot]$ acts as a derivation on $[\cdot,\cdot]$.

By a $\mathfrak{g}$-valued p-form (on M) we mean an object A which to any point $x \in M$ associates a p-linear alternating map $A_x : (\mathrm{T}_x M)^p \to \mathfrak{g}$. The space of $\mathfrak{g}$-valued p-forms is canonically isomorphic to the tensor product $\Omega^p \otimes \mathfrak{g}$. Explicitly, if $u \in \Omega^p$ and $a \in \mathfrak{g}$, $u \otimes a$ can be identified with the $\mathfrak{g}$-valued p-form defined at any $x \in M$ by

$$(\xi_1, \ldots, \xi_p) \mapsto u_x(\xi_1, \ldots, \xi_p)a, \tag{2.12}$$

and any $\mathfrak{g}$-valued p-form is a linear combination of (at most $\dim \mathfrak{g}$) such forms. The bracket of $\mathfrak{g}$-valued forms is a bilinear operator $(\Omega \otimes \mathfrak{g}) \times (\Omega \otimes \mathfrak{g}) \to \Omega \otimes \mathfrak{g}$ giving restrictions $(\Omega^p \otimes \mathfrak{g}) \times (\Omega^q \otimes \mathfrak{g}) \to \Omega^{p+q} \otimes \mathfrak{g}$. It is denoted $[\cdot,\cdot]$ and uniquely characterized by the property that for all $u \in \Omega^p$, $a \in \mathfrak{g}$, $v \in \Omega^q$, and $b \in \mathfrak{g}$

$$[u \otimes a, v \otimes b] = (u \wedge v) \otimes [a, b]. \tag{2.13}$$

In other words, we consider the tensor product of the standard wedge $\wedge$ defined on (real-valued) differential forms on M, with the Lie bracket $[\cdot,\cdot]$ on $\mathfrak{g}$. This gives $\Omega \otimes \mathfrak{g}$ the structure of a graded Lie algebra. That is, if we have $A \in \Omega^p \otimes \mathfrak{g}$, $B \in \Omega^q \otimes \mathfrak{g}$, and $C \in \Omega^r \otimes \mathfrak{g}$, then we deduce from (2.5), (2.9), and (2.11)

$$0 = [A, B] + (-1)^{pq}[B, A], \tag{2.14}$$

$$[A, [B, C]] = [[A, B], C] + (-1)^{pq}[B, [A, C]]. \tag{2.15}$$

The exterior derivative is extended to an operator $\mathrm{d} : \Omega \otimes \mathfrak{g} \to \Omega \otimes \mathfrak{g}$ inducing maps $\Omega^p \otimes \mathfrak{g} \to \Omega^{p+1} \otimes \mathfrak{g}$. It is uniquely characterized by the property that

$$\forall u \in \Omega^p,\ \forall a \in \mathfrak{g} \quad \mathrm{d}(u \otimes a) = (\mathrm{d}u) \otimes a. \tag{2.16}$$

If A is a [illegible], i.e., an element of $\Omega^1 \otimes \mathfrak{g}$, we associate with it a [illegible] denoted d_A defined by

$$\mathrm{d}_A : \begin{cases} \Omega^q \otimes \mathfrak{g} & \to \quad \Omega^{q+1} \otimes \mathfrak{g}, \\ B & \mapsto \quad \mathrm{d}B + [A, B]. \end{cases} \tag{2.17}$$

It is a derivation in the sense that if $B \in \Omega^q \otimes \mathfrak{g}$ and $C \in \Omega^r \otimes \mathfrak{g}$ we have (compare with (2.7) and (2.11))

$$\mathrm{d}_A[B, C] = [\mathrm{d}_A B, C] + (-1)^q [B, \mathrm{d}_A C]. \tag{2.18}$$

The [illegible] of a gauge potential A is the element $\mathcal{F}(A)$ of $\Omega^2 \otimes \mathfrak{g}$ defined by [illegible] as

$$\mathcal{F}(A) = \mathrm{d}A + (1/2)[A, A]. \tag{2.19}$$

This definition is motivated by the property that for all gauge potentials A and all $B \in \Omega^q \otimes \mathfrak{g}$ we have

$$\mathrm{d}_A \mathrm{d}_A B = [\mathcal{F}(A), B]. \tag{2.20}$$

We remark that the bilinear map $[\cdot, \cdot] : (\Omega^1 \otimes \mathfrak{g})^2 \to \Omega^2 \otimes \mathfrak{g}$ is symmetric. Therefore the differential $\mathrm{D}\mathcal{F}$ of the curvature map $\mathcal{F} : \Omega^1 \otimes \mathfrak{g} \to \Omega^2 \otimes \mathfrak{g}$ satisfies for all $A, A' \in \Omega^1 \otimes \mathfrak{g}$

$$\mathrm{D}\mathcal{F}(A)A' = \mathrm{d}_A A'. \tag{2.21}$$

We also have the [illegible]:

$$\mathrm{d}_A \mathcal{F}(A) = 0. \tag{2.22}$$

We suppose that $\mathfrak{g}$ is the Lie algebra associated with a Lie group G. For generalities on Lie groups we refer to, e.g., Duistermaat and Kolk [14]. For notational purposes we suppose that G is a matrix Lie group. Given a function $Q : M \to G$, the associated [illegible] $\mathcal{G}_Q$ is the map which transforms a gauge potential A into the gauge potential $A' = \mathcal{G}_Q(A)$ defined by

$$A' = QAQ^{-1} - (\mathrm{D}Q)Q^{-1}. \tag{2.23}$$

This notation calls for some comments. First, for a given $q \in G$, one shows that $\mathrm{Ad}(q) : a \mapsto qaq^{-1}$ maps $\mathfrak{g}$ into $\mathfrak{g}$ linearly so that the first term on the right-hand side of (2.23), interpreted pointwise, provides a gauge potential. For generalities about the adjoint representation $\mathrm{Ad} : G \to \mathrm{End}(\mathfrak{g})$ see [14, p. 2]. Second, also for a given $q \in G$, (right and left) multiplication by q^{-1} maps the tangent space $\mathrm{T}_q G$ of G at q linearly into $\mathfrak{g}$. And for any function $Q : M \to G$ the differential of Q gives rise to linear maps $\mathrm{D}Q(x) : \mathrm{T}_x M \to \mathrm{T}_{Q(x)} G$. Thus the second term on the right-hand side of (2.23), interpreted pointwise, also provides a gauge potential. Combining these observations shows that A' defined as above is indeed a gauge potential.

Gauge transformations behave nicely with covariant derivatives and curvature. With $A' = \mathcal{G}_Q(A)$ as before, we have

$$\mathrm{d}_{A'}(QBQ^{-1}) = Q(\mathrm{d}_A B)Q^{-1} \tag{2.24}$$

and

$$\mathcal{F}(A') = Q\mathcal{F}(A)Q^{-1}. \tag{2.25}$$

Moreover, $\mathcal{G}$ determines a group action in the sense that

$$\mathcal{G}_{Q'} \circ \mathcal{G}_Q = \mathcal{G}_{Q'Q}, \tag{2.26}$$

where on the right-hand side we use pointwise multiplication to define the product $Q'Q$.

We suppose that a Riemannian metric is given on M. That is, for any $x \in M$ we have a Euclidean scalar product on $\mathrm{T}_x M$, depending smoothly on x. It enables us to construct L^2-products on Ω as follows. First, for any Euclidean space E (we have in mind the tangent spaces $\mathrm{T}_x M$), whose scalar product we denote by $(\cdot|\cdot)$, we first obtain a Euclidean structure on the dual $E^\star$, whose scalar product we also denote by $(\cdot|\cdot)$, by requiring that $u \mapsto (u|\cdot)$ is an isometry $E \to E^\star$. Next, on the space of p-linear forms on E^p (not necessarily alternating), there is a unique scalar product such that, for any $u_1, \ldots, u_p$, $v_1, \ldots, v_p$ in $E^\star$,

$$(u_1 \otimes \cdots \otimes u_p | v_1 \otimes \cdots \otimes v_p) = (u_1|v_1) \cdots (u_p|v_p). \tag{2.27}$$

This scalar product can be restricted to the subspace of alternating p-linear forms on E^p. This construction provides for each $x \in M$ a Euclidean structure on the space of p-linear forms on $(\mathrm{T}_x M)^p$, denoted $(\cdot|\cdot)_x$. We then obtain a scalar product $\langle \cdot, \cdot \rangle$ on Ω defined by

$$\langle u, v \rangle = \int_M (u_x | v_x)_x \mathrm{d}x, \tag{2.28}$$

where we integrate using the measure on M provided by the Riemannian metric.

We suppose furthermore that the Lie group G is compact. Then on $\mathfrak{g}$ there is a scalar product denoted $(\cdot|\cdot)$ which is compatible in the sense that for all $q \in G$ the map $\mathrm{Ad}(q)$ is an isometry of $\mathfrak{g}$ (see, e.g., [14, p. 135]). This property implies that

$$\forall a, b, c \in \mathfrak{g} \quad ([a, b]|c) + (b|[a, c]) = 0. \tag{2.29}$$

With the help of this scalar product and a Riemannian metric on M we can construct L^2-products on $\Omega \otimes \mathfrak{g}$, as well as L^q-spaces, by slightly extending the preceding method. Given as before a Euclidean space E, one first shows that there is a unique scalar product on $(\otimes^p E^\star) \otimes \mathfrak{g}$ such that for $u, v \in \otimes^p E^\star$, and $a, b \in \mathfrak{g}$

$$(u \otimes a | v \otimes b) = (u|v)(a|b). \tag{2.30}$$

This provides scalar products on the spaces $(\otimes^p (\mathrm{T}_x M)^\star) \otimes \mathfrak{g}$ as before, as well as a scalar product on $\Omega \otimes \mathfrak{g}$ denoted $\langle \cdot, \cdot \rangle$. We also obtain L^q-norms defined by

$$\|A\|_{\mathrm{L}^q} = \left(\int_M |A_x|_x^q \mathrm{d}x \right)^{1/q}, \tag{2.31}$$

where $|\cdot|_x$ is the Euclidean norm on $(\otimes^p (\mathrm{T}_x M)^\star) \otimes \mathfrak{g}$ defined by the constructed scalar product. At least for our purposes these norms behave like standard L^q-norms on scalar functions, for which we refer to Folland [16]. We shall allow ourselves to speak about L^q-spaces of real and Lie algebra valued forms without burdening the notation with the precision of what kind of forms we will be considering at each time.

From (2.29) we deduce that for any $A \in \Omega^0 \otimes \mathfrak{g}$ and any $B, C \in \Omega^p \otimes \mathfrak{g}$ we have

$$\langle [A, B], C \rangle + \langle B, [A, C] \rangle = 0. \tag{2.32}$$

In particular, the case $B = C$ yields

$$\langle [A, B], B \rangle = 0. \tag{2.33}$$

2.2. Yang–Mills Lagrangian and equations. In accordance with notions introduced in the previous section we consider the following setting and notation: M is a compact Riemannian manifold (without boundary) and G is a compact Lie group with associated Lie algebra $\mathfrak{g}$. A compatible scalar product on $\mathfrak{g}$ is chosen. The L^2-duality on $\Omega \otimes \mathfrak{g}$ with respect to the Riemannian metric on M and the compatible scalar product on $\mathfrak{g}$ is denoted $\langle \cdot, \cdot \rangle$. The L^2-norm is denoted $\| \cdot \|$.

The hyperbolic Yang–Mills equation, in the temporal gauge, is an equation for gauge potentials on M that can be derived from the Lagrangian $\mathcal{L}$ defined on $(\Omega^1 \otimes \mathfrak{g})^2$ by

$$\mathcal{L}(A, A') = (1/2)\|A'\|^2 - (1/2)\|\mathcal{F}(A)\|^2. \tag{2.34}$$

We look for time dependent gauge potentials $A : t \mapsto A(t) \in \Omega^1 \otimes \mathfrak{g}$ satisfying the stationary action principle. That is, we look for trajectories A which, for any $t, t' \in \mathbb{R}$, are critical points of the action integral

$$B \mapsto \int_t^{t'} \mathcal{L}(B(s), \dot{B}(s)) \mathrm{d}s, \tag{2.35}$$

among trajectories B such that $B(t) = A(t)$ and $B(t') = A(t')$.

If we ignore for the time being some questions about the regularity of solutions and continuity properties of operators, the associated Euler–Lagrange equation for a time dependent gauge potential $A : t \mapsto A(t)$ is (at any time t)

$$\forall A' \in \Omega^1 \otimes \mathfrak{g} \quad \langle \ddot{A}, A' \rangle = -\langle \mathcal{F}(A), \mathrm{D}\mathcal{F}(A)A' \rangle, \tag{2.36}$$

$$= -\langle \mathcal{F}(A), \mathrm{d}_A A' \rangle. \tag{2.37}$$

The so-called strong formulation of this equation is obtained by introducing $\mathrm{d}_A^\star$, the formal adjoint of d_A, with respect to L^2-duality on M. Smooth solutions to (2.37) are those satisfying

$$\ddot{A} = -\mathrm{d}_A^\star \mathcal{F}(A). \tag{2.38}$$

To us the strong formulation is mainly a convenient notation. Discretizations are guided by the weak formulation or indeed the stationary action principle.

Various first order formulations are useful. With a time dependent gauge potential $A : t \mapsto A(t)$ we associate time dependent fields $E : t \mapsto E(t) \in \Omega^1 \otimes \mathfrak{g}$ and $B : t \mapsto B(t) \in \Omega^2 \otimes \mathfrak{g}$ defined by

$$E(t) = -\dot{A}(t), \tag{2.39}$$

$$B(t) = \quad \mathcal{F}(A(t)). \tag{2.40}$$

We remark that these two equations imply the following evolution equation for B:

$$\dot{B}(t) = \mathrm{D}\mathcal{F}(A(t))\dot{A}(t) = -\mathrm{d}_A E. \tag{2.41}$$

So we get (in strong form)

$$\dot{A} = -E, \tag{2.42}$$

$$\dot{E} = \quad \mathrm{d}_A^\star B, \tag{2.43}$$

$$\dot{B} = -\mathrm{d}_A E. \tag{2.44}$$

Conversely, if (A, E, B) satisfies the above system, and the initial condition $B(0) = \mathcal{F}(A(0))$, then for all times t, $B(t) = \mathcal{F}(A(t))$ and A satisfies the Euler–Lagrange equation (2.37).

We will also use the following first order formulation involving only the fields A and E:

$$\dot{A} = -E, \tag{2.45}$$

$$\dot{E} = \quad \mathrm{d}_A^\star \mathcal{F}(A). \tag{2.46}$$

It differs from the standard Hamiltonian reformulation of Euler–Lagrange equations by the sign appearing in (2.45). The choice of signs is motivated by the conventions used in electromagnetics, at which we shall arrive shortly.

We have stated most of the above results as if every field was smooth. For the purposes of numerical analysis it is useful however to keep in mind the following existence and uniqueness results for nonsmooth solutions of the Yang–Mills equations. Perhaps most importantly to us, Klainerman and Machedon [24] have proved global well-posedness in the Sobolev space H^1_{loc} for initial data with finite energy. An essential ingredient here is a compactness result of Uhlenbeck [38]. Tao [36] has obtained local results for norms weaker than H^1_{loc}. Earlier results include existence and uniqueness for high-order Sobolev norms by Segal [34]; see also Eardley and Moncrief [15].

Actually even the H^1_{loc} framework is somewhat inadequate for numerical analysis. It appears mainly because we have gauge potentials A, say, in L^2, with exterior derivative $\mathrm{d}A$ in L^2, for which we have , , , control over $\mathrm{d}^\star A$. Experience from Maxwell's equations (to which we devote the next paragraph) indicates that we should consider discretization spaces consisting of potentials $A \in \mathrm{L}^2$ such that $\mathrm{d}A \in \mathrm{L}^2$, but for which $\mathrm{d}^\star A$ is not necessarily integrable. This term should rather be controlled by the good behavior of $\langle A, \mathrm{d}P' \rangle$ for a large enough space of "test-functions" $P' \in \mathrm{L}^2$ satisfying $\mathrm{d}P' \in \mathrm{L}^2$.

The numerical analysis is as of today incomplete and we do not claim to have found the right framework for carrying it out.

Many of the results of this paper are motivated by analogies with Maxwell's equations and we now make this link explicit. In the context of electromagnetics, M is three-dimensional, E is the electric field, and B is the magnetic field. We choose G to be the group

$$\mathbb{U} = \{u \in \mathbb{C} \ : \ |u| = 1\}, \tag{2.47}$$

so that $\mathfrak{g}$ can be identified with $i\mathbb{R} \subset \mathbb{C}$. Since $\mathbb{U}$ is commutative the Lie bracket is trivial, leading to many simplifications. From (2.43), (2.44) we obtain the familiar (vacuum) Maxwell equations for E and B, given here in vector notation:

$$\dot{E} = \quad \operatorname{curl} B, \tag{2.48}$$

$$\dot{B} = -\operatorname{curl} E. \tag{2.49}$$

Equations (2.39), (2.40) express that the magnetic potential A is a solution of (also in vector notation)

$$\dot{A} = -E, \tag{2.50}$$

$$\operatorname{curl} A = \quad B. \tag{2.51}$$

The strong form of the Euler–Lagrange equation (2.38) for A is

$$\ddot{A} = -\operatorname{curl}\operatorname{curl} A. \tag{2.52}$$

All fields here are $\mathbb{R}$-proportional to the complex number i, so it is customary to cancel it out. Other choices of magnetic potentials are often used but in this paper we restrict our attention to the above choice of gauge, called temporal gauge (because A can be identified with a 1-form on time-space $\mathbb{R} \times M$ having no component along the time axis).

Maxwell's equations (2.48), (2.49) preserve the following constraints on the initial data:

$$\operatorname{div} E = 0, \tag{2.53}$$

$$\operatorname{div} B = 0. \tag{2.54}$$

This follows immediately from the identity $\operatorname{div} \operatorname{curl} = 0$. The Yang–Mills equations have similar constraints as we shall see in the next section.

In this presentation we have of course reversed the historical order, since Yang and Mills obtained their equations in an effort to generalize Maxwell's equations to the case where G is a noncommutative Lie group such as SU(2).

2.3. Conservation properties. For Maxwell's equations (2.48), (2.49) the two divergence constraints (2.53), (2.54) appear as completely symmetric. But once a magnetic potential is introduced as in (2.50), (2.51), these two constraints acquire rather different meanings. This will become even more apparent for the Yang–Mills equations, which have similar constraints. The generalization to Yang–Mills equations of the divergence constraint (2.54) on the magnetic field is the Bianchi identity (2.22). As we shall see, it is not a problem to handle numerically, at least for the discretizations we have chosen. On the other hand, the following identity, which is analogous to charge conservation (2.53), is numerically more challenging. We should have the constraint

$$\forall P' \in \Omega^0 \otimes \mathfrak{g} \quad \langle \dot{A}, \mathrm{d}_A P' \rangle = 0. \tag{2.55}$$

In general (not necessarily temporal) gauge, this equation is actually one of the Yang–Mills equations. In our setting, however, it appears as a constraint which is preserved by the flow , ′ which should be satisfied initially. We now check this in two (related) ways.

Pick $P' \in \Omega^0 \otimes \mathfrak{g}$, and regard it as a constant function of time. Putting $A' = \mathrm{d}_A P'$ in (2.37) we obtain (from (2.20) and (2.33))

$$\langle \ddot{A}, \mathrm{d}_A P' \rangle = -\langle \mathcal{F}(A), [\mathcal{F}(A), P'] \rangle = 0. \tag{2.56}$$

We also remark that

$$\langle \dot{A}, [\dot{A}, P'] \rangle = 0. \tag{2.57}$$

It follows that

$$\langle \dot{A}, \mathrm{d}_A P' \rangle^{\cdot} = \langle \ddot{A}, \mathrm{d}_A P' \rangle + \langle \dot{A}, [\dot{A}, P'] \rangle = 0. \tag{2.58}$$

In analogy with Maxwell's equations we will refer to this conservation property as charge conservation. As already mentioned, condition (2.55) should also be satisfied initially. Thus, with $\dot{A} = -E$, the couple $(A(t), E(t))$ should evolve on the (in general infinite dimensional, nonlinear, and perhaps singular) "manifold" with equation

$$\forall P' \in \Omega^0 \otimes \mathfrak{g} \quad \langle E, \mathrm{d}_A P' \rangle = 0. \tag{2.59}$$

We now show how this property can be obtained using Noether's theorem. First fix a map $Q : M \to G$, and consider the associated gauge transformation $\mathcal{G}_Q : \Omega^1 \otimes \mathfrak{g} \to \Omega^1 \otimes \mathfrak{g}$. For all gauge potentials $A, A' \in \Omega^1 \otimes \mathfrak{g}$ we have (using (2.25) and the compatibility of the scalar product)

$$\mathcal{L}(\mathcal{G}_Q(A), \mathrm{D}\mathcal{G}_Q(A)A') = (1/2)\|QA'Q^{-1}\|^2 - (1/2)\|Q\mathcal{F}(A)Q^{-1}\|^2 \tag{2.60}$$

$$= \mathcal{L}(A, A'). \tag{2.61}$$

In other words the Lagrangian $\mathcal{L}$ is invariant under gauge transformations $\mathcal{G}_Q$ associated with maps $Q : M \to G$. On the other hand, if we fix a gauge potential A and differentiate the map $Q \mapsto \mathcal{G}_Q(A)$ at the constant map $I : M \to G$ equal to the unit of G, we get the following linear map:

$$\Gamma(A) : \left\{ \begin{array}{rcl} \Omega^0 \otimes \mathfrak{g} & \to & \Omega^1 \otimes \mathfrak{g}, \\ P' & \mapsto & -\mathrm{d}_A P'. \end{array} \right. \tag{2.62}$$

Therefore, by Noether's theorem (see, e.g., Marsden and West [28, p. 369]), for each $P' \in \Omega^0 \otimes \mathfrak{g}$ the following function on $(\Omega^1 \otimes \mathfrak{g})^2$ is constant on trajectories $t \mapsto (A(t), \dot{A}(t))$:

$$(A, A') \mapsto \mathrm{D}_2\mathcal{L}(A, A')\Gamma(A)P' = -\langle A', \mathrm{d}_A P' \rangle. \tag{2.63}$$

This provides an interpretation of the charge conservation property.

3. Discretization.

3.1. Unconstrained schemes. For the discussion of numerical schemes we suppose that the spaces Ω^p do not consist solely of smooth fields but have been completed with respect to some norm of the form

$$u \mapsto \|u\|_{\mathrm{L}^q} + \|\mathrm{d}u\|_{\mathrm{L}^q}, \tag{3.1}$$

with $q < +\infty$. Then the exterior derivative extends to a continuous operator $\mathrm{d} : \Omega^p \to \Omega^{p+1}$. The exponent q should be chosen such that the brackets of forms yield sufficiently integrable forms.

For each integer p let X_h^p denote a finite dimensional subspace of Ω^p. We require that $\mathrm{d}X_h^p \subset X_h^{p+1}$. This provides a commuting diagram

$$\begin{array}{ccccccccc} 0 & \to & \Omega^0 & \to & \Omega^1 & \to & \Omega^2 & \to & \cdots \\ \uparrow & & \uparrow & & \uparrow & & \uparrow & & \\ 0 & \to & X_h^0 & \to & X_h^1 & \to & X_h^2 & \to & \cdots, \end{array} \tag{3.2}$$

where the vertical arrows are inclusion maps and the horizontal ones are instances of the exterior derivative. The following discussion of discretizations makes the most sense when X_h^{p+1} is small among the spaces containing $\mathrm{d}X_h^p$ (the precise statement would be that the vertical arrows in diagram (3.2) should induce isomorphisms in cohomology; see Gelfand and Manin [17]). Commuting diagrams are increasingly used to interpret or guide developments in numerical analysis; see Arnold [1].

The index h is there to remind us that we are in fact interested in sequences of spaces indexed by some parameter h and convergence results with respect to h.

For the numerical experiments we equip the domain M with a simplicial mesh $\mathcal{T}_h$ and take (X_h^p) to be the standard lowest-order finite element complex, where X_h^0 consists of continuous and piecewise affine functions, X_h^1 is a space of 1-forms called (lowest-order) Nédélec's edge elements (of the first family), and X_h^2 is a space of 2-forms referred to as Raviart–Thomas face elements. Similar constructions have been used in physics (see Sorkin [35]) and can be traced back at least to Weil [40] who used them for algebraic/topological purposes (see also Whitney [41]). The interpretation of mixed finite element methods in terms of such differential forms is due to Bossavit [6]. In this choice of spaces the dimension of X_h^p is exactly the number of p-dimensional simplexes in the mesh. The parameter h denotes the maximal diameter of the simplexes of $\mathcal{T}_h$.

The finite element point of view provides other choices of spaces as well—for instance, spaces built with polynomials of any degree; see Nédélec [30], [31]. For surveys on these spaces in the context of electromagnetics, we refer the reader to Hiptmair [21] and Monk [29]. So-called *hp* versions of these finite element spaces are also useful for simulating Maxwell's equations, and we refer the reader to Demkowicz [12] for an overview on this topic. For generalities about mixed finite element methods, see [8], [32], [18].

Given such a finite dimensional complex we will use the spaces $X_h^p \otimes \mathfrak{g}$ as discretization spaces for Lie algebra valued differential forms.

We consider time dependent gauge potentials of the form $A : t \mapsto A(t) \in X_h^1 \otimes \mathfrak{g}$. Restricting the Lagrangian (2.34) to $(X_h^1 \otimes \mathfrak{g})^2$, the corresponding stationary action principle yields the Euler–Lagrange equation

$$\forall A' \in X_h^1 \otimes \mathfrak{g} \quad \langle \ddot{A}, A' \rangle = -\langle \mathcal{F}(A), \mathrm{d}_A A' \rangle. \tag{3.3}$$

An equivalent first order formulation is the following system for time dependent A, E in $X_h^1 \otimes \mathfrak{g}$:

$$\dot{A} = -E, \tag{3.4}$$

$$\forall E' \in X_h^1 \otimes \mathfrak{g} \quad \langle \dot{E}, E' \rangle = \langle \mathcal{F}(A), \mathrm{d}_A E' \rangle. \tag{3.5}$$

For these semidiscrete equations we have energy conservation. For instance, for the second order formulation, along any trajectory A, the following quantity is constant:

$$\mathcal{E}(A, \dot{A}) = 1/2(\|\dot{A}\|^2 + \|\mathcal{F}(A)\|^2). \tag{3.6}$$

This can be checked directly by inserting $A' = \dot{A}$ in (3.3) and is of course a general property of Hamiltonian systems.

We remark furthermore that we have the Bianchi identity

$$\mathrm{d}_A \mathcal{F}(A) = 0, \tag{3.7}$$

corresponding to the absence of magnetic charges, since $\mathcal{F}(A)$ is represented exactly. That is, for any given time, $A(t)$ is an element of the finite element space $X_h^1 \otimes \mathfrak{g}$ and $\mathcal{F}(A(t))$ is in a corresponding higher-order finite element space specified in section 3.3. The Bianchi identity for the discretization is then just a particular case of the standard Bianchi identity. (The projection of $\mathcal{F}(A(t))$ down to $X_h^2 \otimes \mathfrak{g}$, on the other hand, would provide a field F not necessarily satisfying $\mathrm{d}_A F = 0$.)

The Bianchi identity corresponds to the constraint $\operatorname{div} B = 0$ in electromagnetics. For the analogue of $\operatorname{div} E = 0$ we have greater problems however. For discretizations

of Maxwell's equations this constraint is obtained in a weak form. We now detail this, as it is a source of inspiration for Yang–Mills equations.

The standard semidiscretization of the Euler–Lagrange equation (2.52) for Maxwell's equations would be to look for a magnetic potential of the form $A : t \mapsto A(t) \in X_h^1$ satisfying

$$\forall A' \in X_h^1 \quad \langle \ddot{A}, A' \rangle = -\langle \operatorname{curl} A, \operatorname{curl} A' \rangle. \tag{3.8}$$

Inserting $A' = \operatorname{grad} P'$ for $P' \in X_h^0$ into this equation gives

$$\langle \dot{A}, \operatorname{grad} P' \rangle^{\cdot} = 0, \tag{3.9}$$

when P' is considered as a constant function of time. In particular, if the equality $\langle \dot{A}, \operatorname{grad} P' \rangle = 0$ holds initially for some $P' \in X_h^0$, it will hold at all times.

If a magnetic potential $A \in X_h^1$ satisfies $\langle \dot{A}, \operatorname{grad} P' \rangle = 0$ for all $P' \in X_h^0$, one says that it is weakly (or Galerkin) divergence-free. Ideally one might want strong divergence freedom, and such divergence freedom appears to be necessary for many applications in electromagnetics. However there are indications that [illegible] weak divergence freedom might be just as good as strong divergence freedom. An important example is Kikuchi's compactness property [23]. It asserts that if we have a sequence (A_h) indexed by a mesh parameter $h \to 0$ of fields $A_h \in X_h^1$ such that A_h and $\operatorname{curl} A_h$ are L^2-bounded, and A_h is weakly divergence-free, one can extract a subsequence which is L^2-convergent. This property mimics the (Rellich) compactness of the injection $\mathrm{H}^1 \to \mathrm{L}^2$, since a vector field on a compact manifold which is in L^2 and has its curl and divergence in L^2 is actually in H^1. This property is an essential ingredient in the analysis of the numerical approximation of eigenvalues of the curl curl operator on edge element spaces (see Boffi et al. [4]) and essentially guarantees the absence of spurious modes. The link between spurious modes in eigenvalue computations and problems encountered in time-evolution simulations has been investigated; see, e.g., Hesthaven and Warburton [20], Boffi and Gastaldi [5], and Boffi, Buffa, and Gastaldi [3].

A refinement of Kikuchi's property which is useful for the convergence analysis of nonlinear terms and which was partly inspired by the present work is proved in Christiansen [10]. It indicates that with respect to the convergence of so-called div-curl expressions, if (A_h) is an L^2-bounded sequence of vector fields $A_h \in X_h^1$ which are Galerkin divergence-free, everything happens [illegible] $\operatorname{div} A_h \to 0$ in the H^{-1}-norm.

For noncommutative Lie groups G the simple trick used in (3.9) can no longer be carried out. One would like to be able to insert $A' = \mathrm{d}_A P'$ into (3.3) for $P' \in X_h^0 \otimes \mathfrak{g}$, but this is not legitimate since in general d_A maps $X_h^0 \otimes \mathfrak{g}$ out of $X_h^1 \otimes \mathfrak{g}$. From the point of view of Noether's theorem we remark that the gauge transformations associated with maps $Q : M \to G$ of the form

$$x \mapsto Q(x) = \exp(P(x)), \quad P \in X_h^0 \otimes \mathfrak{g}, \tag{3.10}$$

do not map $X_h^1 \otimes \mathfrak{g}$ into itself, contrary to the linear Maxwell case. We remark however that if Q is constant the associated gauge transformation does map $X_h^1 \otimes \mathfrak{g}$ into itself, since this space is stable under pointwise application of $\operatorname{Ad}(q) : \mathfrak{g} \to \mathfrak{g}$ for any q in G. Alternatively one can simply remark that if $P' \in X_h^0 \otimes \mathfrak{g}$ is a constant map, $A' = \mathrm{d}_A P' = [A, P']$ is indeed in $X_h^1 \otimes \mathfrak{g}$ providing a valid test-function in (3.3). This proves that the scheme conserves [illegible] charge.

One might hope that for any given $P' \in X_h^0 \otimes \mathfrak{g}$, considered as a constant function of time, the quantities $\langle \dot{A}, \mathrm{d}_A P' \rangle$ just oscillate around the initial value which is assumed to be small. We now present some numerical results to see if these hopes are answered.

We will show numerical results obtained with time discretizations according to the leap-frog scheme, which is by far the most popular time discretization of large scale Hamiltonian systems. For its use in the case of Maxwell's equations see Joly [22]. For additional points on the numerical analysis of time and space discretization of Maxwell's equations see Ciarlet and Zou [11]. This discretization scheme should be seen in the context of so-called symplectic integration methods; see, e.g., Berndt [2] for an introduction to symplectic geometry and Hairer, Lubich, and Wanner [19], Leimkuhler and Reich [27], and Marsden and West [28] for numerical aspects. We introduce a time step τ and consider approximations A^i of $A(i\tau)$ obtained by the recursion formula

$$\left\langle \frac{A^{i+1} - 2A^i + A^{i-1}}{\tau^2}, A' \right\rangle = -\langle \mathcal{F}(A^i), \mathrm{D}\mathcal{F}(A^i)A' \rangle. \tag{3.11}$$

Here and in the following all variables appearing with a "′" should be preceded by a "for all" statement referring to a certain space of the form $X_h^p \otimes \mathfrak{g}$, which should be clear from the context. This will ease the notational burden somewhat.

The leap-frog scheme is usually implemented using coupled one-term recurrences rather than a two-term recurrence. One introduces $E^{i+1/2}$ as an approximation of $E((i+1/2)\tau)$ in the following reformulation of (3.11):

$$\frac{A^i - A^{i-1}}{\tau} = -E^{i-1/2}, \tag{3.12}$$

$$\left\langle \frac{E^{i+1/2} - E^{i-1/2}}{\tau}, E' \right\rangle = \langle \mathcal{F}(A^i), \mathrm{D}\mathcal{F}(A^i)E' \rangle. \tag{3.13}$$

Since E and A are not seen at the same time this technique is often called time-staggering. To some extent time staggering is what lets us remain explicit in the time step, a huge advantage compared with implicit schemes for large scale computations.

In Figure 1 we show numerical results obtained for $\mathfrak{su}(2)$-valued gauge potentials on a two-dimensional sphere. Thus in our previous notation $\mathfrak{g} = \mathfrak{su}(2)$ and $M = \mathbb{S}^2$. Recall that $\mathfrak{su}(2)$ is the three-dimensional Lie algebra consisting of trace-free and skew-hermitian 2×2 complex matrices (see, e.g., [14, p. 10]). It is the Lie algebra associated with the compact Lie group SU(2) consisting of unitary 2×2 complex matrices with determinant 1. The finite element spaces were constructed from an unstructured mesh of the sphere (there is no particular reason for choosing a sphere rather than another surface, except that by using a rather symmetric one embedded in $\mathbb{R}^3$, we could visually check some predictions about symmetry properties of the waves).

We have plotted the L^2-norms squared of A^i, $\mathrm{d}A^i$, and $(1/2)[A_i, A_i]$ as a function of time in multiples of π. For the chosen initial values the L^2-norm of the quadratic term $(1/2)[A_i, A_i]$ is relatively small compared with the linear term $\mathrm{d}A^i$ in the curvature. Linearization of the equations indicates that the waves travel around the sphere with speed 1, which accounts some of the almost 2π-periodic features which can be seen in the graphs.

For Maxwell's equation (3.8) the leap-frog discretization is stable under a CFL condition (i.e., $\tau \leq Ch$). This is proved in Joly [22] using the conservation of a discrete

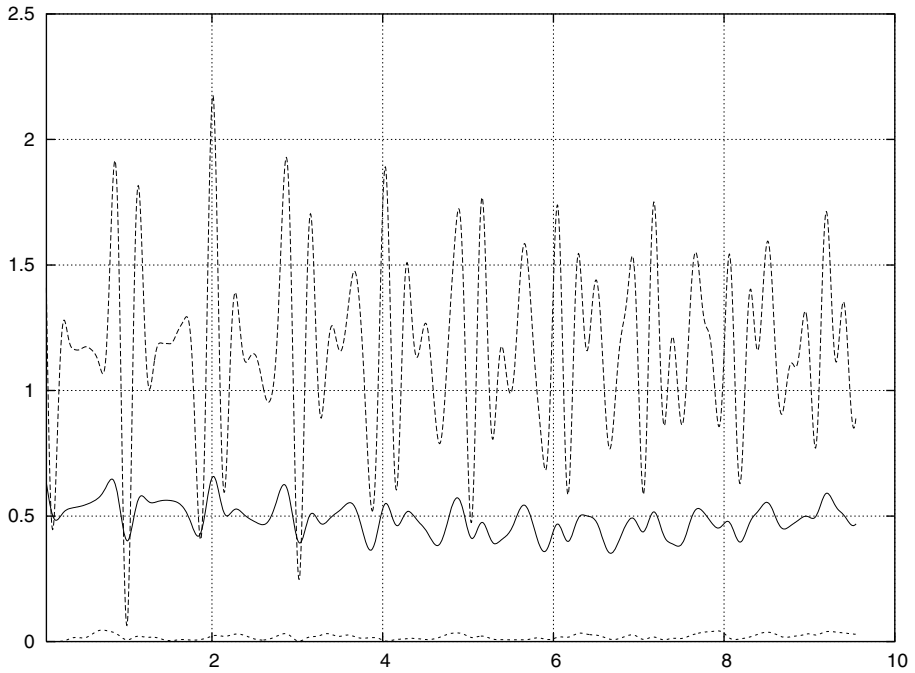


FIG. 1. L^2*-norms squared of* A^i *(plain),* $\mathrm{d}A^i$ *(dashed), and* $(1/2)[A_i, A_i]$ *(finely dashed).*

energy containing a negative term which is controlled by an inverse inequality. This discrete energy is defined as

(3.14)
$$(1/2)\left\|\frac{A^{i+1}-A^i}{\tau}\right\|^2+(1/2)\left(\left\|\operatorname{curl}\left(\frac{A^{i+1}+A^i}{2}\right)\right\|^2-\left\|\operatorname{curl}\left(\frac{A^{i+1}-A^i}{2}\right)\right\|^2\right).$$

For the general Yang–Mills equations we have not found a similar conserved discrete energy. However, at least for the numerical examples, the leap-frog scheme seems to be stable for roughly the same values of the time step τ as for the Maxwell case. For the numerical results presented here the time step is 0.01, which is a few times less than the stability limit. The graphs show 3000 time steps. It might be added that in the course of our simulations we found that there is a remarkably sharp value for τ (depending on initial conditions) above which the scheme was unstable but under which the numerical results appeared rather satisfactory. If we call τ_0 such a value, the interval of time steps for which the behavior was erratic but stable was less than $\tau_0/100$ wide.

In Figure 2 we have plotted some reasonable approximations of the energy as a function of time. These are

$$\mathcal{E}^{i+1/2}=(1/2)\left\|\frac{A^{i+1}-A^i}{\tau}\right\|^2+(1/4)(\|\mathcal{F}(A^i)\|^2+\|\mathcal{F}(A^{i+1})\|^2) \tag{3.15}$$

and

$$\mathcal{E}^{i}=(1/2)\left\|\frac{A^{i+1}-A^{i-1}}{2\tau}\right\|^2+(1/2)\|\mathcal{F}(A^i)\|^2. \tag{3.16}$$

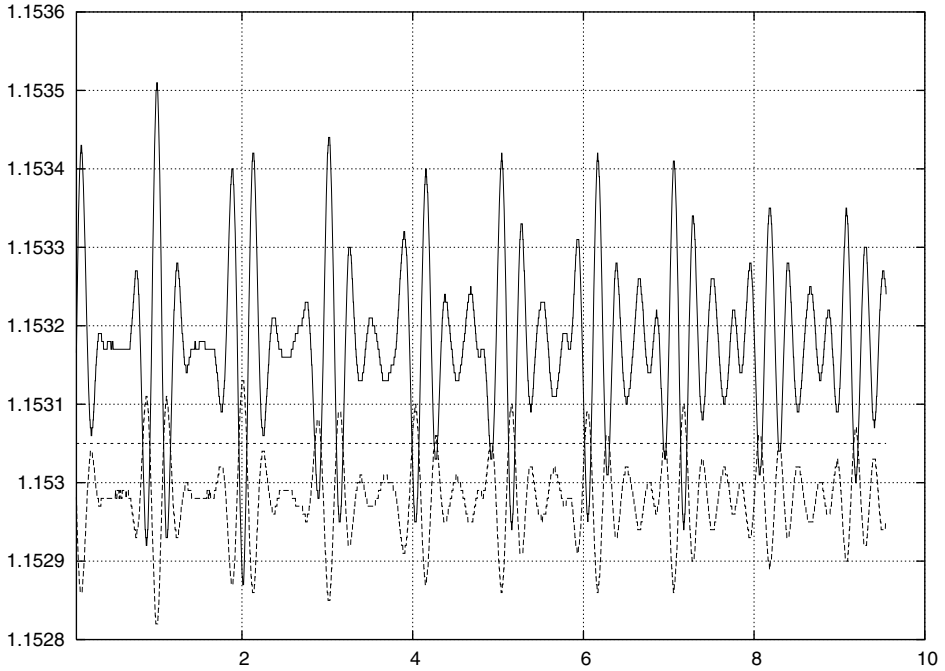


FIG. 2. *Energies* $\mathcal{E}^{i-1/2}$ *(plain),* $\mathcal{E}^i$ *(dashed), and* $(1/6)(\mathcal{E}^{i-1/2} + 4\mathcal{E}^i + \mathcal{E}^{i+1/2})$ *(finely dashed).*

We have also plotted a linear combination of these two energies which yielded very good conservation (the conservation cannot be exact however for convex combinations, since this would imply unconditional stability).

In Figure 3 we have plotted two graphs. One is the L^2-norm squared of a discrete divergence field $D^i \in X_h^0 \otimes \mathfrak{g}$ defined as the solution of (for all $D' \in X_h^0 \otimes \mathfrak{g}$)

$$\langle D^i, D' \rangle = \left\langle \frac{A^{i+1} - A^{i-1}}{2\tau}, \mathrm{d}D' \right\rangle. \tag{3.17}$$

The other graph is the L^2-norm squared of a field $C^i \in X_h^0 \otimes \mathfrak{g}$ measuring charge defined by (for all $C' \in X_h^0 \otimes \mathfrak{g}$)

$$\langle C^i, C' \rangle = \left\langle \frac{A^{i+1} - A^{i-1}}{2\tau}, \mathrm{d}_{A^i} C' \right\rangle. \tag{3.18}$$

Ideally the charge field C^i should remain small compared with D^i, but this appears not to be the case. This tends to indicate that the field D^i is unreliable. Direct visualization shows that it displays strongly mesh-dependent features, as if a smooth (ideal) field D had been perturbed by the noise C^i. As already remarked, total charge is conserved for the scheme, so that C^i must be oscillatory and visualization shows that oscillations are on the scale of the mesh-width h. We remark furthermore that, by energy conservation, the L^2-norm of $(A^{i+1} - A^{i-1})/2\tau$ is bounded so that the H^{-1}-norm of $\mathrm{d}^\star(A^{i+1} - A^{i-1}/2\tau)$ is also bounded. On the other hand Figure 3 shows that the L^2-norm of D^i tends to grow. Together these observations indicate that D^i develops high-frequency modes. We will come back to this observation for the constrained discretizations.

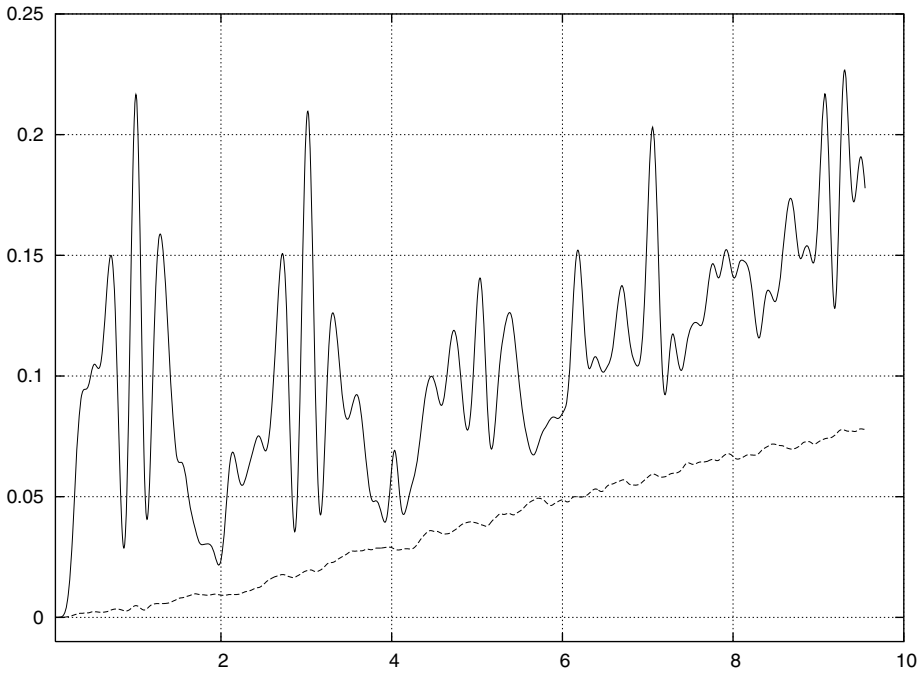


FIG. 3. L^2-*norms squared of divergence* D^i *(plain) and charge* C^i *(dashed).*

We believe this bad behavior with respect to charge conservation to be attributable to the semidiscretization (the space discretization) rather than the leap-frog scheme (the time discretization). That is, the leap-frog scheme just serves to illustrate that the semidiscretization does not provide adequate charge conservation. In this sense the semidiscretization (3.3) is inadequate for simulating Yang–Mills equations. Note that the stability indicated by Figures 1 and 2, together with the drift observed in Figure 3, illustrates the fact that the Yang–Mills system is stable also away from the constraint manifold. But if the Yang–Mills equations were to be coupled with other equations, the development of high frequency modes could be disastrous.

In this paper we will mostly use the leap-frog scheme. This is of course not the only possibility. As an example of an interesting implicit scheme we mention

(3.19)
$$\left\langle \frac{A^{i+1}-2A^i+A^{i-1}}{\tau^2}, A' \right\rangle$$

(3.20)
$$= -\frac{1}{2}\left\langle \mathcal{F}\left(\frac{A^{i-1}+A^i}{2}\right) + \mathcal{F}\left(\frac{A^i+A^{i+1}}{2}\right), \mathrm{D}\mathcal{F}\left(\frac{A^{i-1}}{4}+\frac{A^i}{2}+\frac{A^{i+1}}{4}\right)A' \right\rangle.$$

One of its charms is the following discrete energy conservation property.

PROPOSITION 3.1. [illegible] (3.19), (3.20) [illegible]

(3.21)
$$\frac{1}{2}\left(\left\|\frac{A^{i+1}-A^i}{\tau}\right\|^2 + \left\|\mathcal{F}\left(\frac{A^{i+1}+A^i}{2}\right)\right\|^2\right).$$

. . . . Inserting A' defined by

$$A' = \frac{A^{i+1} - A^{i-1}}{2\tau} = \frac{1}{2}\left(\frac{A^{i+1} - A^i}{\tau} + \frac{A^i - A^{i-1}}{\tau}\right) \tag{3.22}$$

into the left-hand side (3.19) gives

$$\frac{1}{2\tau}\left(\left\|\frac{A^{i+1} - A^i}{\tau}\right\|^2 - \left\|\frac{A^i - A^{i-1}}{\tau}\right\|^2\right). \tag{3.23}$$

Next we use the property that, since the curvature map $\mathcal{F}$ is a polynomial of degree 2, we have, for all gauge potentials A, B,

$$\mathcal{F}(B) = \mathcal{F}(A) + \mathrm{D}\mathcal{F}\left(\frac{A+B}{2}\right)(B - A). \tag{3.24}$$

Applying this identity to

$$A = \frac{A^{i-1} + A^i}{2} \quad \text{and} \quad B = \frac{A^i + A^{i+1}}{2} \tag{3.25}$$

shows that the right-hand side (3.20) with $A' = (1/\tau)(B - A)$ is equal to

$$-\frac{1}{2\tau}\left\langle \mathcal{F}\left(\frac{A^{i-1} + A^i}{2}\right) + \mathcal{F}\left(\frac{A^i + A^{i+1}}{2}\right), \mathcal{F}\left(\frac{A^i + A^{i+1}}{2}\right) - \mathcal{F}\left(\frac{A^{i-1} + A^i}{2}\right)\right\rangle. \tag{3.26}$$

From this the proposition follows. □

This conservation property implies unconditional stability of the scheme. This is useful in particular when the mesh contains strong local refinements.

3.2. Constrained schemes. Since unconstrained schemes have trouble with charge conservation we now examine ways to impose this as a constraint on the flow. This is easier to do on first order formulations such as (3.4), (3.5).

In analogy with the continuous case and standard discretizations of Maxwell's equations, we would like (A, E) to evolve on the subset of $(X_h^1 \otimes \mathfrak{g}) \times (X_h^1 \otimes \mathfrak{g})$ with

$$\forall P' \in X_h^0 \otimes \mathfrak{g} \quad \langle E, \mathrm{d}_A P' \rangle = 0. \tag{3.27}$$

One expects this set to be manifold-like, with singularities located where the kernel of d_A is nontrivial.

We first discuss constraint imposition in general. Suppose we have a system of the form

$$\dot{u} = f(u), \quad g(u) = 0, \tag{3.28}$$

which is compatible in the sense that the vector field f is tangent to the zero-set Z_g of g. One way of imposing constraints on this system is by introducing Lagrange multipliers v and imposing that for all u', v'

$$\langle \dot{u}, u' \rangle + \langle \mathrm{D}g(u)u', v \rangle = \langle f(u), u' \rangle, \tag{3.29}$$

$$\langle \mathrm{D}g(u)\dot{u}, v' \rangle = 0. \tag{3.30}$$

When Dg is onto at each point, the system has a unique solution for given initial values. If the constraint is satisfied initially, it produces trajectories on the zero-set Z_g of g, governed by the vector field defined as the orthogonal projection of f to the tangent spaces of Z_g, according to the duality products $\langle \cdot, \cdot \rangle$. More generally it produces trajectories on the level-sets of g.

When the constraint is compatible, this trajectory must be the one given by (3.28) and the Lagrange multiplier v remains equal to 0. But when the system (3.29), (3.30) is used for incompatible systems, the Lagrange multiplier v will vary, measuring in a sense how incompatible the original system was.

If we apply this method of imposing constraints to the first order system (3.4), (3.5) with the constraint (3.27), we get the following constrained first order system, with Lagrange multipliers $P : t \mapsto P(t) \in X_h^0 \otimes \mathfrak{g}$:

$$\langle \dot{A}, A' \rangle + \langle E, [A', P] \rangle = \langle -E, A' \rangle, \tag{3.31}$$

$$\langle \dot{E}, E' \rangle + \langle E', \mathrm{d}_A P \rangle = \langle \mathcal{F}(A), \mathrm{d}_A E' \rangle, \tag{3.32}$$

$$\langle E, [\dot{A}, P'] \rangle + \langle \dot{E}, \mathrm{d}_A P' \rangle = 0. \tag{3.33}$$

In this system (3.31) can be rewritten using (2.33):

$$\dot{A} = -E + \Pi_h [E, P], \tag{3.34}$$

where Π_h denotes L^2-orthogonal projection onto $X_h^1 \otimes \mathfrak{g}$. In this situation it is not clear what one should call energy of the system, let alone if it is conserved. This is a severe drawback. One could impose an additional constraint on the energy, but this seems cumbersome. Moreover, explicit time discretization of the constrained system (3.31)–(3.33) seems difficult to achieve since the constraint (3.33) appears to be unsuited for time-staggering as in (3.12), (3.13).

In the system (3.31)–(3.33) we see that the term $\langle E, [\dot{A}, P'] \rangle$ ought to be 0. As an alternative to that system we therefore consider the following nonstandard constrained system for fields $A(t), E(t) \in X_h^1 \otimes \mathfrak{g}$ and Lagrange multipliers $P(t) \in X_h^0 \otimes \mathfrak{g}$:

$$\dot{A} = -E, \tag{3.35}$$

$$\langle \dot{E}, E' \rangle + \langle E', \mathrm{d}_A P \rangle = \langle \mathcal{F}(A), \mathrm{d}_A E' \rangle, \tag{3.36}$$

$$\langle \dot{E}, \mathrm{d}_A P' \rangle = 0. \tag{3.37}$$

For the continuous analogue of this system, we remark that by the cancellation property (2.56) given in our first proof of charge conservation in section 2.3, it is equivalent to the unconstrained Yang–Mills equations (2.37) and should yield Lagrange multipliers equal to 0.

For the discrete case we obtain a system which is both energy and constraint preserving (in particular, it is not equivalent to the unconstrained semidiscretization).

PROPOSITION 3.2. [illegible] (3.35)–(3.37) [illegible] (3.27) [illegible]

$$1/2(\|E\|^2 + \|\mathcal{F}(A)\|^2)^{\cdot} = 0. \tag{3.38}$$

[illegible]. Indeed we have for constant P'

$$\langle E, \mathrm{d}_A P' \rangle^{\cdot} = \langle \dot{E}, \mathrm{d}_A P' \rangle + \langle E, [E, P'] \rangle = 0. \tag{3.39}$$

It follows that if the constraint (3.27) is satisfied initially it will be satisfied at all times. Inserting $E' = E$ in (3.36) then gives energy conservation. □

Next we turn to time discretizations of (3.35)–(3.37). Consider the following scheme:

$$\frac{A^i - A^{i-1}}{\tau} = -E^{i-1/2}, \tag{3.40}$$

$$\left\langle \frac{E^{i+1/2} - E^{i-1/2}}{\tau}, E' \right\rangle + \langle E', \mathrm{d}_{A^i} P^i \rangle = \langle \mathcal{F}(A^i), \mathrm{D}\mathcal{F}(A^i) E' \rangle, \tag{3.41}$$

$$\left\langle \frac{E^{i+1/2} - E^{i-1/2}}{\tau}, \mathrm{d}_{A^i} P' \right\rangle = 0. \tag{3.42}$$

Luckily this scheme turns out to be constraint preserving in the following sense.

PROPOSITION 3.3. [illegible] (3.40)–(3.42) [illegible]

$$\left\langle \frac{A^{i+1} - A^{i-1}}{2\tau}, \mathrm{d}_{A^i} P' \right\rangle. \tag{3.43}$$

[illegible] First we remark that

$$\left\langle \frac{A^{i+1} - A^{i-1}}{2\tau}, \mathrm{d}_{A^i} P' \right\rangle = \left\langle \frac{E^{i+1/2} + E^{i-1/2}}{2}, \mathrm{d}_{A^i} P' \right\rangle \tag{3.44}$$

$$= \langle E^{i+1/2}, \mathrm{d}_{A^i} P' \rangle = \langle E^{i-1/2}, \mathrm{d}_{A^i} P' \rangle. \tag{3.45}$$

Then we write

$$\langle E^{i+1/2}, \mathrm{d}_{A^i} P' \rangle - \langle E^{i-1/2}, \mathrm{d}_{A^{i-1}} P' \rangle = \langle E^{i-1/2}, [A^i - A^{i-1}, P'] \rangle \tag{3.46}$$

$$= \langle E^{i-1/2}, [-\tau E^{i-1/2}, P'] \rangle \tag{3.47}$$

$$= 0. \tag{3.48}$$

This gives the desired result. □

Next we show numerical results for the same mesh, time step, and initial values as for the unconstrained case. In Figure 4 we have plotted L^2-norms as in Figure 1 and obtained graphs which are very similar but not identical to the unconstrained case.

In Figure 5 we have plotted the same energies as in Figure 2. Here the similarity with the unconstrained case is surprising. Finally, in Figure 6 we have plotted the divergence and charge defined as for Figure 3. It appears that we have obtained charge conservation without destroying long-time energy conservation. This is somewhat surprising since for symplectic schemes such as the leap-frog scheme existing results on long-time approximate energy conservation are often based on fine properties of Hamiltonian ODEs (see, in particular, [19, chapter 13]) whose extension to constrained discretizations of a PDE are not clear. For the plot of the divergence we have rather large discrepancies with the unconstrained case. For the former discretization, we observed that H^{-1}-boundedness and L^2 growth of D^i indicate the development of high-frequency modes. Since for the constrained scheme, the L^2-norm of D^i grows less (if at all) over a relatively long time period, it appears that we have eliminated the high-frequency modes.

The above constraint preservation is not limited to the leap-frog scheme. The system (3.40)–(3.42) can be expressed as a second order recurrence relation. The discrete acceleration

$$B^i = \frac{A^{i+1} - 2A^i + A^{i-1}}{\tau^2} \tag{3.49}$$

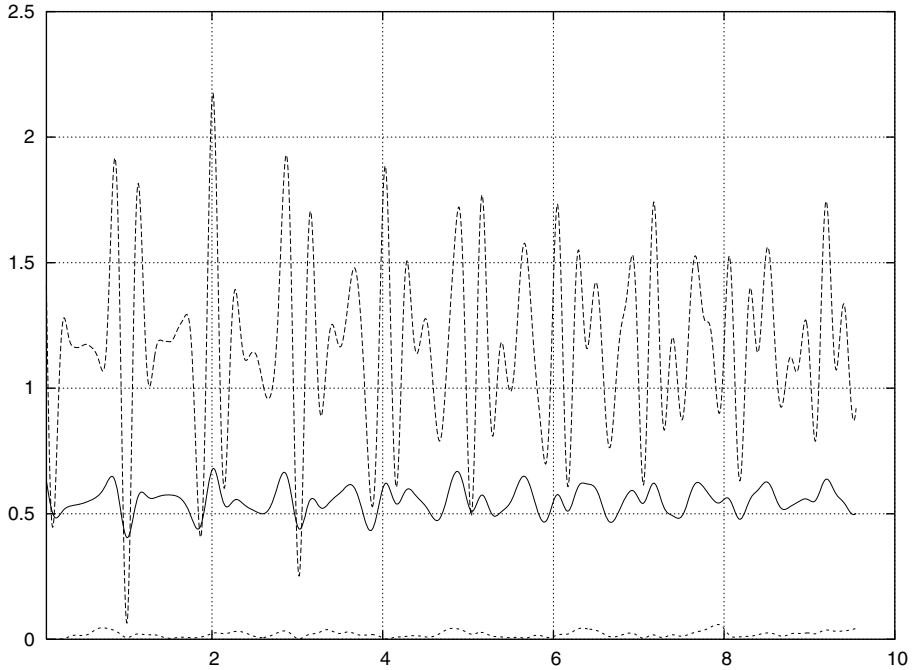


FIG. 4. L^2*-norms squared of* A^i *(plain),* $\mathrm{d}A^i$ *(dashed), and* $(1/2)[A_i, A_i]$ *(finely dashed). Constrained case.*

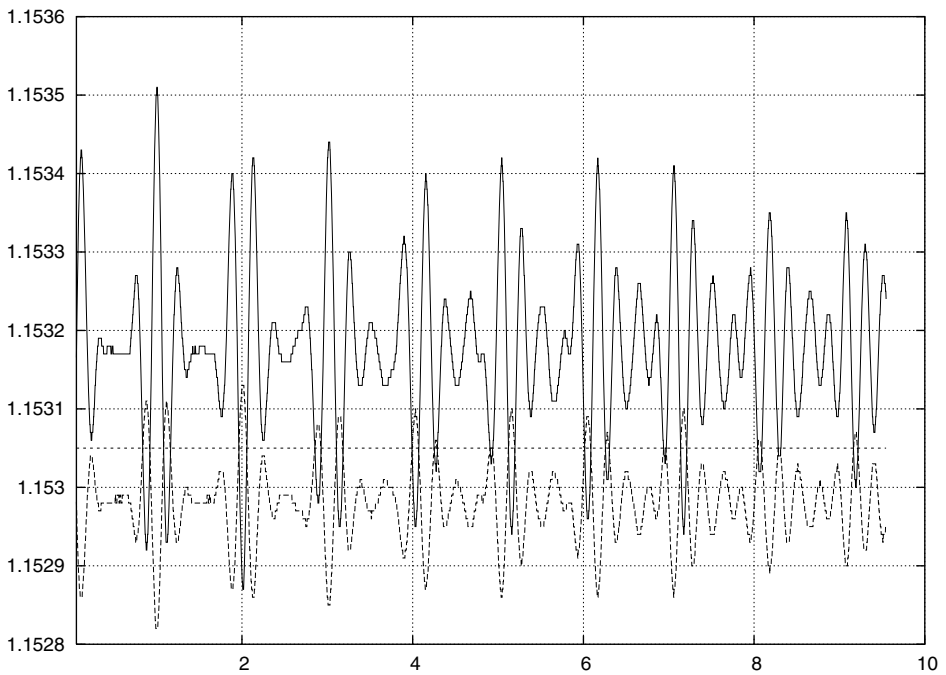


FIG. 5. *Energies* $\mathcal{E}^{i+1/2}$ *(plain),* $\mathcal{E}^i$ *(dashed), and* $(1/6)(\mathcal{E}^{i-1/2}+4\mathcal{E}^i+\mathcal{E}^{i+1/2})$ *(finely dashed). Constrained case.*

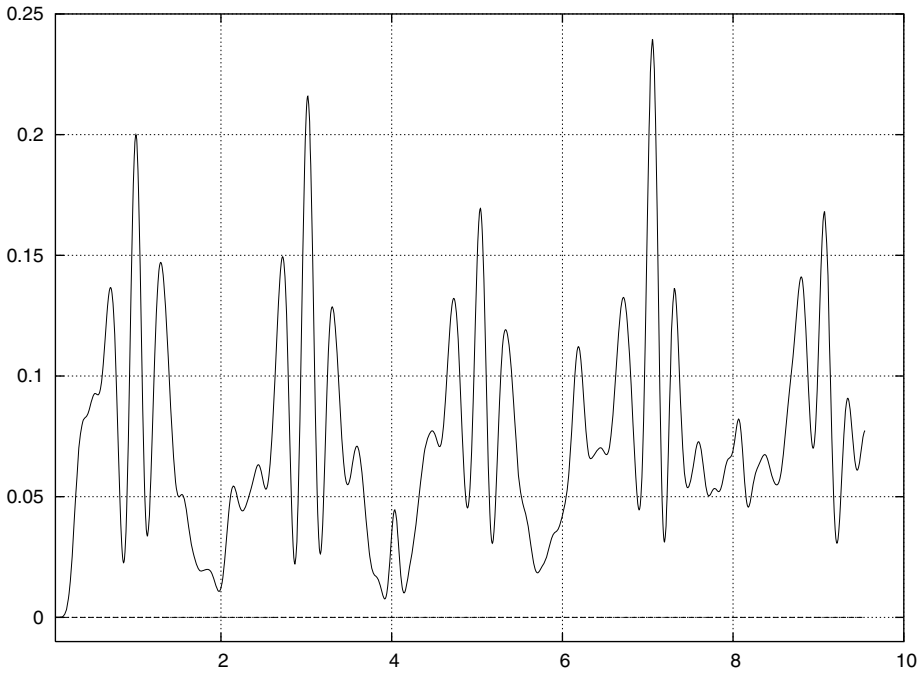


FIG. 6. L^2*-norms squared of divergence* D^i *(plain) and charge* C^i *(dashed). Constrained case.*

should be a solution of

$$\langle B^i, A' \rangle + \langle A', \mathrm{d}_{A^i} P^i \rangle = -\langle \mathcal{F}(A^i), \mathrm{D}\mathcal{F}(A^i) A' \rangle, \tag{3.50}$$

$$\langle B^i, \mathrm{d}_{A^i} P' \rangle = 0. \tag{3.51}$$

The proof of constraint preservation provided for Proposition 3.3 did not depend on the structure of the first term on the right-hand side of (3.40). Therefore we can also impose constraints on the implicit scheme (3.19)–(3.20) by replacing the first term in the right-hand side of (3.50) by the one in (3.20). For this scheme, if the constraint is satisfied initially it will be satisfied for all times, enabling us keep the discrete energy conservation property proved in Proposition 3.1.

3.3. Implementation questions. In this section we comment on some points relevant to the implementation of the above fully discrete algorithm: representation of bilinear maps and resolution of saddle point problems.

Several bilinear maps appear. In particular, we have the wedge of differential forms, the Lie bracket on the chosen Lie algebra, and the bracket of Lie algebra valued differential forms, which is the tensor product of the preceding two maps. Concerning the Lie algebra, a basis (e_i) is chosen and the bilinear map $[\cdot, \cdot]$ is expressed in this basis by the so-called structure constants β_{ij}^k defined by

$$[e_i, e_j] = \sum_k \beta_{ij}^k e_k. \tag{3.52}$$

A similar method can be used to represent the wedge of forms. We are interested in the case of the wedge on $X_h^0 \times X_h^1$ and $X_h^1 \times X_h^1$. From the first one, we can obtain the

bracket of Lie valued forms needed to define the constraint by taking the Kronecker product with the structure constants of the Lie algebra. From the second we obtain the bracket needed to define curvature.

Products of polynomials create polynomials of higher degree so in order to represent these wedge products we must introduce high-order finite element spaces. For simplicity we now skip the index h for the spaces and introduce a new one referring to polynomial degree. For the lowest-order finite element spaces already introduced we use $X^p = X_1^p$.

Denote by X_n^0 the space of continuous functions which are piecewise polynomials of degree at most n. We can also denote by X_0^0 the vector space consisting of the constant functions. The space X_n^0 can also be characterized as the vector space generated by products $u_1 \cdots u_n$, where $u_i \in X_1^0 = X^0$. Next we define the high-order finite element space X_{n+1}^p as the vector space generated by forms uv, where $u \in X_n^0$ and $v \in X_1^p$. Then one checks that

$$\wedge \ : \ X_n^p \times X_m^q \to X_{n+m}^{p+q}. \tag{3.53}$$

It can also be checked that in three dimensions, the family of spaces (X_n^p) corresponds—under the standard identification of differential forms with vector fields—to the first family of mixed finite element spaces as defined in [30], n being the maximum degree of the polynomials used to represent the fields in each tetrahedron (in this context one often says that one uses incomplete polynomials of degree n, since not all differential forms which can be represented piecewise by polynomials of degree n are used; witness the case $n = 1$ for $p \geq 1$).

As already mentioned, to implement our algorithm we are interested in the cases

$$\wedge \ : \ X_1^0 \times X_1^1 \to X_2^1 \quad \text{and} \quad \wedge \ : \ X_1^1 \times X_1^1 \to X_2^2. \tag{3.54}$$

Implementing a basis for high-order finite elements is notoriously cumbersome, since it involves noncanonical choices, which would make for acrobatic indexing exercises when implementing the above wedge products. As a simpler alternative, instead of bases we used generating families which might have linear dependencies. Generally speaking, given a bilinear map $\alpha : E \times F \to G$ between finite dimensional spaces E, F, and G equipped with finite generating families (e_i), (f_j), and (g_k), respectively, suppose we have some family (α_{ij}^k) such that

$$\alpha(e_i, f_j) = \sum_k \alpha_{ij}^k g_k. \tag{3.55}$$

Such a family exists but is in general not unique, but the action of α on vectors expressed as linear combinations of the families (e_i), (f_j) can still be recovered. Explicitly, if

$$u = \sum_i u^i e_i \quad \text{and} \quad v = \sum_j v^j f_j, \tag{3.56}$$

then we have, by simple bilinearity,

$$\alpha(u, v) = w = \sum_k w^k g_k, \quad \text{with} \quad w_k = \sum_{ij} \alpha_{ij}^k u^i v^j. \tag{3.57}$$

We now detail the choice of generating families for $\wedge \ : \ X_1^0 \times X_1^1 \to X_2^1$. For $E = X_1^0$ and $F = X_1^1$ we equip them with their standard (canonical) bases denoted

$(e_i)_{i\in I}$ and $(f_j)_{j\in J}$ indexed by nodes and edges, respectively. A generating family of X_2^1 is provided by the family $(e_i f_j)_{(i,j)\in I\times J}$ but most of these elements are 0. We therefore use the subfamily consisting of, on the one hand, for each edge $j \in J$ with extremities j_0 and j_1 (members of I) in some orientation of the edges, the two 1-forms

$$e_{j_0} f_j \quad \text{and} \quad e_{j_1} f_j, \tag{3.58}$$

and on the other hand, for each triangle k with vertices k_0, k_1, and k_2, if we denote by k_l' the edge opposite to the vertex k_l,

$$e_{k_l} f_{k_l'} \quad \text{with} \quad l = 0, 1, 2. \tag{3.59}$$

This last family is not linearly independent since for each triangle

$$\sum_l e_{k_l} f_{k_l'} = 0. \tag{3.60}$$

As already indicated, it turned out that it was easier to implement the algorithms in this framework rather than ejecting one of the three terms $e_{k_l} f_{k_l'}$ in each triangle (which is enough to ensure linear independence).

Working with tensors of the form (α_{ij}^k) requires routines for treating sparse tensors. For each k there is only a small number (independent of h) of nonzero coefficients. For each k we introduce sets I_k and J_k defined by

$$I_k = \{i \in I \ : \ \exists j \in J \quad \alpha_{ij}^k \neq 0\}, \tag{3.61}$$

$$J_k = \{j \in J \ : \ \exists i \in I \quad \alpha_{ij}^k \neq 0\}. \tag{3.62}$$

We number the elements of I_k and J_k consecutively with integers starting at 1 and store for each element of the $|I_k| \times |J_k|$ matrix thus obtained the corresponding coefficient α_{ij}^k.

While this is certainly not the sparsest representation of the family (α_{ij}^k) it makes the following contraction operation rather quick and easy. In order to assemble the constraint matrix appearing in the saddle point problem (3.41), (3.42) we need to assemble the sparse matrix obtained by contracting the bracket[1] $[\cdot,\cdot] : (X_1^0 \otimes \mathfrak{g}) \times (X_1^1 \otimes \mathfrak{g}) \to X_2^1 \otimes \mathfrak{g}$ with a gauge potential $A^i \in X_1^1 \otimes \mathfrak{g}$. In the above notation, if α is the bracket and v is the gauge potential we want to assemble the matrix of the operator $u \mapsto \alpha(u, v)$. The sparsity pattern of this matrix is determined by the sets I_k and assembling this matrix involves only summing the products

$$\sum_{j\in J_k} \alpha_{ij}^k v^j \tag{3.63}$$

for each k and each $i \in I_k$. This is a small number (independent of h) of operations for each k .

In order to treat the saddle point problem (3.41), (3.42) we need not only assemble the matrix but of course also solve the resulting system. To do this we used an Usawa algorithm. It is an iterative algorithm which requires preconditioning the Schur complement. Indeed, computing the Lagrange multiplier P^i is essentially equivalent to solving a problem of the form

$$\forall P' \in X_h^0 \otimes \mathfrak{g} \quad \langle \Pi_h \mathrm{d}_{A^i} P', \mathrm{d}_{A^i} P^i \rangle = L^i(P'), \tag{3.64}$$

[1]The L^2-duality should also be incorporated in the discussion through its Gram (or mass) matrix on $X_2^1 \otimes \mathfrak{g}$, but this is secondary for the point we want to make here.

where Π_h denotes L^2 projection $X_2^1 \otimes \mathfrak{g} \to X_1^1 \otimes \mathfrak{g}$, and L^i is some linear form on $X_1^0 \otimes \mathfrak{g}$ (this can be checked by directly inserting $E' = \Pi_h \mathrm{d}_{A^i} P'$ into the system). For the numerical results presented here we used as a preconditioner the exact inverse of the matrix of the bilinear form on $(X_1^0 \otimes \mathfrak{g})^2$ defined as

$$(P, P') \to \langle \mathrm{d}P, \mathrm{d}P' \rangle + \langle P, P' \rangle. \tag{3.65}$$

This inverse can be represented by factorization once the nodes have been optimally numbered. The number of times this matrix is used justifies the use of a factorization algorithm, as compared with iterative methods which are often preferred for large scale problems. The above choice of preconditioner was guided by the fact that lower-order terms are compact perturbations of the Laplacian, so that one expects spectral clustering around 1 for the preconditioned system. The Babuska–Brezzi compatibility condition (see Brezzi [7]) we will prove in the next section also proves that this preconditioner yields a condition number bounded uniformly in h.

We now prove a Babuska–Brezzi compatibility condition, ensuring that the above saddle point problem (3.41), (3.42) (or, equivalently, (3.64)) is well-posed. It is slightly more technical than usual since d_A might have a nontrivial kernel. Recall that a continuous operator is said to be left semi-Fredholm if it has closed range and finite dimensional kernel. The sum of a left semi-Fredholm operator and a compact operator is again left semi-Fredholm. In the following proposition we suppose that M is three-dimensional.

PROPOSITION 3.4. [illegible] $A \in \mathrm{L}^3$ [illegible] $\mathrm{d}_A : \mathrm{H}^1 \to \mathrm{L}^2$ [illegible] S [illegible] H^1 [illegible] $S \cap \ker \mathrm{d}_A = \{0\}$ [illegible] $\overline{h} > 0$ [illegible] $C > 0$ [illegible] $h < \overline{h}$ [illegible]

$$\inf_{P' \in X_h^0 \otimes \mathfrak{g} \cap S} \sup_{A' \in X_h^1 \otimes \mathfrak{g}} \frac{|\langle A', \mathrm{d}_A P' \rangle|}{\|A'\|_0 \|P'\|_1} \geq 1/C, \tag{3.66}$$

[illegible]. Since $\mathrm{d}^\star \mathrm{d} : \mathrm{H}^1 \to \mathrm{H}^{-1}$ is Fredholm, d is left semi-Fredholm $\mathrm{H}^1 \to \mathrm{L}^2$. Next we show that for A in L^3 the map $\mathrm{H}^1 \to \mathrm{L}^2$ given by $P \mapsto [A, P]$ is compact. This in turn immediately implies that d_A is left semi-Fredholm.

We have a continuous (but noncompact) Sobolev injection $\mathrm{H}^1 \to \mathrm{L}^6$. Moreover Hölder's inequality shows that $[\cdot, \cdot]$ is continuous $\mathrm{L}^3 \times \mathrm{L}^6 \to \mathrm{L}^2$. Thus $[\cdot, \cdot]$ is continuous $\mathrm{L}^3 \times \mathrm{H}^1 \to \mathrm{L}^2$.

Pick gauge potentials $A_n \in \mathrm{L}^\infty$ such that $(A_n) \to A$ in L^3. Then $[A_n, \cdot]$ converges in norm to $[A, \cdot]$ in the space of maps $\mathrm{H}^1 \to \mathrm{L}^2$. Moreover $[A_n, \cdot]$ is a compact map $\mathrm{H}^1 \to \mathrm{L}^2$ as follows from the (Rellich) compactness of the injection $\mathrm{H}^1 \to \mathrm{L}^2$. Thus $[A, \cdot]$ is a norm limit of compact operators $\mathrm{H}^1 \to \mathrm{L}^2$, and hence it is compact.

Since d maps X_h^0 into X_h^1 there is trivially $C > 0$ such that for all h we have

$$\inf_{P' \in X_h^0 \otimes \mathfrak{g},\ \int P' = 0} \sup_{A' \in X_h^1 \otimes \mathfrak{g}} \frac{|\langle A', \mathrm{d}P' \rangle|}{\|A'\|_0 \|P'\|_1} \geq 1/C. \tag{3.67}$$

Thus $\mathrm{d}_A = \mathrm{d} + [A, \cdot]$ is a compact perturbation of a map satisfying a (one-sided) uniform discrete Inf-Sup condition (up to a finite dimensional space). The proposition then follows from Corollaries 1.11 and 1.13 in Christiansen [9]. □

In geometric terms this proposition shows that the infinitesimal generators of the gauge transformations $\mathcal{G}_Q$ associated with Lie group valued maps $Q : M \to G$ of the

form

$$x \mapsto Q(x) = \exp(P(x)), \tag{3.68}$$

with $P \in X_h^0 \otimes \mathfrak{g}$, are far from orthogonal to $X_h^1 \otimes \mathfrak{g}$. This property should be compared to the case of a commutative Lie group G (e.g., Maxwell) for which $X_h^1 \otimes \mathfrak{g}$ is stable under such gauge transformations, which, as already mentioned, provides an interpretation of the conservation property (3.9) in terms of Noether's theorem.

Typically one would like to apply the Babuska–Brezzi compatibility property along trajectories $t \mapsto A(t) \in X_h^1 \otimes \mathfrak{g}$. For this purpose we mention that Corollary 1.17 in Christiansen [9] automatically extends certain estimates which follow from Proposition 3.4 to hold [illegible] for [illegible] of gauge potentials A that are compact in L^3.

4. Perspectives. We believe that Yang–Mills equations provide a test bench for many problems in the theory of discretizations of constrained PDEs and we mention in this conclusion directions for future investigations.

Extension to nontrivial principal bundles seems to be a challenging geometric problem and is under investigation.

Multiplying the Lie bracket on $\mathfrak{g}$ by a real parameter provides some freedom in tuning the strength of the nonlinearity. That is, we can introduce a coupling parameter $\gamma \in \mathbb{R}$ and consider the curvature

$$\mathcal{F}_\gamma(A) = \mathrm{d}A + (\gamma/2)[A, A]. \tag{4.1}$$

We are interested in trajectories obtained from the associated Lagrangian

$$\mathcal{L}_\gamma(A, A') = (1/2)\|A'\|^2 - (1/2)\|\mathcal{F}_\gamma(A)\|^2. \tag{4.2}$$

The limit $\gamma \to 0$ is often referred to as the weak coupling limit, while the case $\gamma = 0$ is equivalent to Maxwell's equations. Various series expansions, e.g., in powers of γ, around the "trivial" case $\gamma = 0$ are used in many circumstances (for instance, KAM theory) to provide information where other methods fail.

We mention also that, in physics, quantized versions of the Yang–Mills equations are the most important. Given a representation of the Lie group G on a vector space V, a gauge potential on M gives rise to a connection on the space of V-valued maps on M which is compatible with G, and vice versa. The lattice gauge theory due to Wilson consists in using a cubic lattice and attaching Lie group elements to each edge, representing the parallel transport of a connection. The curvature of the connection has degrees of freedom attached to each square of the lattice, called Wilson loops. This discretization appears to be linked to the presently proposed finite element method in much the same way as, for Maxwell's equations, the Yee scheme, which is a staggered finite difference scheme on cubic lattices, is related to variational mixed finite element methods.

Acknowledgments. This work was inspired by a discussion among Douglas N. Arnold, Alan D. Rendall, and the second author at an IMA "hot topics" workshop on Numerical Relativity in June 2002. We thank Geir Ellingsrud and Roman Kozlov for interesting discussions on connections and geometric integration. We also thank Per Christian Moan and Jan B. Thomassen for their remarks.

REFERENCES

[1] D. N. Arnold, *Differential complexes and numerical stability*, Plenary address delivered at the International Congress of Mathematicians (ICM 2002), Beijing, 2002.

[2] R. Berndt, *An Introduction to Symplectic Geometry*, Grad. Stud. Math. 26, AMS, Providence, RI, 2000.

[3] D. Boffi, A. Buffa, and L. Gastaldi, *Convergence Analysis for Hyperbolic Evolution Problems in Mixed Form*, IMATI Technical report 17-PV, IMATI del C.N.R., Pavia, Italy, 2005.

[4] D. Boffi, P. Fernandes, L. Gastaldi, and I. Perugia, *Computational models of electromagnetic resonators: Analysis of edge element approximation*, SIAM J. Numer. Anal., 36 (1999), pp. 1264–1290.

[5] D. Boffi and L. Gastaldi, *Analysis of finite element approximation of evolution problems in mixed form*, SIAM J. Numer. Anal., 42 (2004), pp. 1502–1526.

[6] A. Bossavit, *Mixed finite elements and the complex of Whitney forms*, in The Mathematics of Finite Elements and Applications VI (Uxbridge, UK, 1987), Academic Press, London, 1988, pp. 137–144.

[7] F. Brezzi, *On the existence, uniqueness and approximation of saddle-point problems arising from Lagrangian multipliers*, RAIRO Anal. Numér., 8 (1974), pp. 129–151.

[8] F. Brezzi and M. Fortin, *Mixed and Hybrid Finite Element Methods*, Springer-Verlag, New York, 1991.

[9] S. H. Christiansen, *Discrete Fredholm properties and convergence estimates for the electric field integral equation*, Math. Comp., 73 (2004), pp. 143–167.

[10] S. H. Christiansen, *A div-curl lemma for edge elements*, SIAM J. Numer. Anal., 43 (2005), pp. 116–126.

[11] P. Ciarlet, Jr., and J. Zou, *Fully discrete finite element approaches for time-dependent Maxwell's equations*, Numer. Math., 82 (1999), pp. 193–219.

[12] L. Demkowicz, *hp-adaptive finite elements for time-harmonic Maxwell equations*, in Topics in Computational Wave Propagation, Lect. Notes Comput. Sci. Eng. 31, Springer-Verlag, Berlin, 2003, pp. 163–199.

[13] S. K. Donaldson and P. B. Kronheimer, *The Geometry of Four-Manifolds*, Oxford University Press, Oxford, UK, 1990.

[14] J. J. Duistermaat and J. A. C. Kolk, *Lie groups*, Springer-Verlag, New York, 2000.

[15] D. Eardley and V. Moncrief, *The global existence of Yang-Mills-Higgs fields in* M^{3+1}, Comm. Math. Phys., 83 (1982), pp. 171–212.

[16] G. B. Folland, *Real Analysis: Modern Techniques and Their Applications*, 2nd ed., John Wiley & Sons, New York, 1999.

[17] S. I. Gelfand and Y. I. Manin, *Homological Algebra*, Springer-Verlag, Berlin, 1999.

[18] V. Girault and P.-A. Raviart, *Finite Element Methods for Navier-Stokes Equations*, Springer-Verlag, Berlin, 1986.

[19] E. Hairer, C. Lubich, and G. Wanner, *Geometric Numerical Integration: Structure Preserving Algorithms for Ordinary Differential Equations*, Springer Ser. Comput. Math. 31, Springer-Verlag, New York, 2002.

[20] J. S. Hesthaven and T. Warburton, *High-order nodal discontinuous Galerkin methods for the Maxwell eigenvalue problem*, Philos. Trans. R. Soc. Lond. Ser. A Math. Phys. Eng. Sci., 362 (2004), pp. 493–524.

[21] R. Hiptmair, *Finite elements in computational electromagnetism*, in Acta Numerica 2002, Acta Numer. 339, Cambridge University Press, Cambridge, UK, 2002, pp. 237–339.

[22] P. Joly, *Variational methods for time-dependent wave propagation problems*, in Topics in Computational Wave Propagation, Lect. Notes Comput. Sci. Eng. 31, Springer-Verlag, Berlin, 2003, pp. 201–264.

[23] F. Kikuchi, *On a discrete compactness property for the Nédélec finite elements*, J. Fac. Sci. Univ. Tokyo Sect. 1A Math., 36 (1989), pp. 479–490.

[24] S. Klainerman and M. Machedon, *Finite energy solutions for Yang-Mills equations in* $\mathbb{R}^{3+1}$, Ann. of Math. (2), 142 (1995), pp. 39–119.

[25] S. Lang, *Differential and Riemannian Manifolds*, 3rd ed., Springer-Verlag, New York, 1995.

[26] L. Lehner, *Numerical relativity: A review*, Classical Quantum Gravity, 18 (2001), pp. 25–86.

[27] B. Leimkuhler and S. Reich, *Simulating Hamiltonian Dynamics*, Cambridge Monogr. Appl. Comput. Math. 14, Cambridge University Press, Cambridge, UK, 2004.

[28] J. E. Marsden and M. West, *Discrete mechanics and variational integrators*, in Acta Numerica, 2001, Acta Numer. 10, Cambridge University Press, Cambridge, UK, 2001, pp. 357–514.

[29] P. Monk, *Finite Element Methods for Maxwell's Equations*, Oxford University Press, Oxford, UK, 2003.
[30] J.-C. Nédélec, *Mixed finite elements in* $\mathbb{R}^3$, Numer. Math., 35 (1980), pp. 315–341.
[31] J.-C. Nédélec, *A new family of mixed finite elements in* $\mathbb{R}^3$, Numer. Math., 50 (1986), pp. 57–81.
[32] J. E. Roberts and J.-M. Thomas, *Mixed and hybrid methods*, in Handbook of Numerical Analysis, Vol. II: Finite Element Methods, P. G. Ciarlet and J.-L. Lions, eds., North–Holland, Amsterdam, 1991, pp. 523–639.
[33] R. A. Ryan, *Introduction to Tensor Products of Banach Spaces*, Springer Monogr. Math., Springer-Verlag, London, 2002.
[34] I. Segal, *The Cauchy problem for the Yang-Mills equations*, J. Funct. Anal., 33 (1979), pp. 175–194.
[35] R. Sorkin, *The electromagnetic field on a simplicial net*, J. Math. Phys., 16 (1975), pp. 2432–2440.
[36] T. Tao, *Local well-posedness of the Yang-Mills equation in the temporal gauge below the energy norm*, J. Differential Equations, 189 (2003), pp. 366–382.
[37] M. Taylor, *Partial Differential Equations, Vol.* II: *Qualitative Studies of Linear Equations*, Springer-Verlag, New York, 1996.
[38] K. Uhlenbeck, *Connections with* L^p *bounds on curvature*, Comm. Math. Phys., 82 (1982), pp. 31–42.
[39] R. M. Wald, *General Relativity*, University of Chicago Press, Chicago, IL, 1984.
[40] A. Weil, *Sur les théorèmes de Rham*, Comment. Math. Helv., 26 (1952), pp. 119–145.
[41] H. Whitney, *Geometric Integration Theory*, Princeton University Press, Princeton, NJ, 1957.

SIAM J. SCI. COMPUT.
Vol. 28, No. 1, pp. 102–122
© 2006 Society for Industrial and Applied Mathematics

NEW NONOVERLAPPING DOMAIN DECOMPOSITION METHODS FOR THE HARMONIC MAXWELL SYSTEM*

ANA ALONSO RODRÍGUEZ† AND LUCA GERARDO-GIORDA†

Abstract. We study a nonoverlapping domain decomposition method for the harmonic Maxwell equations with a new kind of interface condition. Using Fourier analysis we derive suitable families of transmission conditions in $\mathbb{R}^3$ that involve second order tangential differential operators and that guarantee convergence for both propagative and evanescent modes. Such families depend upon parameters that are chosen to optimize the convergence rate of the corresponding iterative algorithm. We then propose iterative solvers for the Maxwell equations based on a domain decomposition procedure where such conditions are enforced on the interface. Some numerical results for a two-domain decomposition show the effectiveness of the optimized interface conditions.

Key words. domain decomposition methods, harmonic Maxwell system, optimized interface conditions

AMS subject classifications. 65N55

DOI. 10.1137/040608696

1. Introduction.

The numerical solution of the time-harmonic Maxwell system over a large three-dimensional domain is a difficult task. The linear system resulting from the finite element approximation is complex and the matrix is neither Hermitian nor definite. To construct an efficient iterative method to solve this system, domain decomposition can be used. We concentrate on nonoverlapping domain decomposition methods (see, for instance, [9]). The idea is to decompose the domain into several nonoverlapping subdomains and to solve in parallel the local problems. This procedure leads to an iterative method that converges to the solution of the original problem if the solutions in the subdomains are related by means of suitable boundary conditions at the interface. The performance of the method depends drastically on the choice of this interface conditions.

The earliest nonoverlapping domain decomposition method for the time-harmonic Maxwell system was introduced in [4] with an interface condition of Robin type. For this algorithm it is proved that the local solutions converge to the corresponding restrictions of the global solution weakly in the spaces of complex vector functions of L^2 with curl in L^2. In his Ph.D. thesis, Chevalier [2] proposed a modification of the Robin interface condition by adding a second order differential operator along the interface. For this new algorithm there is not an accurate convergence analysis. In [3] an interface condition involving a second order tangential operator is analyzed and some numerical results are given for a model problem with conditions of radiative type on the whole boundary. A Fourier analysis shows that, for the interface conditions used in these three algorithms, the iteration map is not a contraction in the presence of evanescent modes.

The use of a second order interface operator arises naturally since the Fourier analysis shows that the exact interface conditions involve a pseudodifferential operator and second order tangential derivatives. By approximating the pseudodifferential

*Received by the editors May 20, 2004; accepted for publication (in revised form) August 15, 2005; published electronically March 3, 2006.
http://www.siam.org/journals/sisc/28-1/60869.html
†Dipartimento di Matematica, Università di Trento, Via Sommarive 14, I-38050 Povo (TN), Italy (alonso@science.unitn.it, gerardo@cmapx.polytechnique.fr).

operator we identify two families of interface conditions that lead to iteration maps that are contractive for both propagative and evanescent modes. The first family depends on a complex parameter $p \in \mathbb{C}$, while the second one depends on two parameters, one real ($r \in \mathbb{R}$) and one purely imaginary (iq, $q \in \mathbb{R}$). We will refer in what follows to the first family as [illegible] interface conditions and to the second as [illegible] interface conditions. The parameters $p \in \mathbb{C}$ and $r, q \in \mathbb{R}$ can then be optimized, namely, they can be chosen to optimize the convergence rate of the corresponding iterative procedure, as done by Gander, Magoulès, and Nataf for the Helmholtz equation (see [5]).

The paper is organized as follows. In section 2 the model problem in $\mathbb{R}^3$ is solved by means of an additive Schwarz method in a very general framework: using Fourier analysis, we derive two families of interface conditions which guarantee convergence for both propagative and evanescent modes. An optimization is then performed on the parameters describing such families. In section 3 the proposed domain decomposition algorithms are detailed for a two-domain decomposition at both the continuous and the discrete level. In section 4 the algorithms are validated by some numerical tests. Finally, in section 5, a survey of the nonoverlapping domain decomposition algorithms for the harmonic Maxwell system that previously appeared in literature is given: the interface conditions in these algorithms can be easily included in the general framework we introduce in section 2 and are compared numerically with the conditions we propose.

2. Optimized interface conditions for a model problem in $\mathbb{R}^3$. The harmonic solutions of Maxwell equations are complex valued fields $\mathbf{E}$ and $\mathbf{H}$ such that the fields

$$\mathcal{E}(\mathbf{x}, t) = \mathcal{R}e(\mathbf{E}(\mathbf{x})\exp(i\omega t)),$$
$$\mathcal{H}(\mathbf{x}, t) = \mathcal{R}e(\mathbf{H}(\mathbf{x})\exp(i\omega t))$$

satisfy the Maxwell system. The positive number ω is called the pulsation of the harmonic wave. $\mathbf{E}$ and $\mathbf{H}$ satisfy the time-harmonic equations

$$\begin{aligned} &\operatorname{curl} \mathbf{E} + i\omega\mu\mathbf{H} = \mathbf{0}, \\ &\operatorname{curl} \mathbf{H} - i\omega\varepsilon\mathbf{E} = \mathbf{J}, \end{aligned} \tag{1}$$

where $\mathbf{J}$ is the amplitude of the applied current density.

The model problem we consider is the second order time-harmonic Maxwell equation obtained from (1) by eliminating either the magnetic field $\mathbf{H}$ or the electric filed $\mathbf{E}$. It reads

$$\operatorname{curl}\operatorname{curl} \mathbf{u} - \omega^2\mathbf{u} = \mathbf{F} \quad \text{in } \Omega, \tag{2}$$

where, for the sake of simplicity in notations, we have normalized both the electric permittivity ε and the magnetic permeability μ (i.e., $\varepsilon = \mu = 1$). When $\mathbf{u}$ stands for the electric field, $\mathbf{F} := -i\omega\mathbf{J}$; on the other hand, if $\mathbf{u}$ is the magnetic field, $\mathbf{F} := \operatorname{curl} \mathbf{J}$.

We present the analysis in the domain $\Omega = \mathbb{R}^3$ with the radiation condition

$$\lim_{r\to\infty} r(\operatorname{curl} \mathbf{u} \times \mathbf{n} + i\omega\mathbf{u}) = \mathbf{0}, \tag{3}$$

where $r = |\mathbf{x}|$, $\mathbf{n} = \mathbf{x}/|\mathbf{x}|$.

We decompose the domain in two nonoverlapping subdomains $\Omega_B := \mathbb{R}^2 \times (-\infty, 0)$ and $\Omega_T := \mathbb{R}^2 \times (0, +\infty)$ and we set $\Gamma := \overline{\Omega_B} \cap \overline{\Omega_T}$. Since, as is well known in domain

decomposition literature, an iterative algorithm of Schwarz type without overlap does not converge when Dirichlet interface conditions are used, we consider the following iterative algorithm with more general interface conditions: given $\mathbf{u}_B^0$ and $\mathbf{u}_T^0$, for $j \geq 0$ solve

$$
\begin{cases} \operatorname{curl}\operatorname{curl}\mathbf{u}_B^{j+1} - \omega^2\mathbf{u}_B^{j+1} = \mathbf{F} & \text{in } \Omega_B, \\ \operatorname{curl}\mathbf{u}_B^{j+1}\times\mathbf{n}_B + S_B(\mathbf{u}_{B,\tau}^{j+1}) = \operatorname{curl}\mathbf{u}_T^{j}\times\mathbf{n}_B + S_B(\mathbf{u}_{T,\tau}^{j}) & \text{on } \Gamma, \end{cases}
$$

$$
\begin{cases} \operatorname{curl}\operatorname{curl}\mathbf{u}_T^{j+1} - \omega^2\mathbf{u}_T^{j+1} = \mathbf{F} & \text{in } \Omega_T, \\ \operatorname{curl}\mathbf{u}_T^{j+1}\times\mathbf{n}_T + S_T(\mathbf{u}_{T,\tau}^{j+1}) = \operatorname{curl}\mathbf{u}_B^{j}\times\mathbf{n}_T + S_T(\mathbf{u}_{B,\tau}^{j}) & \text{on } \Gamma. \end{cases} \tag{4}
$$

Here S_B and S_T are two linear operators acting in the tangential direction on the interface and $\mathbf{u}_{L,\tau}^j := \mathbf{n}_B \times \mathbf{u}_L^j \times \mathbf{n}_B = \mathbf{n}_T \times \mathbf{u}_L^j \times \mathbf{n}_T$ for $L = B, T$. It is then easy to see that if $\ker(S_B + S_T) = \{\mathbf{0}\}$, we recover at convergence the solution of the single domain problem.

Since the problems involved are linear, it is enough to analyze the convergence to the zero solution for the homogeneous system. Note that since $\operatorname{curl}\operatorname{curl}\mathbf{v} - \nabla\operatorname{div}\mathbf{v} = -\Delta\mathbf{v}$, the homogeneous equation $\operatorname{curl}\operatorname{curl}\mathbf{u}_L^{j+1} - \omega^2\mathbf{u}_L^{j+1} = \mathbf{0}$ is equivalent to

$$
\begin{aligned} -\Delta\mathbf{u}_L^{j+1} - \omega^2\mathbf{u}_L^{j+1} &= \mathbf{0} \quad \text{in } \Omega_L, \\ \operatorname{div}\mathbf{u}_L^{j+1} &= 0 \quad\;\; \text{in } \Omega_L, \qquad L = B, T. \end{aligned} \tag{5}
$$

We perform a partial Fourier transform in the x and y directions, which we denote with $\mathcal{F}$, and we call k_1 and k_2 the corresponding dual variables. The transform $\mathcal{F}$ is defined as

$$
\mathcal{F} : \mathbf{u}(x, y, z) \to \widehat{\mathbf{u}}(k_1, k_2, z) = \int\!\!\int_{\mathbb{R}^2} \mathbf{u}(x, y, z) e^{-i(k_1 x + k_2 y)}\, dx\, dy.
$$

Then we obtain

$$
\begin{cases} -\frac{\partial^2\widehat{\mathbf{u}}_B^{j+1}}{\partial z^2} + (k_1^2 + k_2^2 - \omega^2)\widehat{\mathbf{u}}_B^{j+1} = \mathbf{0}, & (k_1, k_2) \in \mathbb{R}^2,\ z < 0, \\ -ik_1\widehat{u}_{B,1}^{j+1} - ik_2\widehat{u}_{B,2}^{j+1} + \partial_z\widehat{u}_{B,3}^{j+1} = 0, & (k_1, k_2) \in \mathbb{R}^2,\ z < 0, \\ \mathcal{F}(\operatorname{curl}\mathbf{u}_B^{j+1})\times\mathbf{n}_B + \sigma_B(k_1, k_2)(\widehat{\mathbf{u}}_{B,\tau}^{j+1}) & \\ \qquad = \mathcal{F}(\operatorname{curl}\mathbf{u}_T^{j})\times\mathbf{n}_B + \sigma_B(k_1, k_2)(\widehat{\mathbf{u}}_{T,\tau}^{j}), & (k_1, k_2) \in \mathbb{R}^2,\ z = 0, \end{cases} \tag{6}
$$

$$
\begin{cases} -\frac{\partial^2\widehat{\mathbf{u}}_T^{j+1}}{\partial z^2} + (k_1^2 + k_2^2 - \omega^2)\widehat{\mathbf{u}}_T^{j+1} = \mathbf{0}, & (k_1, k_2) \in \mathbb{R}^2,\ z > 0, \\ -ik_1\widehat{u}_{T,1}^{j+1} - ik_2\widehat{u}_{T,2}^{j+1} + \partial_z\widehat{u}_{T,3}^{j+1} = 0, & (k_1, k_2) \in \mathbb{R}^2,\ z > 0, \\ \mathcal{F}(\operatorname{curl}\mathbf{u}_T^{j+1})\times\mathbf{n}_T + \sigma_T(k_1, k_2)(\widehat{\mathbf{u}}_{T,\tau}^{j+1}) & \\ \qquad = \mathcal{F}(\operatorname{curl}\mathbf{u}_B^{j})\times\mathbf{n}_T + \sigma_T(k_1, k_2)(\widehat{\mathbf{u}}_{B,\tau}^{j}), & (k_1, k_2) \in \mathbb{R}^2,\ z = 0, \end{cases}
$$

where σ_L denotes the symbol of the operator S_L.

The general solutions of $-\frac{\partial^2\widehat{\mathbf{u}}_L}{\partial z^2} + (k_1^2 + k_2^2 - \omega^2)\widehat{\mathbf{u}}_L = \mathbf{0}$, $(L = B, T)$ are

$$
\widehat{\mathbf{u}}_L(k_1, k_2, z) = \mathbf{a}_L(k_1, k_2)e^{\lambda z} + \mathbf{b}_L(k_1, k_2)e^{-\lambda z},
$$

where $\lambda = \lambda(k_1, k_2) = \sqrt{k_1^2 + k_2^2 - \omega^2}$ denotes the root of the characteristic equation which is either real and positive or purely imaginary. Since the radiation conditions exclude growing solutions as well as incoming modes at infinity we obtain that $\mathbf{b}_B(k_1, k_2) = \mathbf{a}_T(k_1, k_2) = \mathbf{0}$, hence

$$\widehat{\mathbf{u}}_B(k_1, k_2, z) = \mathbf{a}_B(k_1, k_2)e^{\lambda z},$$
$$\widehat{\mathbf{u}}_T(k_1, k_2, z) = \mathbf{b}_T(k_1, k_2)e^{-\lambda z}.$$

From the transformed divergence-free conditions $-ik_1\widehat{u}_{L,1} - ik_2\widehat{u}_{L,2} + \partial_z\widehat{u}_{L,3} = 0$ we get

$$\begin{aligned} \lambda\widehat{u}_{B,3} &= \partial_z\widehat{u}_{B,3} = i(k_1\widehat{u}_{B,1} + k_2\widehat{u}_{B,2}) \\ -\lambda\widehat{u}_{T,3} &= \partial_z\widehat{u}_{T,3} = i(k_1\widehat{u}_{T,1} + k_2\widehat{u}_{T,2}); \end{aligned}$$

thus, for $\lambda \neq 0$,

$$\widehat{u}_{B,3} = \frac{i}{\lambda}(k_1\widehat{u}_{B,1} + k_2\widehat{u}_{B,2}), \qquad \widehat{u}_{T,3} = -\frac{i}{\lambda}(k_1\widehat{u}_{T,1} + k_2\widehat{u}_{T,2}).$$

This means that we can write $\mathcal{F}(\operatorname{curl} \mathbf{u}_L) \times \mathbf{n}_L$ in terms of the tangential components on Γ, $(\widehat{u}_{L,1}, \widehat{u}_{L,2})$. In fact we can prove the following result.

LEMMA 1. [illegible] $\mathbf{u}_L$ $L = B, T$ [illegible]

$$\begin{aligned} &-\Delta\mathbf{u}_L - \omega^2\mathbf{u}_L = \mathbf{0} && \text{[illegible]}\ \Omega_L, \\ &\operatorname{div} \mathbf{u}_L = 0 && \text{[illegible]}\ \Omega_L, \\ &\lim_{r\to\infty} r(\operatorname{curl} \mathbf{u}_L \times \mathbf{n} + i\omega\mathbf{u}_L) = 0, \end{aligned}$$

[illegible] $\mathbf{U}_L := \begin{bmatrix} \widehat{u}_{L,1} \\ \widehat{u}_{L,2} \end{bmatrix}$ [illegible]

$$\lambda\,\mathcal{F}(\operatorname{curl} \mathbf{u}_L) \times \mathbf{n}_L = \begin{bmatrix} R\,\mathbf{U}_L - \omega^2\mathbf{U}_L \\ \mathbf{0} \end{bmatrix},$$

[illegible] $R = R(k_1, k_2)$ [illegible]

$$R := \begin{bmatrix} k_2^2 & -k_1k_2 \\ -k_1k_2 & k_1^2 \end{bmatrix}.$$

[illegible]. This result can be found in [2]. We include here the proof for completeness. We consider the case $L = B$ but the proof is analogous for $L = T$. Since

$$\mathcal{F}(\operatorname{curl} \mathbf{u}_B) = \begin{bmatrix} -ik_2\widehat{u}_{B,3} - \dfrac{\partial\widehat{u}_{B,2}}{\partial z} \\ \dfrac{\partial\widehat{u}_{B,1}}{\partial z} + ik_1\widehat{u}_{B,3} \\ -ik_1\widehat{u}_{B,2} + ik_2\widehat{u}_{B,1} \end{bmatrix} = \begin{bmatrix} \dfrac{k_2}{\lambda}(k_1\widehat{u}_{B,1} + k_2\widehat{u}_{B,2}) - \lambda\widehat{u}_{B,2} \\ \lambda\widehat{u}_{B,1} - \dfrac{k_1}{\lambda}(k_1\widehat{u}_{B,1} + k_2\widehat{u}_{B,2}) \\ -ik_1\widehat{u}_{B,2} + ik_2\widehat{u}_{B,1} \end{bmatrix}$$

we easily conclude

$$\lambda\mathcal{F}(\operatorname{curl}\mathbf{u}_B)\times\mathbf{n}_B=\begin{bmatrix}\lambda^2\widehat{u}_{B,1}-k_1(k_1\widehat{u}_{B,1}+k_2\widehat{u}_{B,2})\\ -k_2(k_1\widehat{u}_{B,1}+k_2\widehat{u}_{B,2})+\lambda^2\widehat{u}_{B,2}\\ 0\end{bmatrix}$$

$$=\begin{bmatrix}(k_2^2-\omega^2)\widehat{u}_{B,1}-k_1k_2\widehat{u}_{B,2}\\ -k_2k_1\widehat{u}_{B,1}+(k_1^2-\omega^2)\widehat{u}_{B,2}\\ 0\end{bmatrix}. \qquad \square$$

1. We recall that for a function $\mathbf{q}:\mathbb{R}^2\to\mathbb{R}^2$, $\operatorname{curl}\mathbf{q}=\frac{\partial q_2}{\partial x}-\frac{\partial q_1}{\partial y}$ and

$$\overrightarrow{\operatorname{curl}}_\Gamma\operatorname{curl}_\Gamma\mathbf{q}=\begin{bmatrix}\dfrac{\partial}{\partial y}\left(\dfrac{\partial q_2}{\partial x}-\dfrac{\partial q_1}{\partial y}\right)\\ -\dfrac{\partial}{\partial x}\left(\dfrac{\partial q_2}{\partial x}-\dfrac{\partial q_1}{\partial y}\right)\end{bmatrix}.$$

It is then easy to see that

$$R\,\mathbf{U}_L=\mathcal{F}(\overrightarrow{\operatorname{curl}}_\Gamma\operatorname{curl}_\Gamma\mathbf{u}_{L,\tau}).$$

Since the third components in the above expressions vanish identically, we can express the action of one iteration of the algorithm (6), in terms of $\mathbf{U}_B$ and $\mathbf{U}_T$, as

$$\tfrac{1}{\lambda}(R\,\mathbf{U}_B^{j+1}-\omega^2\mathbf{U}_B^{j+1})+\sigma_B\mathbf{U}_B^{j+1}=-\tfrac{1}{\lambda}(R\,\mathbf{U}_T^{j}-\omega^2\mathbf{U}_T^{j})+\sigma_B\mathbf{U}_T^{j},$$

$$\tfrac{1}{\lambda}(R\,\mathbf{U}_T^{j+1}-\omega^2\mathbf{U}_T^{j+1})+\sigma_T\mathbf{U}_T^{j+1}=-\tfrac{1}{\lambda}(R\,\mathbf{U}_B^{j}-\omega^2\mathbf{U}_B^{j})+\sigma_T\mathbf{U}_B^{j}.$$

If σ_B and σ_T are such that the matrix $(R-\omega^2I+\lambda\sigma_L)$ is invertible for $L=B$, T, we have

$$\mathbf{U}_B^{j+1}=(R-\omega^2I+\lambda\sigma_B)^{-1}(-R+\omega^2I+\lambda\sigma_B)\mathbf{U}_T^{j},$$

$$\mathbf{U}_T^{j+1}=(R-\omega^2I+\lambda\sigma_T)^{-1}(-R+\omega^2I+\lambda\sigma_T)\mathbf{U}_B^{j}.$$

If we let

$$M_B:=(R-\omega^2I+\lambda\sigma_B)^{-1}(-R+\omega^2I+\lambda\sigma_B)(R-\omega^2I+\lambda\sigma_T)^{-1}(-R+\omega^2I+\lambda\sigma_T)$$

and

$$M_T=(R-\omega^2I+\lambda\sigma_T)^{-1}(-R+\omega^2I+\lambda\sigma_T)(R-\omega^2I+\lambda\sigma_B)^{-1}(-R+\omega^2I+\lambda\sigma_B),$$

we can write $\mathbf{U}_B^{j+2}=M_B\mathbf{U}_B^{j}$, $\mathbf{U}_T^{j+2}=M_T\mathbf{U}_T^{j}$. Given $\mathbf{U}_B^0$ and $\mathbf{U}_T^0$, we get

$$\mathbf{U}_B^{2n}=M_B^n\mathbf{U}_B^0,\qquad\qquad \mathbf{U}_T^{2n}=M_T^n\mathbf{U}_T^0.$$

Since the matrices M_B and M_T share the same spectrum as a function of k_1 and k_2, we can define the convergence rate of the algorithm as being the spectral radius of M_B (or, equivalently, M_T), namely,

$$\rho(k_1,k_2):=\rho(M_T)=\rho(M_B).$$

Thus, the sequence $(\mathbf{U}_B^j,\mathbf{U}_T^j)$ converges to zero if and only if $\rho<1$.

Notice that choosing $\sigma_L = \frac{1}{\lambda_L}(R - \omega^2 I)$, for $L = B, T$, we can easily derive

$$M_B = M_T = \frac{-\lambda_B + \lambda}{\lambda_B + \lambda} \frac{-\lambda_T + \lambda}{\lambda_T + \lambda} I. \tag{7}$$

Choosing $\lambda_B = \lambda_T = \lambda(k)$ we have $\rho(M_T) = \rho(M_B) = 0$, and the algorithm converges in two iterations. Unfortunately, due to the presence of the square root in the symbol, this choice corresponds to nonlocal operators S_L which cannot be used in practice in the real domain where computations take place. To overcome this problem we approximate λ by the symbol of a local operator. The easiest solution to avoid an increase in the order of the interface operator is to replace λ with a constant. In other words, we take $\lambda_L = p_L \in \mathbb{C}$ independent of k_1 and k_2 for $L = B, T$. In this way S_L is the second order operator

$$S_L = \frac{1}{p_L}(\overrightarrow{\mathrm{curl}}_\Gamma \mathrm{curl}_\Gamma - \omega^2 I), \tag{8}$$

where curl_Γ is the surface scalar curl operator and $\overrightarrow{\mathrm{curl}}_\Gamma$ the surface vector curl operator. In the remaining part of the section we present two suitable choices for such parameters.

2. Another choice would be to approximate $\lambda(|k|)$ by means of a low order truncation of its Taylor expansion that is given by

$$\lambda(|k|) = i\omega - i\frac{|k|^2}{2\omega} - i\frac{|k|^4}{8\omega^3} + O(|k|^6).$$

Such expansion is a suitable approximation only for small values of $|k|$. Moreover, since $\lambda(|k|)$ takes positive real values for $|k| > \omega$, a purely imaginary approximation as the one provided by this expansion is not efficient: when $\lambda(|k|)$ is real the numerators and denominators in (7) are complex conjugated numbers and $\rho(M_B) = \rho(M_T) = 1$.

2.1. Symmetric interface conditions. The most natural choice is to take $p_B = p_T = p \in \mathbb{C}$, which amounts to use the same operator in the interface condition on both subdomains: we thus refer to this choice as [illegible]. With this assumption, owing to (7), the sequence $(\mathbf{U}_B^j, \mathbf{U}_T^j)$ converges if and only if

$$\rho(p, |k|) = \left|\frac{-p + \lambda}{p + \lambda}\right|^2 < 1.$$

On one hand, for evanescent modes ($|k|^2 > \omega^2$), we have that

$$\begin{aligned}\left|\frac{-p + \lambda}{p + \lambda}\right|^2 &= \frac{(-\mathcal{R}e\, p + \sqrt{|k|^2 - \omega^2})^2 + (\mathcal{I}m\ p)^2}{(\mathcal{R}e\, p + \sqrt{|k|^2 - \omega^2})^2 + (\mathcal{I}m\ p)^2} \\ &= \frac{|p|^2 + |k|^2 - \omega^2 - 2\,\mathcal{R}e\, p\sqrt{|k|^2 - \omega^2}}{|p|^2 + |k|^2 - \omega^2 + 2\,\mathcal{R}e\, p\sqrt{|k|^2 - \omega^2}} < 1\end{aligned}$$

if and only if $\mathcal{R}e\, p > 0$.

On the other hand, for propagative modes ($|k|^2 < \omega^2$), we similarly have

$$\begin{aligned}\left|\frac{-p + \lambda}{p + \lambda}\right|^2 &= \frac{(\mathcal{R}e\, p)^2 + (-\mathcal{I}m\, p + \sqrt{\omega^2 - |k|^2})^2}{(\mathcal{R}e\, p)^2 + (\mathcal{I}m\, p + \sqrt{\omega^2 - |k|^2})^2} \\ &= \frac{|p|^2 + \omega^2 - |k|^2 - 2\,\mathcal{I}m\, p\sqrt{\omega^2 - |k|^2}}{|p|^2 + \omega^2 - |k|^2 + 2\,\mathcal{I}m\, p\sqrt{\omega^2 - |k|^2}} < 1\end{aligned}$$

if and only if $\mathcal{I}m\, p > 0$.

Notice, however, that for $|k|^2 = \omega^2$ the convergence rate $\rho(p, |k|) = 1$ for all $p \in \mathbb{C}$.

This analysis leads us to consider algorithm (4) with the second order interface conditions

$$
(9) \quad \begin{cases} \operatorname{curl} \mathbf{u}_B^{j+1} \times \mathbf{n}_B + \frac{1}{p}(\overrightarrow{\operatorname{curl}}_\Gamma \operatorname{curl}_\Gamma - \omega^2 I)(\mathbf{u}_{B,\tau}^{j+1}) \\ \qquad\qquad = \operatorname{curl} \mathbf{u}_T^{j} \times \mathbf{n}_B + \frac{1}{p}(\overrightarrow{\operatorname{curl}}_\Gamma \operatorname{curl}_\Gamma - \omega^2 I)(\mathbf{u}_{T,\tau}^{j}), \\ \operatorname{curl} \mathbf{u}_T^{j+1} \times \mathbf{n}_T + \frac{1}{p}(\overrightarrow{\operatorname{curl}}_\Gamma \operatorname{curl}_\Gamma - \omega^2 I)(\mathbf{u}_{T,\tau}^{j+1}) \\ \qquad\qquad = \operatorname{curl} \mathbf{u}_B^{j} \times \mathbf{n}_T + \frac{1}{p}(\overrightarrow{\operatorname{curl}}_\Gamma \operatorname{curl}_\Gamma - \omega^2 I)(\mathbf{u}_{B,\tau}^{j}) \end{cases}
$$

on Γ. The complex parameter p is chosen with $\mathcal{R}e\, p, \mathcal{I}m\, p \in \mathbb{R}^+$. In the following we call

$$\mathbb{C}^+ := \{q \in \mathbb{C} \mid \mathcal{R}e\, q > 0, \mathcal{I}m\, q > 0\}.$$

The best choice of p is the one that gives a minimal spectral radius. As we have noted before, if $|k|^2 = \omega^2$, then $\rho(p, |k|) = 1$ for all $p \in \mathbb{C}^+$. However, following [5], we can restrict ourselves to consider spatial frequencies such that $|k| \in I := (m, \omega_-) \cup (\omega_+, M)$, where ω_- and ω_+ are parameters to be chosen such that $\omega_- < \omega < \omega_+$, where m denotes the smallest value of $|k|$ relevant to the subdomain and where M is the largest value of $|k|$ supported by the numerical grid. This largest frequency is of order π/h. Hence we choose p^{opt} as being the solution of the optimization problem

$$\rho_{\text{S}}^{\text{opt}} = \min_{p \in \mathbb{C}^+} \max_{|k| \in I} |\rho(p, |k|)|$$

that coincides with the one analyzed in [5], whose solution is given by

$$
(10) \qquad \mathcal{R}e\, p^{\text{opt}} = \mathcal{I}m\, p^{\text{opt}} = \frac{\left[(\omega^2 - \omega_-^2)(M^2 - \omega^2)\right]^{1/4}}{\sqrt{2}}
$$

and provides the convergence rate

$$\rho_{\text{S}}^{\text{opt}} = \frac{1 - \sqrt{2}\left(\frac{\omega^2 - \omega_-^2}{M^2 - \omega^2}\right)^{1/4} + \left(\frac{\omega^2 - \omega_-^2}{M^2 - \omega^2}\right)^{1/2}}{1 + \sqrt{2}\left(\frac{\omega^2 - \omega_-^2}{M^2 - \omega^2}\right)^{1/4} + \left(\frac{\omega^2 - \omega_-^2}{M^2 - \omega^2}\right)^{1/2}}.$$

2.2. Unsymmetric interface conditions. Another opportunity stems from the expression of matrices M_B and M_T in (7) and from the fact that λ is either real and positive or purely imaginary: choosing one parameter real and positive and the other one purely imaginary (with positive imaginary part), the reduction factor is strictly smaller than 1 for all modes. To fix the ideas, if we take, for instance, $p_B = r \in \mathbb{R}^+$, and $p_T = i\, q,\ q \in \mathbb{R}^+$, the reduction factor is given by

$$
(11) \qquad \rho(r, q, |k|) = \left| \frac{-r + \lambda}{r + \lambda} \, \frac{-iq + \lambda}{iq + \lambda} \right|
$$

(and we claim that the inverse choice provides the same result). We immediately have for propagative modes

$$
(12) \qquad \left| \frac{-r + \lambda}{r + \lambda} \right| = 1, \qquad \left| \frac{-iq + \lambda}{iq + \lambda} \right| < 1,
$$

as well as for evanescent modes

$$\left|\frac{-r+\lambda}{r+\lambda}\right| < 1, \qquad \left|\frac{-iq+\lambda}{iq+\lambda}\right| = 1. \tag{13}$$

The algorithm we are thus led to consider is based on the following interface conditions on Γ (recall that $(iq)^{-1} = -i/q$):

$$\begin{cases} \operatorname{curl} \mathbf{u}_B^{j+1} \times \mathbf{n}_B + \frac{1}{r}(\overrightarrow{\operatorname{curl}}_\Gamma \operatorname{curl}_\Gamma - \omega^2 I)(\mathbf{u}_{B,\tau}^{j+1}) \\ \qquad = \operatorname{curl} \mathbf{u}_T^{j} \times \mathbf{n}_B + \frac{1}{r}(\overrightarrow{\operatorname{curl}}_\Gamma \operatorname{curl}_\Gamma - \omega^2 I)(\mathbf{u}_{T,\tau}^{j}), \\ \operatorname{curl} \mathbf{u}_T^{j+1} \times \mathbf{n}_T - \frac{i}{q}(\overrightarrow{\operatorname{curl}}_\Gamma \operatorname{curl}_\Gamma - \omega^2 I)(\mathbf{u}_{T,\tau}^{j+1}) \\ \qquad = \operatorname{curl} \mathbf{u}_B^{j} \times \mathbf{n}_T - \frac{i}{q}(\overrightarrow{\operatorname{curl}}_\Gamma \operatorname{curl}_\Gamma - \omega^2 I)(\mathbf{u}_{B,\tau}^{j}). \end{cases} \tag{14}$$

As this choice entails the use of different operators in the construction of matching conditions on the two sides of the interface, we refer to it as [illegible].

Again, the best choice of the parameters $p_B = r$ and $p_T = iq$ is the one that gives a minimal convergence rate. Once again, if $|k|^2 = \omega^2$, then $\rho(r, q, |k|) = 1$ for each $r, q \in \mathbb{R}$. However, in this case, owing to (12) and (13), an optimization procedure for the parameters can be performed separately on the two intervals $I_1 = (m, \omega_-)$ and $I_2 = (\omega_+, M)$, where m, M, ω_-, and ω_+ are the ones introduced in the previous section. The min-max problem we are dealing with is thus decoupled and reads

$$\rho_{\mathrm{NS}}^{\mathrm{opt}} = \begin{cases} \min\limits_{q \in \mathbb{R}^+} \max\limits_{|k| \in I_1} \left|\dfrac{-iq+\lambda}{iq+\lambda}\right| & \text{for propagative modes,} \\ \min\limits_{r \in \mathbb{R}^+} \max\limits_{|k| \in I_2} \left|\dfrac{-r+\lambda}{r+\lambda}\right| & \text{for evanescent modes,} \end{cases} \tag{15}$$

which has an elegant analytical solution, given by the following lemma.

LEMMA 2. [illegible] (15) [illegible]

$$\begin{cases} q^{\mathrm{opt}} = \left[(\omega^2 - \omega_-^2)\,(\omega^2 - m^2)\right]^{1/4} & \text{for propagative modes,} \\ r^{\mathrm{opt}} = \left[(\omega_+^2 - \omega^2)\,(M^2 - \omega^2)\right]^{1/4} & \text{for evanescent modes,} \end{cases}$$

[illegible]

$$\rho_{\mathrm{NS}}^{\mathrm{opt}} = \begin{cases} \dfrac{(\omega^2 - \omega_-^2)^{1/4} - (\omega^2 - m^2)^{1/4}}{(\omega^2 - \omega_-^2)^{1/4} + (\omega^2 - m^2)^{1/4}} & \text{for propagative modes,} \\ \dfrac{(M^2 - \omega^2)^{1/4} - (\omega_+^2 - \omega^2)^{1/4}}{(M^2 - \omega^2)^{1/4} + (\omega_+^2 - \omega^2)^{1/4}} & \text{for evanescent modes.} \end{cases} \tag{16}$$

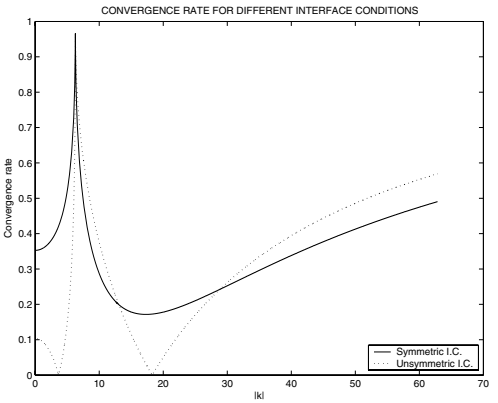


FIG. 1. *Convergence rate (as a function of $|k|$) in Fourier space of the optimized Schwarz algorithm with symmetric (solid line) and unsymmetric (dashed line) interface conditions.*

. We outline the solution for propagative modes, the solution for evanescent ones being similar. With the change of variables

$$x := \sqrt{\omega^2 - |k|^2}$$

we are led to study

$$\min_{q\in\mathbb{R}^+} \max_{x\in I^-} \left|\frac{x-q}{x+q}\right|,$$

where $I^- = (\sqrt{\omega^2-\omega_-^2}, \sqrt{\omega^2-m^2}) = (x_0, x_1)$. Since the function $\varphi(x,q) = \frac{x-q}{x+q}$ is monotone increasing for $x \in I^-$, $\varphi(0,q) = -1$, and $\lim_{x\to\infty}\varphi(x,q) = 1$, the maximum of the modulus is attained on the boundary of the interval. The min-max problem reduces thus to

$$\min_{q\in\mathbb{R}^+} \max \left\{ \left|\frac{x_0-q}{x_0+q}\right|, \left|\frac{x_1-q}{x_1+q}\right| \right\}.$$

We thus have that the optimal parameter for propagative modes belongs to the interval (x_0, x_1) and is given by

$$q^{\text{opt}} = \sqrt{x_0 x_1} = \left[(\omega^2-\omega_-^2)\,(\omega^2-m^2)\right]^{1/4}. \tag{17}$$

For evanescent modes, in a similar way, the change of variable $x := \sqrt{|k|^2-\omega^2}$ reduces to study the min-max problem on $\mathbb{R}^+ \times I^+$, where $I^+ = (\sqrt{\omega_+^2-\omega^2}, \sqrt{M^2-\omega^2})$, whose solution is given by

$$r^{\text{opt}} = \left[(\omega_+^2-\omega^2)\,(M^2-\omega^2)\right]^{1/4}. \tag{18}$$

So far, a simple algebra leads to the expression of the optimal reduction factor (16). □

We report in Figure 1 the convergence rates $\rho(p^{\text{opt}}, |k|)$ and $\rho(r^{\text{opt}}, q^{\text{opt}}, |k|)$ for $\omega = 2\pi$. In the solid line we show the convergence rate as a function of $|k|$ of the

symmetric algorithm for $p = p^{\text{opt}}$, and in the dashed line, the convergence rate of the unsymmetric algorithm for $r = r^{\text{opt}}$ and $q = q^{\text{opt}}$. Notice that in both cases it is less than or equal to one and it is equal to one only for $|k| = \omega$. We recall that the optimization has been done separately for the symmetric and the unsymmetric algorithm.

3. Two-domain decomposition. In the following we consider a bounded domain $\Omega \subset \mathbb{R}^3$ and the boundary value problem

$$\text{(19)} \qquad \begin{cases} \text{curl curl } \mathbf{u} - \omega^2 \mathbf{u} = \mathbf{F} & \text{in } \Omega, \\ \text{curl } \mathbf{u} \times \mathbf{n} + i\omega \mathbf{u}_\tau = \mathbf{0} & \text{on } \Gamma_R, \\ \mathbf{u} \times \mathbf{n} = \mathbf{0} & \text{on } \Gamma_D, \end{cases}$$

where Γ_R and Γ_D are two disjoint open subsets of $\partial\Omega$ such that $\overline{\Gamma_R} \cup \overline{\Gamma_D} = \partial\Omega$.

We decompose the domain Ω in two nonoverlapping subdomains Ω_l, $l = 1, 2$. We set $\Gamma = \overline{\Omega_1} \cap \overline{\Omega_2}$ and for simplicity we assume that $\overline{\Gamma} \cap \overline{\Gamma_R} = \emptyset$. Let us set $\Gamma_{R,l} = \Gamma_R \cap \partial\Omega_l$ and $\Gamma_{D,l} = \Gamma_D \cap \partial\Omega_l$. We denote $\mathbf{n}_l$ the unit normal outward vector on $\partial\Omega_l \cap \Gamma$.

We consider the following iterative procedure:

(20)

$$\begin{cases} \text{curl curl } \mathbf{u}_1^{j+1} - \omega^2 \mathbf{u}_1^{j+1} = \mathbf{F} & \text{in } \Omega_1, \\ \text{curl } \mathbf{u}_1^{j+1} \times \mathbf{n}_1 + i\omega \mathbf{u}_{1,\tau}^{j+1} = \mathbf{0} & \text{on } \Gamma_{R,1}, \\ \mathbf{u}_1^{j+1} \times \mathbf{n}_1 = \mathbf{0} & \text{on } \Gamma_{D,1}, \\ \text{curl } \mathbf{u}_1^{j+1} \times \mathbf{n}_1 + \frac{1}{p_1}(\overrightarrow{\text{curl}}_\Gamma \text{curl}_\Gamma - \omega^2 I)(\mathbf{u}_{1,\tau}^{j+1}) & \\ \qquad = \text{curl } \mathbf{u}_2^j \times \mathbf{n}_1 + \frac{1}{p_1}(\overrightarrow{\text{curl}}_\Gamma \text{curl}_\Gamma - \omega^2 I)(\mathbf{u}_{2,\tau}^j) & \text{on } \Gamma; \end{cases}$$

$$\begin{cases} \text{curl curl } \mathbf{u}_2^{j+1} - \omega^2 \mathbf{u}_2^{j+1} = \mathbf{F} & \text{in } \Omega_2, \\ \text{curl } \mathbf{u}_2^{j+1} \times \mathbf{n}_2 + i\omega \mathbf{u}_{2,\tau}^{j+1} = \mathbf{0} & \text{on } \Gamma_{R,2}, \\ \mathbf{u}_2^{j+1} \times \mathbf{n}_2 = \mathbf{0} & \text{on } \Gamma_{D,2}, \\ \text{curl } \mathbf{u}_2^{j+1} \times \mathbf{n}_2 + \frac{1}{p_2}(\overrightarrow{\text{curl}}_\Gamma \text{curl}_\Gamma - \omega^2 I)(\mathbf{u}_{2,\tau}^{j+1}) & \\ \qquad = \text{curl } \mathbf{u}_1^j \times \mathbf{n}_2 + \frac{1}{p_2}(\overrightarrow{\text{curl}}_\Gamma \text{curl}_\Gamma - \omega^2 I)(\mathbf{u}_{1,\tau}^j) & \text{on } \Gamma; \end{cases}$$

where either $p_1 = p_2 = p^{\text{opt}}$, or $p_1 = r^{\text{opt}}$, and $p_2 = iq^{\text{opt}}$, according to the choice of symmetric or unsymmetric interface conditions.

By introducing the new variables

$$\boldsymbol{\lambda}_1^j := -\text{curl } \mathbf{u}_2^j \times \mathbf{n}_2 + \frac{1}{p_1}(\overrightarrow{\text{curl}}_\Gamma \text{curl}_\Gamma - \omega^2 I)(\mathbf{u}_{2,\tau}^j)$$

and

$$\boldsymbol{\lambda}_2^j := -\text{curl } \mathbf{u}_1^j \times \mathbf{n}_1 + \frac{1}{p_2}(\overrightarrow{\text{curl}}_\Gamma \text{curl}_\Gamma - \omega^2 I)(\mathbf{u}_{1,\tau}^j)$$

the algorithm becomes

$$
\begin{cases}
\operatorname{curl}\operatorname{curl}\mathbf{u}_1^{j+1}-\omega^2\mathbf{u}_1^{j+1}=\mathbf{F} & \text{in } \Omega_1,\\
\operatorname{curl}\mathbf{u}_1^{j+1}\times\mathbf{n}_1+i\omega\mathbf{u}_{1,\tau}^{j+1}=\mathbf{0} & \text{on } \Gamma_{R,1},\\
\mathbf{u}_1^{j+1}\times\mathbf{n}_1=\mathbf{0} & \text{on } \Gamma_{D,1},\\
\operatorname{curl}\mathbf{u}_1^{j+1}\times\mathbf{n}_1+\frac{1}{p_1}(\overrightarrow{\operatorname{curl}}_\Gamma\operatorname{curl}_\Gamma-\omega^2 I)(\mathbf{u}_{1,\tau}^{j+1})=\boldsymbol{\lambda}_1^j & \text{on } \Gamma;
\end{cases}
$$

$$
\begin{cases}
\operatorname{curl}\operatorname{curl}\mathbf{u}_2^{j+1}-\omega^2\mathbf{u}_2^{j+1}=\mathbf{F} & \text{in } \Omega_2,\\
\operatorname{curl}\mathbf{u}_2^{j+1}\times\mathbf{n}_2+i\omega\mathbf{u}_{2,\tau}^{j+1}=\mathbf{0} & \text{on } \Gamma_{R,2},\\
\mathbf{u}_2^{j+1}\times\mathbf{n}_2=\mathbf{0} & \text{on } \Gamma_{D,2},\\
\operatorname{curl}\mathbf{u}_2^{j+1}\times\mathbf{n}_2+\frac{1}{p_2}(\overrightarrow{\operatorname{curl}}_\Gamma\operatorname{curl}_\Gamma-\omega^2 I)(\mathbf{u}_{2,\tau}^{j+1})=\boldsymbol{\lambda}_2^j & \text{on } \Gamma;
\end{cases}
\tag{21}
$$

$$
\boldsymbol{\lambda}_1^{j+1}=-\boldsymbol{\lambda}_2^j+\left(\frac{1}{p_1}+\frac{1}{p_2}\right)(\overrightarrow{\operatorname{curl}}_\Gamma\operatorname{curl}_\Gamma-\omega^2 I)(\mathbf{u}_{2,\tau}^{j+1}),
$$

$$
\boldsymbol{\lambda}_2^{j+1}=-\boldsymbol{\lambda}_1^j+\left(\frac{1}{p_1}+\frac{1}{p_2}\right)(\overrightarrow{\operatorname{curl}}_\Gamma\operatorname{curl}_\Gamma-\omega^2 I)(\mathbf{u}_{1,\tau}^{j+1}).
$$

3.1. Weak formulation. We are interested in the finite elements approximation of the bidomain problem (20), so we will consider its weak formulation. First we need to introduce some functional spaces.

Let Σ be a connected subset of $\partial\Omega$

$$
L^2_{\tan}(\Sigma):=\{\mathbf{v}\in L^2(\Sigma)^3\ :\ \mathbf{v}\cdot\mathbf{n}=0 \text{ on } \Sigma\},
$$

$$
H(\operatorname{curl}_\Gamma;\Sigma):=\{\mathbf{q}\in L^2_{\tan}(\Sigma)\ :\ \operatorname{curl}_\Gamma\mathbf{q}\in L^2_{\tan}(\Sigma)\},
$$

$$
H_0(\operatorname{curl}_\Gamma;\Sigma):=\{\mathbf{q}\in H(\operatorname{curl}_\Gamma;\Sigma)\ :\ \mathbf{q}\cdot\mathbf{t}=0 \text{ on } \partial\Sigma\},
$$

where $\mathbf{t}$ denotes a tangent vector to $\partial\Sigma$.

We consider the spaces

$$
H_{\mathrm{imp}}(\operatorname{curl};\Omega):=\{\mathbf{v}\in H(\operatorname{curl};\Omega)\ :\ \mathbf{v}_\tau\in L^2_{\tan}(\partial\Omega)\},
$$

$$
H_{0,\Sigma}(\operatorname{curl};\Omega):=\{\mathbf{v}\in H(\operatorname{curl};\Omega)\ :\ \mathbf{v}_\tau=0 \text{ on } \Sigma\},
$$

$$
H_{\mathrm{reg},\Sigma}(\operatorname{curl};\Omega):=\{\mathbf{v}\in H_{\mathrm{imp}}(\operatorname{curl};\Omega)\ :\ \mathbf{v}_{\tau|\Sigma}\in H(\operatorname{curl}_\Gamma;\Sigma)\}.
$$

The variational formulation of the boundary value problems will be set in the space

$$
V_l:=H_{0,\Gamma_{D,l}}(\operatorname{curl};\Omega_l)\cap H_{\mathrm{reg},\Gamma_{D,l}\cup\Gamma}(\operatorname{curl};\Omega_l)
$$

with $l=1,2$.

We also need the space

$$
V_l^0:=\{\mathbf{v}\in V_l\ :\ \mathbf{v}_\tau=\mathbf{0} \text{ on } \Gamma\}(=H_{0,\Gamma_{D,l}\cup\Gamma}(\operatorname{curl};\Omega_l)\cap H_{\mathrm{imp}}(\operatorname{curl};\Omega_l))\ .
$$

Notice that from the assumption $\overline{\Gamma}\cap\overline{\Gamma_R}=\emptyset$ it follows that $\partial\Gamma\subset\overline{\Gamma_{D,l}}$, hence for all $\mathbf{v}\in V_l$, $\mathbf{v}_{\tau|\Gamma}\in H_0(\operatorname{curl}_\Gamma;\Gamma)$. Moreover for all $\mathbf{p}\in H_0(\operatorname{curl}_\Gamma;\Gamma)$ there exists $\mathcal{E}_l(\mathbf{p})\in H_{0,\partial\Omega_l\setminus\Gamma}(\operatorname{curl};\Omega_l)$ such that $\mathcal{E}_l(\mathbf{p})_{\tau|\Gamma}=\mathbf{p}$. To give the variational formulation

of (20) we will decompose $\mathbf{v} \in V_l$ in the following way: $\mathbf{v} = \mathbf{v}^0 + \mathcal{E}_l(\mathbf{p})$, where $\mathbf{p} = \mathbf{v}_{\tau|\Gamma} \in H_0(\mathrm{curl}_\Gamma; \Gamma)$ and $\mathbf{v}^0 = \mathbf{v} - \mathcal{E}_l(\mathbf{p}) \in V_l^0$.

Owing to the Green formula,

$$\int_\Sigma \overrightarrow{\mathrm{curl}}_\Gamma \mathrm{curl}_\Gamma \mathbf{q} \cdot \mathbf{p} = \int_\Sigma \mathrm{curl}_\Gamma \mathbf{q}\, \mathrm{curl}_\Gamma \mathbf{p} + \int_{\partial\Sigma} \mathrm{curl}_\Gamma \mathbf{q}\, (\mathbf{p} \cdot \mathbf{t}),$$

we are led to consider the bilinear form in $H(\mathrm{curl}_\Gamma; \Gamma)$,

$$s(\mathbf{p}, \mathbf{q}) := \int_\Gamma (\mathrm{curl}_\Gamma \mathbf{p}\, \mathrm{curl}_\Gamma \overline{\mathbf{q}} - \omega^2 \mathbf{p} \cdot \overline{\mathbf{q}}),$$

the bilinear form in $H_{\mathrm{imp}}(\mathrm{curl}; \Omega_l)$,

$$a_l(\mathbf{w}, \mathbf{v}) := \int_{\Omega_l} (\mathrm{curl}\, \mathbf{w} \cdot \mathrm{curl}\, \overline{\mathbf{v}} - \omega^2 \mathbf{w} \cdot \overline{\mathbf{v}}) + i\omega \int_{\Gamma_{R,l}} \mathbf{w}_\tau \cdot \overline{\mathbf{v}}_\tau,$$

the bilinear form in V_l,

$$r_l(\mathbf{w}, \mathbf{v}) = a_l(\mathbf{w}, \mathbf{v}) + \frac{1}{p_l} s(\mathbf{w}_{\tau|\Gamma}, \mathbf{v}_{\tau|\Gamma}),$$

and the linear form

$$L_l(\mathbf{v}) = \int_{\Omega_l} \mathbf{F} \cdot \mathbf{v}.$$

The weak formulation of (20) reads

$$\begin{cases} \text{Find } \mathbf{u}_1^{j+1} \in V_1 : \\ r_1(\mathbf{u}_1^{j+1}, \mathbf{v}) = L_1(\mathbf{v}) & \forall\, \mathbf{v} \in V_1^0, \\ r_1(\mathbf{u}_1^{j+1}, \mathcal{E}_1\mathbf{q}) = L_1(\mathcal{E}_1\mathbf{q}) + L_2(\mathcal{E}_2\mathbf{q}) - r_2(\mathbf{u}_2^j, \mathcal{E}_2\mathbf{q}) & \\ \qquad\qquad + \left(\dfrac{1}{p_1} + \dfrac{1}{p_2}\right) s(\mathbf{u}_{2,\tau}^j, \mathbf{q}) & \forall\, \mathbf{q} \in H_0(\mathrm{curl}_\Gamma, \Gamma), \end{cases}$$

$$\begin{cases} \text{Find } \mathbf{u}_2^{j+1} \in V_2 : \\ r_2(\mathbf{u}_2^{j+1}, \mathbf{v}) = L_2(\mathbf{v}) & \forall\, \mathbf{v} \in V_2^0, \\ r_2(\mathbf{u}_2^{j+1}, \mathcal{E}_2\mathbf{q}) = L_2(\mathcal{E}_2\mathbf{q}) + L_1(\mathcal{E}_1\mathbf{q}) - r_1(\mathbf{u}_1^j, \mathcal{E}_1\mathbf{q}) & \\ \qquad\qquad + \left(\dfrac{1}{p_1} + \dfrac{1}{p_2}\right) s(\mathbf{u}_{1,\tau}^j, \mathbf{q}) & \forall\, \mathbf{q} \in H_0(\mathrm{curl}_\Gamma, \Gamma). \end{cases}$$

The new variables $\boldsymbol{\lambda}_l^j$ introduced in (21) must be intended in the following sense: $\boldsymbol{\lambda}_l^j \in (H_0(\mathrm{curl}_\Gamma; \Gamma))'$ such that for all $\mathbf{q} \in H_0(\mathrm{curl}_\Gamma; \Gamma)$

$$\begin{aligned} \langle \boldsymbol{\lambda}_l^j, \mathbf{q} \rangle_\Gamma &= -[a_m(\mathbf{u}_m^j, \mathcal{E}_m\mathbf{q}) - L_m(\mathcal{E}_m\mathbf{q})] + \frac{1}{p_l} s(\mathbf{u}_{m,\tau}^j, \mathbf{q}) \\ &= -[r_m(\mathbf{u}_m^j, \mathcal{E}_m\mathbf{q}) - L_m(\mathcal{E}_m\mathbf{q})] + \left(\frac{1}{p_1} + \frac{1}{p_2}\right) s(\mathbf{u}_{m,\tau}^j, \mathbf{q}), \end{aligned}$$

$l = 1, 2$, $m = 1, 2$, $l \neq m$.

Hence the weak formulation of (21) reads

$$\begin{cases} \text{Find } \mathbf{u}_1^{j+1} \in V_1 : \\ r_1(\mathbf{u}_1^{j+1}, \mathbf{v}) = L_1(\mathbf{v}) & \forall\, \mathbf{v} \in V_1^0, \\ r_1(\mathbf{u}_1^{j+1}, \mathcal{E}_1\mathbf{q}) = L_1(\mathcal{E}_1\mathbf{q}) + \langle \boldsymbol{\lambda}_1^j, \mathbf{q}\rangle_\Gamma & \forall\, \mathbf{q} \in H_0(\mathrm{curl}_\Gamma, \Gamma), \end{cases}$$

$$\begin{cases} \text{Find } \mathbf{u}_2^{j+1} \in V_2 : \\ r_2(\mathbf{u}_2^{j+1}, \mathbf{v}) = L_2(\mathbf{v}) & \forall\, \mathbf{v} \in V_2^0, \\ r_2(\mathbf{u}_2^{j+1}, \mathcal{E}_2\mathbf{q}) = L_2(\mathcal{E}_2\mathbf{q}) + \langle \boldsymbol{\lambda}_2^j, \mathbf{q}\rangle_\Gamma & \forall\, \mathbf{q} \in H_0(\mathrm{curl}_\Gamma, \Gamma), \end{cases} \tag{22}$$

$$\langle \boldsymbol{\lambda}_1^{j+1}, \mathbf{q}\rangle_\Gamma = -\langle \boldsymbol{\lambda}_2^j, \mathbf{q}\rangle_\Gamma + \left(\frac{1}{p_1} + \frac{1}{p_2}\right) s(\mathbf{u}_{2,\tau}^{j+1}, \mathbf{q}) \qquad \forall\, \mathbf{q} \in H_0(\mathrm{curl}_\Gamma, \Gamma),$$

$$\langle \boldsymbol{\lambda}_2^{j+1}, \mathbf{q}\rangle_\Gamma = -\langle \boldsymbol{\lambda}_1^j, \mathbf{q}\rangle_\Gamma + \left(\frac{1}{p_1} + \frac{1}{p_2}\right) s(\mathbf{u}_{1,\tau}^{j+1}, \mathbf{q}) \qquad \forall\, \mathbf{q} \in H_0(\mathrm{curl}_\Gamma, \Gamma).$$

3.2. Finite elements discretization. We will employ a curl-conforming finite element approximation of the problem. We can use the first family of Nédélec finite elements in a tetrahedral mesh or in a hexaedral mesh (see [7, 1, 6]). Other possibles choices are the finite element families introduced and analyzed by Nédélec in [8].

Let $\{\mathcal{T}_h\}_{h>0}$ be a regular family of finite element meshes of Ω of elements of maximum diameter h such that each $K \in \mathcal{T}_h$ is contained either in $\overline{\Omega_1}$ or in $\overline{\Omega_2}$. We denote $\mathcal{T}_{l,h}$ the restriction of $\mathcal{T}_h$ to Ω_l, $l = 1, 2$.

Let $\mathbf{P}_k$, $k \geq 1$, be the space of polynomials of degree less than or equal to k, and denote by $\mathbf{P}_k^*$ the space of homogeneous polynomials of degree k. We set

$$\Sigma_k := \{\mathbf{p} \in (\mathbf{P}_k^*)^3 |\ \mathbf{p}(\mathbf{x}) \cdot \mathbf{x} = 0\}, \quad R_k := (\mathbf{P}_{k-1})^3 \oplus \Sigma_k.$$

Notice that $(\mathbf{P}_{k-1})^3 \subset R_k \subset (\mathbf{P}_k)^3$. To discuss finite elements built on hexaedra, we also need the polynomial space

$$Q_{n_1,n_2,n_3} := \{\text{polyomials of maximum degree } n_1 \text{ in } x_1,\ n_2 \text{ in } x_2,\ \text{and } n_3 \text{ in } x_3\}.$$

If $\{\mathcal{T}_h\}_{h>0}$ is a family of tetrahedral meshes, we employ the finite element spaces

$$N_{l,h}^k := \{\mathbf{v}_h \in H(\mathrm{curl}; \Omega_l) \mid \mathbf{v}_{h|K} \in R_k \ \forall\ K \in \mathcal{T}_{l,h}\}.$$

On the other hand, if $\{\mathcal{T}_h\}_{h>0}$ is a family of hexaedral meshes, then the curl-conforming finite element space that we employ is given by

$$N_{l,h}^k := \{\mathbf{v}_h \in H(\mathrm{curl}; \Omega_l) \mid \mathbf{v}_{h|K} \in Q_{k-1,k,k} \times Q_{k,k-1,k} \times Q_{k,k,k-1} \ \forall\ K \in \mathcal{T}_{l,h}\}.$$

Now we consider the spaces

$$X_{l,h} = \{\mathbf{v}_h \in N_{l,h}^k \mid \mathbf{v}_h \times \mathbf{n}_l = \mathbf{0} \text{ in } \Gamma_{D,l}\}.$$

A mesh $\mathcal{T}_{l,h}$ of Ω_l induces a mesh on Γ in the sense that the faces of the elements in $\mathcal{T}_{l,h}$ that lie on Γ obey the usual finite elements meshing constraints. Since we have matching grids in the interface, $\mathcal{T}_{1,h}$ and $\mathcal{T}_{2,h}$ induce the same partition of Γ. Moreover the tangential component of an element of $N_{l,h}^k$ on Γ lies in the appropriate two-dimensional analogue of the set $N_{l,h}^k$. Let us denote Y_h the finite elements space induced on Γ by the tangential components of functions of $X_{l,h}$.

Let $\{\mathbf{w}_i^l\}_{i=1}^{N_l}$ be a basis of $X_{l,h}$ and $\{\mathbf{q}_i\}_{i=1}^{n}$ the corresponding basis of Y_h. For simplicity we assume that these bases are such that $\mathbf{q}_i = \mathbf{w}_{N_l-n+i,\tau}$ for $i = 1, \ldots, n$, $l = 1, 2$. We consider the matrices R_l, $l = 1, 2$, and S with entries

$$(R_l)_{i,j} = r_l(\mathbf{w}_j^l, \mathbf{w}_i^l), \quad 1 \le i, j \le N_l,$$

$$S_{i,j} = s(\mathbf{q}_j, \mathbf{q}_i), \quad 1 \le i, j \le n.$$

Moreover we consider the matrices

$$B_l \in \mathbb{R}^{n \times N_l} \quad B_l = [0 \quad I],$$

where 0 denotes the $n \times (N_l - n)$ matrix with all zero entries and I is the $n \times n$ identity matrix. B_1 and B_2 are the trace operators of domains Ω_1 and Ω_2 on the interface Γ.

Finally let us consider the vectors

$$(\mathbf{f}_l)_j = L_l(\mathbf{w}_j^l), \quad 1 \le j \le N_l.$$

Then, the discrete analogous of algorithm (22) reads as follows.

ALGORITHM 1. [illegible] $\Lambda_l^0 \in \mathbb{C}^n$ [illegible] $j \ge 0$ [illegible]

$$R_l \mathbf{U}_l^{j+1} = \mathbf{f}_l + B_l^T \Lambda_l^j,$$

[illegible] $l = 1, 2$ [illegible]

$$\Lambda_1^{j+1} = -\Lambda_2^j + \left(\frac{1}{p_1} + \frac{1}{p_2}\right) S B_2 \mathbf{U}_2^{j+1},$$

$$\Lambda_2^{j+1} = -\Lambda_1^j + \left(\frac{1}{p_1} + \frac{1}{p_2}\right) S B_1 \mathbf{U}_1^{j+1}.$$

Denoting $\mathbf{g}_l = R_l^{-1} \mathbf{f}_l$ this algorithm can be rewritten as

$$\begin{aligned} \Lambda_1^{j+1} &= -\Lambda_2^j + \left(\frac{1}{p_1} + \frac{1}{p_2}\right) S B_2 (\mathbf{g}_2 + R_2^{-1} B_2^T \Lambda_2^j), \\ \Lambda_2^{j+1} &= -\Lambda_1^j + \left(\frac{1}{p_1} + \frac{1}{p_2}\right) S B_1 (\mathbf{g}_1 + R_1^{-1} B_1^T \Lambda_1^j), \end{aligned} \tag{23}$$

which is a fixed-point iteration to solve the linear system

$$\begin{bmatrix} I & I - \left(\frac{1}{p_1} + \frac{1}{p_2}\right) S B_2 R_2^{-1} B_2^T \\ I - \left(\frac{1}{p_1} + \frac{1}{p_2}\right) S B_1 R_1^{-1} B_1^T & I \end{bmatrix} \begin{bmatrix} \Lambda_1 \\ \Lambda_2 \end{bmatrix} = \left(\frac{1}{p_1} + \frac{1}{p_2}\right) \begin{bmatrix} S B_2 \mathbf{g}_2 \\ S B_1 \mathbf{g}_1 \end{bmatrix}. \tag{24}$$

The linear system (24) can now be solved by a more efficient Krylov subspace method. This corresponds to use the optimized Schwarz method as a preconditioner for the Krylov subspace method.

ALGORITHM 2. [illegible] (24) [illegible]

Notice that when the GMRES method is applied to the linear system (24), each multiplication of the matrix

$$M := \begin{bmatrix} I & I - \left(\frac{1}{p_1} + \frac{1}{p_2}\right) SB_2 R_2^{-1} B_2^T \\ I - \left(\frac{1}{p_1} + \frac{1}{p_2}\right) SB_1 R_1^{-1} B_1^T & I \end{bmatrix}$$

by a vector requires the solution of two local problems of the type $R_l \mathbf{x} = \mathbf{b}$, $l = 1, 2$.

4. Numerical experiments. In this section we validate the interface conditions proposed in the previous sections by means of some numerical tests. For the implementation we have used MATLAB 6.5. We compare the performance of the two type of interface conditions for both Algorithm 1 and 2.

The computational domain throughout this section will be

$$\Omega = [0, 2] \times [0, 1] \times [0, 1],$$

in which we consider the following problem:

$$\begin{cases} \text{curl curl } \mathbf{u} - \omega^2 \mathbf{u} = \mathbf{F} & \text{in } [0,2] \times [0,1] \times [0,1], \\ \mathbf{u} \times \mathbf{n} = \mathbf{0} & \text{on } \{y = 0\},\ \{y = 1\},\ \{z = 0\},\ \{z = 1\}, \\ \text{curl } \mathbf{u} \times \mathbf{n} + i\,\omega \mathbf{u}_\tau = \mathbf{0} & \text{on } \{x = 0\},\ \{x = 2\}\,. \end{cases} \tag{25}$$

We take $\omega = 2\pi$. The function $\mathbf{F}$ is chosen such that the exact solution $\mathbf{u}$ is the gradient of the function $\Psi = x(2-x)y(1-y)z(1-z)e^{i\omega(y+z)/2}$.

We employ curl-conforming linear finite elements in a uniform hexahedral mesh and we perform the computations with the optimized parameters p^{opt}, r^{opt}, and q^{opt} as defined in the previous sections.

4.1. Influence of the mesh size. In this first series of tests we decompose Ω into two equal subdomains, $\Omega_1 = [0, 1] \times [0, 1] \times [0, 1]$ and $\Omega_2 = [1, 2] \times [0, 1] \times [0, 1]$.

In Table 1 we report the number of iterations required to achieve an increment vector (in the case of Algorithm 1) or a residual vector (in the case of Algorithm 2) of norm less than 10^{-8}. We compare the four algorithms for different mesh sizes. In Figure 2 we report the convergence histories of the four algorithms for a fixed mesh size, and in Figure 3 we report the convergence histories of each algorithm with respect to the mesh size.

The most efficient method is the Krylov one with symmetric interface condition. The speed-up of the Krylov accelerations is remarkable for both symmetric and unsymmetric interface conditions (no wonder in that).

TABLE 1
Iteration counts for different algorithms and different mesh sizes. Nonconstant initial data.

h	Iterative S	Iterative NS	Krylov S	Krylov NS
1/8	107	77	15	27
1/12	249	126	17	33
1/16	478	205	19	37
1/20	> 500	305	20	40

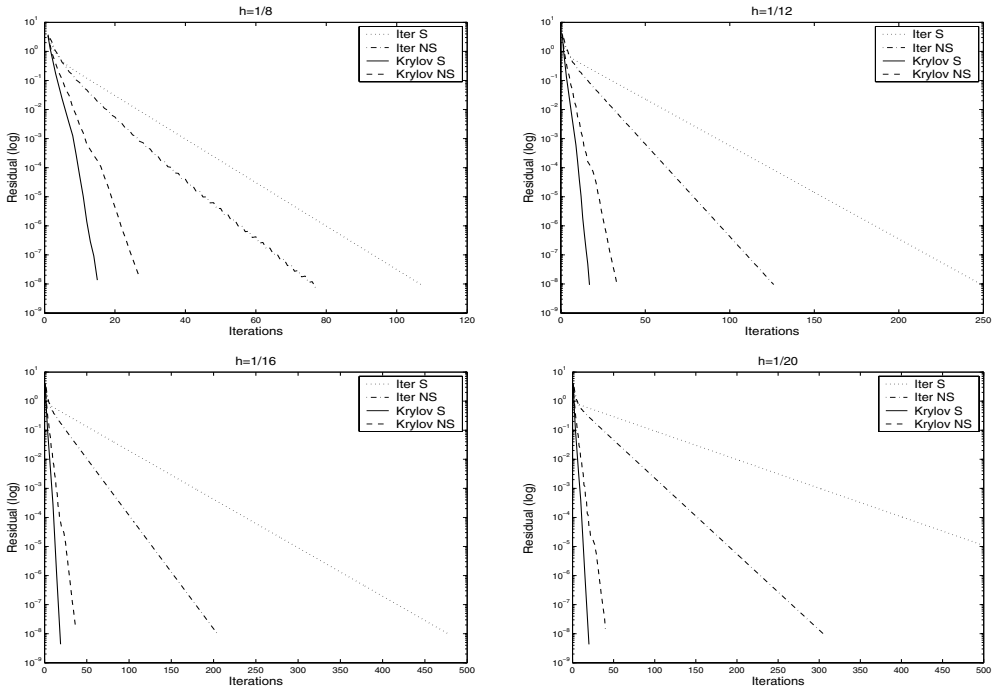


FIG. 2. *Convergence history of the different algorithms for different mesh sizes: from top left* $h = 1/8$, $h = 1/12$, $h = 1/16$, $h = 1/20$.

4.2. Influence of the position of the interface. In the previous tests the two subdomains were symmetric with respect to the interface. In the second series of tests we compare the performance of the four algorithms when the position of the interface changes. We choose $h = 1/12$, and we consider three different positions of the interface, at $x = 1/3$, $x = 2/3$, and $x = 4/3$, yielding the following decompositions:

$$\Omega_1 = [0, 1/3] \times [0, 1]^2, \qquad \Omega_2 = [1/3, 2] \times [0, 1]^2,$$

$$\Omega_1 = [0, 2/3] \times [0, 1]^2, \qquad \Omega_2 = [2/3, 2] \times [0, 1]^2,$$

and

$$\Omega_1 = [0, 4/3] \times [0, 1]^2, \qquad \Omega_2 = [4/3, 2] \times [0, 1]^2.$$

We report in Table 2 the iteration counts for the four algorithms. Only the Krylov algorithm with symmetric interface conditions shows a significative increase in the number of iterations.

5. Other interface conditions used in domain decomposition methods for the harmonic Maxwell system. In this section we give a brief survey of other interface conditions proposed in literature for the harmonic Maxwell system, always in the framework of nonoverlapping domain decomposition methods. We apply to these interface conditions the analysis we made in section 2 and we note that for all of them the iterative procedure (6) fails to converge, in the presence of evanescent modes. We will keep throughout this section the notation used in section 2.

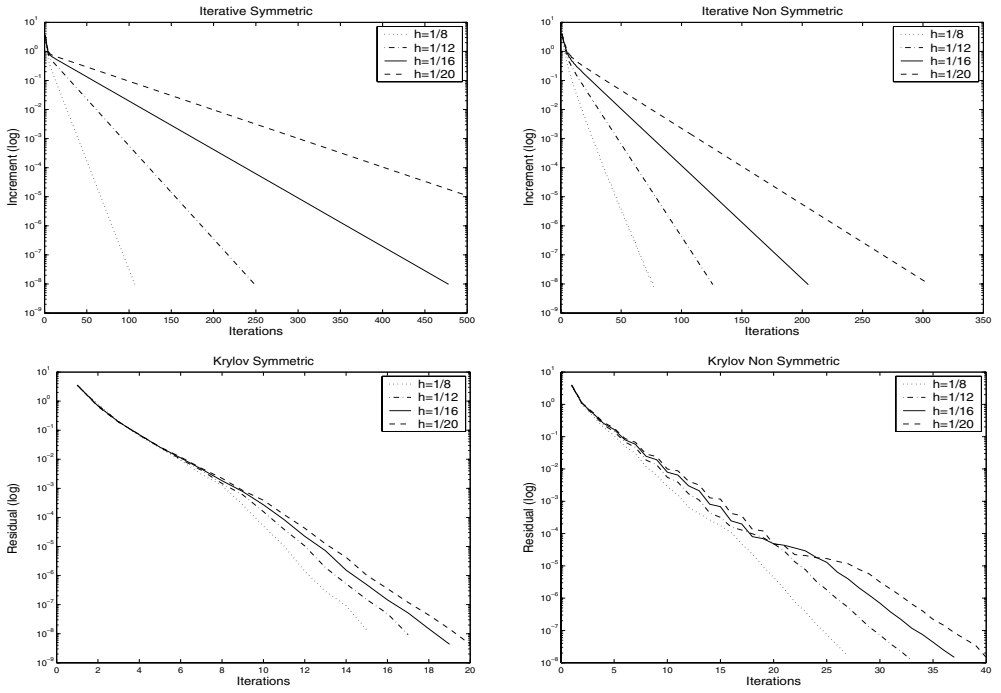


FIG. 3. *Convergence history of the different algorithms with respect to the mesh size: from top left, iterative symmetric, iterative nonsymmetric, Krylov symmetric, Krylov nonsymmetric.*

TABLE 2
Iteration counts for different position of the interface, $h = 1/12$.

Ω_1	Iterative S	Iterative NS	Krylov S	Krylov NS
$[0, 1/3] \times [0, 1]^2$	310	142	32	32
$[0, 2/3] \times [0, 1]^2$	260	126	32	32
$[0, 1] \times [0, 1]^2$	249	126	17	33
$[0, 4/3] \times [0, 1]^2$	260	135	32	32

Despres–Joly–Roberts [4]. The earliest nonoverlapping domain decomposition algorithm for Maxwell system was proposed in [4], where the interface operator is given by

$$S_B = S_T = i\omega I. \tag{26}$$

In [4], the authors, considering a decomposition into an arbitrary number of subdomains and a boundary condition of radiative type at finite distance, show that the proposed algorithm converges weakly in $H(\mathrm{curl}, \Omega_j)$ $(j = 1, \ldots, N)$ to the solution of the single domain problem and that $\mathbf{u}_j^n$ and curl $\mathbf{u}_j^n \times \mathbf{n}$ converge to $\mathbf{u}_j$ and curl $\mathbf{u}_j \times \mathbf{n}$, respectively. The convergence properties of this algorithm depend heavily on the boundary condition, as it allows control of the energy on the outer boundaries by means of the energy on the inner ones.

For the interface operator (26) the Fourier analysis shows that the iterative procedure (6) converges only for propagative modes, while for the evanescent ones the convergence rate is exactly 1.

In fact, in this case we have $M_B = M_T = N^2$, where

$$N = \frac{1}{(\lambda + i\omega)^2} \begin{bmatrix} k_1^2 - k_2^2 & 2k_1k_2 \\ 2k_1k_2 & k_2^2 - k_1^2 \end{bmatrix},$$

and the eigenvalues of N are

$$\mu^{\pm} = \pm \frac{|k|^2}{(\lambda + i\omega)^2}.$$

If $|k|^2 < \omega^2$, then $\lambda = i\sqrt{\omega^2 - |k|^2}$ and

$$|\mu^{\pm}| = \frac{|k|^2}{(\sqrt{\omega^2 - |k|^2} + \omega)^2} < \frac{\omega^2}{(\sqrt{\omega^2 - |k|^2} + \omega)^2} < 1.$$

On the other hand, if $|k|^2 > \omega^2$, then λ is real and

$$|\mu^{\pm}| = \frac{|k|^2}{\lambda^2 + \omega^2} = 1.$$

Thus, the algorithm (4) with the interface conditions proposed in [4] does not converge for evanescent modes. However, the boundary condition (a radiation at finite distance) considered by the authors does not allow evenescent modes to come into the system: thus, the above result is not in contradiction with their work.

Chevalier [2]. To improve the convergence of the algorithm introduced in [4], Chevalier proposed in his thesis [2] the use of the interface operator

$$S_B = S_T = i\omega I + \zeta \overrightarrow{\text{curl}}_\Gamma \text{curl}_\Gamma, \tag{27}$$

$\zeta \in \mathbb{C}$ being a complex parameter to be tuned, which consists in adding to the previous interface conditions a space derivative of second order in the direction tangential to the interface, following what is done in the literature for Helmholtz equations (see, for instance, [2] and [5]).

In [2] there is neither an accurate analysis of the algorithm nor a clear tuning of the parameter ζ, and we show here that, different from the case of the Helmholtz equation, the use of the second order transmission condition does not guarantee convergence for evanescent modes. In fact, we have again $M_B = M_T = N^2$ with

$$\begin{aligned} N &= [R - \omega^2 I + \lambda(\zeta R + i\omega I)]^{-1}[-R + \omega^2 I + \lambda(\zeta R + i\omega I)] \\ &= [(\lambda\zeta + 1)R + (i\lambda\omega - \omega^2)I]^{-1}[(\lambda\zeta - 1)R + (i\lambda\omega + \omega^2)I]. \end{aligned}$$

If $\lambda\zeta \neq \pm 1$,

$$= \frac{\lambda\zeta - 1}{\lambda\zeta + 1} \left[R + \frac{i\lambda\omega - \omega^2}{\lambda\zeta + 1} I\right]^{-1} \left[R + \frac{i\lambda\omega + \omega^2}{\lambda\zeta - 1} I\right].$$

Setting $a = \frac{i\lambda\omega - \omega^2}{\lambda\zeta + 1}$ and $b = \frac{i\lambda\omega + \omega^2}{\lambda\zeta - 1}$,

$$\begin{aligned} N &= \frac{\lambda\zeta - 1}{\lambda\zeta + 1}(R + aI)^{-1}(R + bI) \\ &= \frac{\lambda\zeta - 1}{\lambda\zeta + 1} \frac{1}{a|k|^2 + a^2} \begin{bmatrix} k_1^2 + a & k_1k_2 \\ k_1k_2 & k_2^2 + a \end{bmatrix} \begin{bmatrix} k_2^2 + b & -k_1k_2 \\ -k_1k_2 & k_1^2 + b \end{bmatrix} \\ &= \frac{\lambda\zeta - 1}{\lambda\zeta + 1} \frac{1}{a|k|^2 + a^2} \begin{bmatrix} k_1^2 b + k_2^2 a + ab & -k_1k_2(a - b) \\ -k_1k_2(a - b) & k_1^2 a + k_2^2 b + ab \end{bmatrix}. \end{aligned}$$

The eigenvalues of the matrix

$$\begin{bmatrix} k_1^2 b + k_2^2 a + ab & -k_1 k_2 (a-b) \\ -k_1 k_2 (a-b) & k_1^2 a + k_2^2 b + ab \end{bmatrix}$$

are $\gamma^+ = a(|k|^2 + b)$ and $\gamma^- = b(|k|^2 + a)$; hence the eigenvalues of N are

$$\mu^+ = \frac{\lambda\zeta - 1}{\lambda\zeta + 1} \frac{|k|^2 + b}{|k|^2 + a}, \quad \mu^- = \frac{\lambda\zeta - 1}{\lambda\zeta + 1} \frac{b}{a} = \frac{i\lambda + \omega}{i\lambda - \omega}.$$

It is now easy to observe that when λ is real $|\mu^-| = 1$, and the algorithm does not converge for evanescent modes.

Collino–Delbue–Joly–Piacentini [3]. The Fourier analysis in section 2 (Lemma 1 and Remark 1) shows that the operator linking $\mathbf{n} \times \mathbf{u} \times \mathbf{n}$ to curl $\mathbf{u} \times \mathbf{n}$ is

$$\mathcal{T}(\text{curl } \mathbf{u} \times \mathbf{n}) + (\omega^2 - \overrightarrow{\text{curl}}_\Gamma \text{curl}_\Gamma)\, \mathbf{n} \times \mathbf{u} \times \mathbf{n} = 0, \tag{28}$$

where $\mathcal{T}$ is a pseudodifferential operator corresponding, in the Fourier space, to the multiplication by $\lambda(|k|) = \sqrt{|k|^2 - \omega^2}$: due to its nonlocality, this operator cannot be used in practice.

Another interface condition, based on a zeroth order Taylor expansion of the function $\lambda(|k|)$ in the neighborhood of $|k| = 0$, has then been proposed in [3], where the interface operators are given by

$$S_B = S_T = -i\omega\beta \left(I + \frac{\alpha}{\omega^2} \overrightarrow{\text{curl}}_\Gamma \text{curl}_\Gamma \right). \tag{29}$$

Taking $\alpha = -1$ this interface condition coincides with our symmetric interface conditions for $p = i\frac{\beta}{\omega}$. If $\beta \in \mathbb{R}$, then $\mathcal{R}e\, p = 0$ and we have from section 2.1 that $\rho = 1$ for evanescent modes.

Notice, however, that the authors assume $\sqrt{\omega^2 - |k|^2} \sim \omega$, which amounts to taking into account, at once, only propagative modes, and, among them, only the ones with very small space frequency. Our approach differs a fortiori from this latter one, since we do not impose any restriction on the spatial frequency $|k|^2$.

Numerical experiments. In Figure 4 we compare the convergence history of the iterative procedure with the following interface conditions: optimized symmetric (i.e., (8) with $p_T = p_B$ given by (10)), optimized unsymmetric (i.e., (8) with $p_B = r$ and $p_T = iq$, r given in (17) and q given in (18)), the interface condition (26), and the one introduced by Chevalier (27) with $\zeta = i/2\omega$. We apply them to the model problem (25) and also to a similar one with a radiation condition on the whole boundary

$$\begin{cases} \text{curl curl } \mathbf{u} - \omega^2 \mathbf{u} = \mathbf{F} & \text{in } \Omega = [0,2] \times [0,1] \times [0,1], \\ \text{curl } \mathbf{u} \times \mathbf{n} + i\,\omega \mathbf{u}_\tau = \mathbf{0} & \text{on } \partial\Omega\,. \end{cases} \tag{30}$$

Also in this case the data $\mathbf{F}$ is chosen such that the exact solution is $\mathbf{u} = \nabla[x(2-x)y(1-y)z(1-z)e^{i\omega(y+z)/2}]$.

We plot the norm of the increment for the first 100 iterations since its norm is greater that 10^{-8}. For the model problem (25) the interface conditions (26) and (27) do not converge. (Note that the boundary condition in (25) differs from the one considered in [4].) For the model problem (30) the convergence of the four algorithms

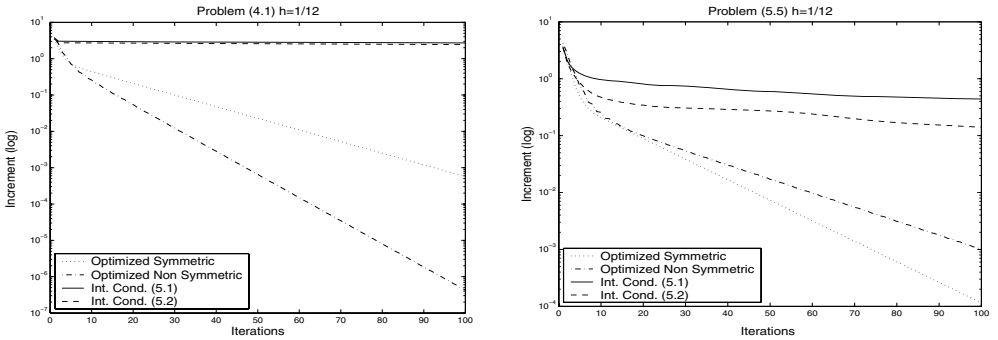


FIG. 4. *Convergence history for different interface conditions. Problem* (25) *(left) and problem* (30) *(right): iterative algorithm.*

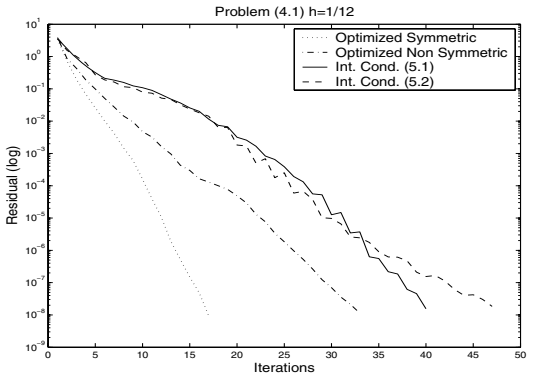


FIG. 5. *Convergence history for different interface conditions, Problem* (25)*: Krylov algorithm.*

is very slow. In this case the fastest one is the one that uses the optimized unsymmetric interface conditions.

Finally, in Figure 5 we compare the convergence histories for the different interface conditions when the Krylov algorithm is applied to problem (25). Notice that neither of the interface conditions (26) and (27) has been previously used in such a framework. All algorithms converge, and the ones with optimized conditions are faster. In this case the fastest one is the one that uses the optimized symmetric interface conditions.

6. Conclusions. We proposed and analyzed two families of second order interface conditions to be used in the framework of nonoverlapping domain decomposition methods for the time-harmonic Maxwell equations. These conditions have been derived by means of a Fourier analysis: the first family depends on a complex parameter, while the second family depends on two real parameters. These parameters have been chosen to optimize the convergence rate of the method. The resulting interface operators have been applied to two different algorithms in the case of a two-domain decomposition: the first algorithm is an iterative procedure of fixed point type and the second one is a Krylov subspace method. The numerical solution of some test problems has been addressed to validate the two algorithms: in both cases we obtained a convergent method. In particular, the Krylov subspace procedure, based on GMRES iterations, appears to be very efficient. Finally, we compared these optimized

interface conditions with other conditions proposed in literature: evidence is that the optimized conditions provide more efficient and robust algorithms.

REFERENCES

[1] A. Bossavit, *Computational Electromagnetism. Variational Formulation, Complementarity, Edge Elements*, Electromagnetism Series 2, Academic Press, San Diego, CA, 1998.
[2] P. Chevalier, *Méthodes numériques pour les tubes hyperfréquences. Résolution par décomposition de domaine*, Ph.D. thesis, Université Paris VI, 1998.
[3] P. Collino, G. Delbue, P. Joly, and A. Piacentini, *A new interface condition in the non-overlapping domain decomposition for the Maxwell equations*, Comput. Methods Appl. Mech. Engrg., 148 (1997), pp. 195–207.
[4] B. Després, P. Joly, and J. E. Roberts, *A domain decomposition method for the harmonic Maxwell equations*, in Iterative Methods in Linear Algebra, North–Holland, Amsterdam, 1992, pp. 475–484.
[5] M. J. Gander, F. Magoulès, and F. Nataf, *Optimized schwarz methods without overlap for the Helmholtz equation*, SIAM J. Sci. Comput., 24 (2002), pp. 38–60.
[6] P. Monk, *Finite Element Methods for Maxwell's Equations*, Oxford University Press, Oxford, UK, 2003.
[7] J. Nédélec, *Mixed finite elements in* R^3, Numer. Math., 35 (1980), pp. 315–341.
[8] J. Nédélec, *Eléments finis mixtes incompressibles pour l'équation des Stokes dans* $\mathbb{R}^3$, Numer. Math., 39 (1982), pp. 97–112.
[9] A. Quarteroni and A. Valli, *Domain Decompostion Methods for Partial Differential Equations*, Oxford University Press, Oxford, UK, 1999.

SIAM J. SCI. COMPUT.
Vol. 28, No. 1, pp. 123–140
© 2006 Society for Industrial and Applied Mathematics

A CLASS OF EXTENDED LIMITERS APPLIED TO PIECEWISE HYPERBOLIC METHODS*

SUSANA SERNA†

Abstract. Piecewise hyperbolic methods (PHMs) have been widely used for the approximation of hyperbolic conservation laws. In this paper we have studied the "power limiters" in order to analyze their suitable use in the design of PHMs. We formulate a new PHM, based on these limiters, improving the behavior of the PHM in [SIAM J. Sci. Comput., 15 (1994), pp. 892–915], at local extrema, jump discontinuities, and discontinuities in derivative. We show that this new hyperbolic reconstruction (we call it PowerPHM) is local total variation bounded and it behaves stably with a reduced numerical viscosity compared to PHM. We show numerical evidence of the above mentioned features by means of one- and two-dimensional numerical tests with the Euler equations of gas dynamics.

Key words. conservation laws, shock capturing schemes, high order accuracy, piecewise hyperbolic reconstructions, limiters, local total variation bounded

AMS subject classifications. 65M06, 35L65, 76M20

DOI. 10.1137/040611811

1. Introduction.

Our general goal consists in obtaining high order accurate numerical approximations to piecewise smooth real functions of a real variable with a finite number of jump discontinuities to approximate the solution of hyperbolic systems of conservation laws.

Our discussion here will focus on a fixed uniform grid and a fixed degree of local accuracy, trying to minimize the spurious information generated when dealing simultaneously with coarse and fine scales.

A first order approximation to a piecewise smooth function in a fixed grid can be defined as a piecewise constant function, with jump discontinuities located at the cell interfaces [4]. Van Leer considered second order approximations defined as piecewise linear functions, introducing for the first time the minmod limiter function applied on neighboring slopes to avoid the Gibbs phenomena near discontinuities when computing approximate solutions to hyperbolic conservation laws [21, 22, 10]. Total variation diminishing (TVD) schemes were introduced in [6] to get high order accurate nonoscillatory approximations of the solution of hyperbolic conservation laws, but the main drawback of these schemes was the degeneration to first order of accuracy at local extrema [13]. Essentially nonoscillatory (ENO) methods were introduced to get high order accurate reconstruction procedures of order larger than two, avoiding Gibbs phenomena and spurious oscillations, up to local truncation errors [7]. ENO procedures use the smoothest polynomial interpolation by means of choosing the divided differences of smallest size following a tree-like algorithm. The selection procedure of the ENO methods is a limiter function acting on the successive divided differences of the data. One of the important issues of the high order accurate reconstruction procedures is the size of the stencil, i.e., the number of data points needed to perform

*Received by the editors July 16, 2004; accepted for publication (in revised form) August 29, 2005; published electronically March 3, 2006. This work was supported by DGICYT projects BFM2001-2814 and MTM2005-07708.

http://www.siam.org/journals/sisc/28-1/61181.html

†Departamento de Matematica Aplicada, University of Valencia, C/Dr. Moliner, 50, Burjassot-Valencia, Valencia 46100, Spain (susana.serna@uv.es).

the approximation. If the stencil is small the behavior of the procedure is robust when used in evolution problems where discontinuities are either moving or interacting, but the degree of accuracy is limited. By using a limiter function we may design reconstruction procedures with an optimal size of the stencil and good resolution of discontinuities.

The analysis and research of new classes of limiters has been a fruitful research field in investigating how to modify the Lax–Wendroff scheme to get useful, simple, second order in space and time, and one time stepping schemes. Limiters for this purpose have been introduced by van Leer, Roe, and other authors (see [6, 14, 21, 22, 10]). Our idea to explore new classes of limiters has been applied to methods of order larger than two.

In [17] a new class of limiters, the so-called power limiters, were introduced to be applied on neighboring second order differences to define new five point stencil ENO parabolic reconstruction procedures. The resulting third order PowerENO method and its fifth order accurate sequel Weighted PowerENO method showed better capability to resolve discontinuities and local extrema than ENO methods when combined with shock capturing schemes for the approximation of hyperbolic conservation laws.

In [11] a third order accurate piecewise hyperbolic procedure using three data points was proposed as a compact and robust scheme. The piecewise hyperbolic method (PHM) was designed using the harmonic mean as the basic average to limit the gradient of the local reconstruction. The piecewise hyperbolic reconstruction procedure works robustly combined with shock capturing schemes [2, 3, 11, 12, 19], relaxation schemes [1, 15], and moving mesh methods [9]. This good behavior is mainly due to the following:

1. Reduced stencil.
2. Hyperbolas are less oscillatory than parabolas.
3. The use of a limiter based on the harmonic mean.

A well-known drawback of the PHM reconstruction procedure is the loss of total variation at local extrema as a consequence of the strong effect of using the harmonic mean as a slope limiter.

In this research work we analyze the use of power limiters as slope limiters for the design of three point stencil piecewise hyperbolic reconstructions. To this end we need to perform a new analytical study of power limiters to know the qualitative properties required for a power limiter to define correctly a hyperbolic approximation with local variation bounded. As a result of this study we propose a new piecewise hyperbolic reconstruction procedure based on the power limiter that accomplishes the required conditions. The new PowerPHM procedure shares with the PHM the mentioned good properties, prescribes more total variation in cell, and improves the resolution of local extrema, jump discontinuities, and discontinuities in derivative in comparison with the PHM. We have presented a PowerPHM shock capturing scheme to approximate the solution of compressible fluid flow. We show evidence of the robustness and accuracy of our new scheme through a set of numerical experiments that include the evolution and interaction of strong shocks in one and two dimensions.

The paper is organized as follows. In section 2 we introduce the necessary notation and we review the basic facts on piecewise hyperbolic reconstructions. In section 3 we present a new analytical study of the power limiters, establishing the necessary conditions to obtain well-behaved hyperbolic reconstructions. Then we define the local hyperbolic power reconstruction and we show that it is well defined with local total variation bounded (LTVB). In section 4, we present a set of numerical tests showing numerical evidence of the features analyzed in section 3. We draw our conclusions in section 5.

2. Piecewise hyperbolic reconstruction. In this section we briefly review a number of basic constructions and analytical results on PHMs that we use later in section 3 in the development and analysis of our new reconstruction procedure based on power limiters. We analyze how oscillatory a hyperbolic reconstruction is compared to a parabolic one in terms of the local variation in cell.

In order to settle our notation and computational framework, we consider numerical approximations to the scalar initial value problem

$$u_t + f(u)_x = 0, \tag{2.1}$$

$$u(x,t) = u_0(x), \tag{2.2}$$

where u_0 is a periodic or compactly supported piecewise smooth function.

We consider the following computational grid: $x_j = jh$ (h being the spatial step), $t_n = n\Delta t$ is the time discretization (with Δt the time step), and $I_j = [x_{j-\frac{1}{2}}, x_{j+\frac{1}{2}}]$ is the spatial cell, where $x_{j+\frac{1}{2}} = x_j + \frac{h}{2}$ is the cell interface and $C_j^n = [x_{j-\frac{1}{2}}, x_{j+\frac{1}{2}}] \times [t_n, t_{n+1}]$ is the computational cell. Let u_j^n be an approximation of the mean value in I_j, $\frac{1}{h}\int_{x_{j-\frac{1}{2}}}^{x_{j+\frac{1}{2}}} u(x,t_n)dx$, of the exact solution $u(x,t_n)$ of the initial value problem (2.1) and (2.2), obtained from a finite volume scheme in conservation form:

$$u_j^{n+1} = u_j^n - \frac{\Delta t}{h}(\tilde{f}_{j+\frac{1}{2}} - \tilde{f}_{j-\frac{1}{2}}), \tag{2.3}$$

where the numerical flux, $\tilde{f}$, is a function of $k+l$ variables

$$\tilde{f}_{j+\frac{1}{2}} = \tilde{f}(u_{j-k+1}^n, \ldots, u_{j+l}^n), \tag{2.4}$$

which is consistent with the flux of (2.1),

$$\tilde{f}(u, \ldots, u) = f(u). \tag{2.5}$$

We will use the above basic framework to apply high order accurate reconstruction procedures in numerical tests. We want to find numerical approximations of a piecewise smooth function $g(x)$ from its cell averages, given for the cells of the above defined uniform grid of stepsize $h > 0$, $x_j = jh$,

$$v_j = \frac{1}{h}\int_{x_{j-\frac{1}{2}}}^{x_{j+\frac{1}{2}}} g(x,t_n)dx. \tag{2.6}$$

The specific uniform grid considered is the scale of computation chosen for our approximation. The idea is to capture subscales of the original chosen scale by means of a piecewise smooth function, R, such that the restriction to each computational cell, $[x_{j-\frac{1}{2}}, x_{j+\frac{1}{2}}]$, is a suitable elementary function, R_j, such that

$$v_j = \frac{1}{h}\int_{x_{j-\frac{1}{2}}}^{x_{j+\frac{1}{2}}} R_j(x,t_n)dx \tag{2.7}$$

and it approximates g up to a degree of accuracy.

We are going to analyze the behavior of R in terms of h, when h tends to zero. This behavior might be undesirable around jump discontinuities of g since the function R may develop either Gibbs phenomena or Runge oscillatory patterns. On the other hand, the order of accuracy may degenerate through smooth regions of g due to the presence of the above singularities. The mentioned computational problems are always present when using the described framework to approximate the solution of hyperbolic conservation laws, since those evolution equations may develop discontinuities (shocks) in finite time.

We focus on third order accurate piecewise smooth functions, R, such that each R_j is determined from condition (2.7) and two more conditions, usually obtained from divided differences of v_j. We denote

$$d_{j+\frac{1}{2}} = \frac{v_{j+1} - v_j}{h}, \tag{2.8}$$

$$d_j = \frac{d_{j+\frac{1}{2}} + d_{j-\frac{1}{2}}}{2}, \tag{2.9}$$

$$D_j = d_{j+\frac{1}{2}} - d_{j-\frac{1}{2}}, \tag{2.10}$$

$$\alpha_j = \frac{D_j}{2d_j}. \tag{2.11}$$

If we know the first derivatives of g at the cell interfaces, i.e., $g'(x_{j+\frac{1}{2}})$, we can use the following two interpolating conditions to ensure third order accuracy in smooth regions:

$$R'_j\left(x_j - \frac{h}{2}\right) = g'(x_{j-\frac{1}{2}}), \tag{2.12}$$

$$R'_j\left(x_j + \frac{h}{2}\right) = g'(x_{j+\frac{1}{2}}). \tag{2.13}$$

We use the first order divided differences $d_{j-\frac{1}{2}}$ and $d_{j+\frac{1}{2}}$ instead of $g'(x_{j-\frac{1}{2}})$ and $g'(x_{j+\frac{1}{2}})$, respectively, if the last values are not available. If we consider a local parabola of the form

$$p_j(x) = a_j + b_j(x - x_j) + \frac{c_j}{2}(x - x_j)^2, \tag{2.14}$$

there is a unique parabola such that conditions (2.7), (2.12), and (2.13) are satisfied; that is, a_j, b_j, and c_j are uniquely defined. Following [11], if we consider hyperbolas of the form

$$r_j = e_j + \frac{\lambda_j}{(x - x_j) + f_j}, \tag{2.15}$$

we can compute a unique hyperbola satisfying (2.7), (2.12), and (2.13) (λ_j, e_j and f_j are uniquely defined), assuming that $g'(x_{j-\frac{1}{2}}) \cdot g'(x_{j+\frac{1}{2}}) > 0$, since hyperbolas are monotone functions.

We want to study how oscillatory the approximation R of the function $g(x)$ is in terms of h. To the best of our knowledge, the total variation of R in each cell is a measure of how oscillatory our approximation is [6]. In order to get a coarse comparison between the parabolic and the hyperbolic reconstructions we may assume for the sake of simplicity the following notation. We consider the cell for $j = 0$, $[x_{-\frac{1}{2}}, x_{\frac{1}{2}}]$, and we use $d_l := d_{-\frac{1}{2}}$ and $d_r := d_{\frac{1}{2}}$.

If $d_l \cdot d_r > 0$, the total variation of the parabola p_0 (assuming p_0 is monotone) is

$$TV(p_0) = h\frac{|d_l| + |d_r|}{2} \tag{2.16}$$

and the total variation of the hyperbola r_0 is (see [11])

$$TV(r_0) = h\sqrt{|d_l| \cdot |d_r|}. \tag{2.17}$$

From the arithmetic-geometric inequality we conclude that the parabolic reconstruction prescribes more total variation than the hyperbolic one. Let us remark that formulas (2.16) and (2.17) are the product of h times an average of two positive numbers. If g is smooth on a region containing our computational cell $[-\frac{h}{2}, \frac{h}{2}]$, then d_l and d_r are bounded and therefore, $TV(p_0) = O(h)$ and $TV(r_0) = O(h)$. Thus, p_0 and r_0 are smooth enough to be nonoscillatory and those quantities tend to zero with h.

On the other hand, if g has a jump discontinuity at $x_0 = 0$, either d_l or d_r is $O(\frac{1}{h})$, and

$$TV(p_0) = O(1) \tag{2.18}$$

and

$$TV(r_0) = O\big(\sqrt{h}\big). \tag{2.19}$$

Thus, in the presence of discontinuities, hyperbolas are much less oscillatory than parabolas. One way to make a piecewise smooth approximation less oscillatory is by using slope limiters to preprocess first order derivatives in a way that the order of accuracy does not degenerate.

We can assert that hyperbolas have a natural mechanism to limit the gradient by means of an average. In fact, the first derivative of r_0 at $x_0 = 0$ is, from [11],

$$r_0'(x_0) = \operatorname{sgn}(d_l)\left(\frac{2\sqrt{|d_l|}\sqrt{|d_r|}}{\sqrt{|d_l|} + \sqrt{|d_r|}}\right)^2, \tag{2.20}$$

which is much smaller in size than the one of the parabola,

$$p_0'(x_0) = \operatorname{sgn}(d_l)\frac{|d_l| + |d_r|}{2}, \tag{2.21}$$

assuming $d_l \cdot d_r > 0$, (see Hardy, Littlewood, and Polya [5]).

Applying Marquina [11, p. 899] we can write any hyperbola $r_0(x)$ in terms of any d_0 and α (for $-2 < \alpha < 2$), and satisfying the conservation property (2.7) and $r_0'(x_0) = d_0$, in the form

$$r_0(x) = v_0 + d_0 h \frac{1}{\alpha^2}\left[\log\left(\frac{2-\alpha}{2+\alpha}\right) - \frac{\alpha h}{\alpha(x-x_0)-h}\right]. \tag{2.22}$$

An appropriate choice of d_0 and α gives a hyperbola that interpolates one of the lateral derivatives.

Two cases are relevant for this purpose:

(a) If $d_0 \cdot d_l > 0$ then $\alpha = 2(\sqrt{\frac{d_0}{d_l}} - 1)$ if and only if $d_l = r'(x_0 - \frac{h}{2})$.

(b) If $d_0 \cdot d_r > 0$ then $\alpha = 2(1 - \sqrt{\frac{d_0}{d_r}})$ if and only if $d_r = r'(x_0 + \frac{h}{2})$.

Another important issue for piecewise hyperbolic reconstructions is to control the total variation growth. The LTVB condition was introduced in [11] for this purpose.

DEFINITION 2.1. [illegible] R [illegible] $M > 0$ [illegible] h [illegible] j

$$TV(R_j) \le Mh. \tag{2.23}$$

To ensure that a hyperbolic reconstruction is LTVB, it is sufficient to restrict the range of values of α to an interval $[-K, K]$ such that $0 < K < 2$.

Marquina in [11] proposed an appropriate choice of d_0 to accomplish the above property. Indeed, the local hyperbolic harmonic reconstruction (LHHR) designed in [11] prescribes, in each computational cell, a hyperbola r_j that satisfies (2.7), interpolates the lateral derivative with the smallest absolute value, and assigns, as the central derivative (d_0), the harmonic mean of both lateral derivatives instead of (2.20). In this case, the range of values for α, independent of h, is confined to

$$-2(\sqrt{2}-1) < \alpha < 2(\sqrt{2}-1). \tag{2.24}$$

This inequality is essential to prove that the LHHR is LTVB [11].

The main drawback of the LHHR is the loss of accuracy at local extrema and, as a consequence, the loss of total variation at those points.

One way to increase locally the total variation of the reconstruction is to consider another choice of d_0 allowing a wider range of α values than the one prescribed by the d_0 of the LHHR in (2.24).

Next, we are going to analyze a class of slope limiters suitable for hyperbolic reconstructions that improves the behavior of the LHHR and satisfies the LTVB condition.

3. Power limiters and piecewise hyperbolic power reconstruction. In this section we present the main analytical results of our paper. After reviewing the definition and basic properties of the power limiters we examine the qualitative properties of the "power means" related to the classical means, appearing in hyperbolic reconstructions to design well-behaved PHM based on power limiters. Then, we introduce the PowerPHM reconstruction procedure showing that it is well defined and it is LTVB.

A limiter might be considered as a function of two or more variables based on an average of nonnegative numbers. As was pointed out in [17], we can define the most usual limiters in terms of classical averages.

The minmod, ENO, and harmonic limiters are based on classical means between two nonnegative numbers. In fact, if x and y are real numbers,

$$\text{minmod}(x,y) = \frac{(\text{sgn}(x)+\text{sgn}(y))}{2}\ \min(|x|,|y|), \tag{3.1}$$

$$\text{mineno}(x,y) = \text{minsign}(x,y)\ \min(|x|,|y|), \tag{3.2}$$

$$\text{harmod}(x,y) = \frac{(\text{sgn}(x)+\text{sgn}(y))}{2}\ \frac{2|x||y|}{|x|+|y|}, \tag{3.3}$$

$$\text{hareno}(x,y) = \text{minsign}(x,y)\ \frac{2|x||y|}{|x|+|y|}, \tag{3.4}$$

where $\text{sgn}(x)$ is the sign function, and

$$\text{minsign}(x,y) = \begin{cases} \text{sgn}(x), & |x| <= |y|, \\ \text{sgn}(y) & \text{otherwise.} \end{cases}$$

In [17], an extended class of limiters was introduced in order to design improved ENO reconstructions, including as particular cases the above mentioned limiters. The $\text{power}_p(x,y)$ mean for $x>0$, $y>0$, and p a positive integer is defined in [17] as

$$\text{power}_p(x,y) = \frac{(x+y)}{2}\ \left(1-\left|\frac{x-y}{x+y}\right|^p\right). \tag{3.5}$$

The corresponding limiters are defined for any x and y as

$$\text{powermod}_p(x,y) = \frac{(\text{sgn}(x)+\text{sgn}(y))}{2}\,\text{power}_p(|x|,|y|), \tag{3.6}$$

$$\text{powereno}_p(x,y) = \text{minsign}(x,y)\,\text{power}_p(|x|,|y|). \tag{3.7}$$

The following inequalities are satisfied for any $x>0$ and $y>0$:

$$\min(x,y) \le \text{power}_p(x,y) \le \text{power}_q(x,y) \le \frac{x+y}{2}$$

for $0<p<q$, and

$$\text{power}_1(x,y) = \min(x,y), \tag{3.8}$$

$$\text{power}_2(x,y) = \frac{2xy}{x+y}. \tag{3.9}$$

The power_3 mean can be written in terms of the min and max functions for $x\ge 0$ and $y\ge 0$, not simultaneously zero [17]:

$$\text{power}_3(x,y) = \min(x,y)\frac{x^2+y^2+2(\max(x,y))^2}{(x+y)^2}. \tag{3.10}$$

Our goal is to improve the resulting effect of applying the harmonic mean on lateral derivatives in the LHHR relaxing this limiting effect by using power_3 mean instead and, likewise, ensuring the existence of a positive number β satisfying $-2\beta \leq \alpha \leq 2\beta$ and $\sqrt{2}-1 < \beta < 1$, thus allowing more local variation in the cell.

To this end we have to compare the power_3 mean with the harmonic one and the natural mean for the hyperbola (2.20).

We use the following notation to settle the next propositions. If $x > 0$ and $y > 0$ then $\mathcal{M}(x,y) = \min(x,y)$, $\mathcal{A}(x,y) = \frac{x+y}{2}$, $\mathcal{G}(x,y) = \sqrt{xy}$, $\mathcal{H}(x,y) = \frac{2xy}{x+y}$, and $\mathcal{H}_4 = (\frac{2\sqrt{x}\sqrt{y}}{\sqrt{x}+\sqrt{y}})^2$. Let us remark that $\mathcal{H}_4$ appears in the expression (2.20) as the natural derivative of the hyperbola at the center.

In order to be more concise we will skip the dependence on x and y. We denote $\mathcal{P}_j = \text{power}_j(x,y)$.

Let us compare these averages.

PROPOSITION 3.1.

. . $x \geq 0$. , $y \geq 0$ . . . . $r = \frac{\mathcal{G}}{\mathcal{A}}$. , $R = |\frac{x-y}{x+y}|$ . . .

1. $R^2 + r^2 = 1$
2. $\mathcal{H} = \mathcal{A}(1-R^2)$
3. $\mathcal{H}_4 = \mathcal{H}(1+R^2\frac{1}{(1+r)^2})$
4. $\mathcal{P}_3 = \mathcal{H}(1+R^2\frac{1}{1+R})$
5. $\mathcal{P}_4 = \mathcal{H}(1+R^2)$
6. $\mathcal{H} \leq \mathcal{P}_3 \leq \mathcal{P}_4$
7. $\mathcal{H} \leq \mathcal{H}_4 \leq \mathcal{P}_4$

. . . . The proof is straightforward. □

PROPOSITION 3.2. . . . . $\mathcal{P}_3$. , $\mathcal{H}_4$ . . . .

Since $\mathcal{R}_3$ is a strictly decreasing function of λ in $[0,1]$ and $\mathcal{R}_3(0) = 3$, we have that $\mathcal{R}_3 \leq 3$. □

Following [17], we define $\text{powereno}(x,y) = \text{powereno}_3(x,y)$. We use this limiter for the design of the local hyperbolic power reconstruction.

The following algorithm defines a hyperbola in terms of d_0 and α according to (2.22).

LOCAL HYPERBOLIC POWER RECONSTRUCTION (LHPR) METHOD

Let define $tol = h^2$,

if $(|d_l| \leq tol)$ and $(|d_r| \leq tol)$ **then**

$d_0 = 0$ and $\alpha = 0$

else

$d_0 = \text{powereno}(d_l, d_r) = \text{minsign}(dl, dr)\min(|dl|, |dr|)\frac{dl^2+dr^2+2(\max(|dl|,|dr|))^2}{(|dl|+|dr|)^2}$

if $|d_l| \leq |d_r|$ **then**

$ratio = \frac{d_l^2+3d_r^2}{(|d_l|+|d_r|)^2}$ $(= \mathcal{R}_3)$

$\alpha = 2(\sqrt{ratio} - 1)$

else

$ratio = \frac{3d_l^2+d_r^2}{(|d_l|+|d_r|)^2}$ $(= \mathcal{R}_3)$

$\alpha = 2(1 - \sqrt{ratio})$

end

end

The following lemma shows that this algorithm gives a well-defined hyperbola.

LEMMA 3.4. [illegible] α [illegible]

$$-2(\sqrt{3} - 1) < \alpha < 2(\sqrt{3} - 1). \tag{3.12}$$

[illegible]. The proof follows easily from Lemma 3.3. □

Thus, we get the LTVB property for the LHPR method.

THEOREM 3.5. [illegible]

[illegible]. We can find a constant $M > 0$ such that for all j except for a finite number of isolated j's for which $d_{j+\frac{1}{2}} = O(\frac{1}{h})$, $|d_{j+\frac{1}{2}}| \leq M$.

If $|d_{j-\frac{1}{2}}| \leq |d_{j+\frac{1}{2}}|$, $d_{l_j} = d_{j-\frac{1}{2}}$ and

$$d_{r_j} = d_{j-\frac{1}{2}}\Big(\frac{2}{2-\alpha}\Big)^2. \tag{3.13}$$

Due to Lemma 3.4, we have that

$$TV(r_j) \leq (2 + \sqrt{3}) \cdot M \cdot h. \qquad \square$$

To illustrate our study we present in Figure 3.1 the harmonic piecewise hyperbolic, the power piecewise hyperbolic, and PowerENO parabolic [17] static reconstructions computed from a set of discrete data points representing the cell averages of some function containing a kink, a local extremum, and a jump discontinuity. Cell average values are represented by horizontal lines covering each computational cell. The dotted line represents PHM, the dash-dotted line the PowerPHM, and the solid line the PowerENO in each computational cell, respectively.

We observe that the local power hyperbolic reconstruction assigns hyperbolas with larger gradient than the ones for the PHM reconstruction.

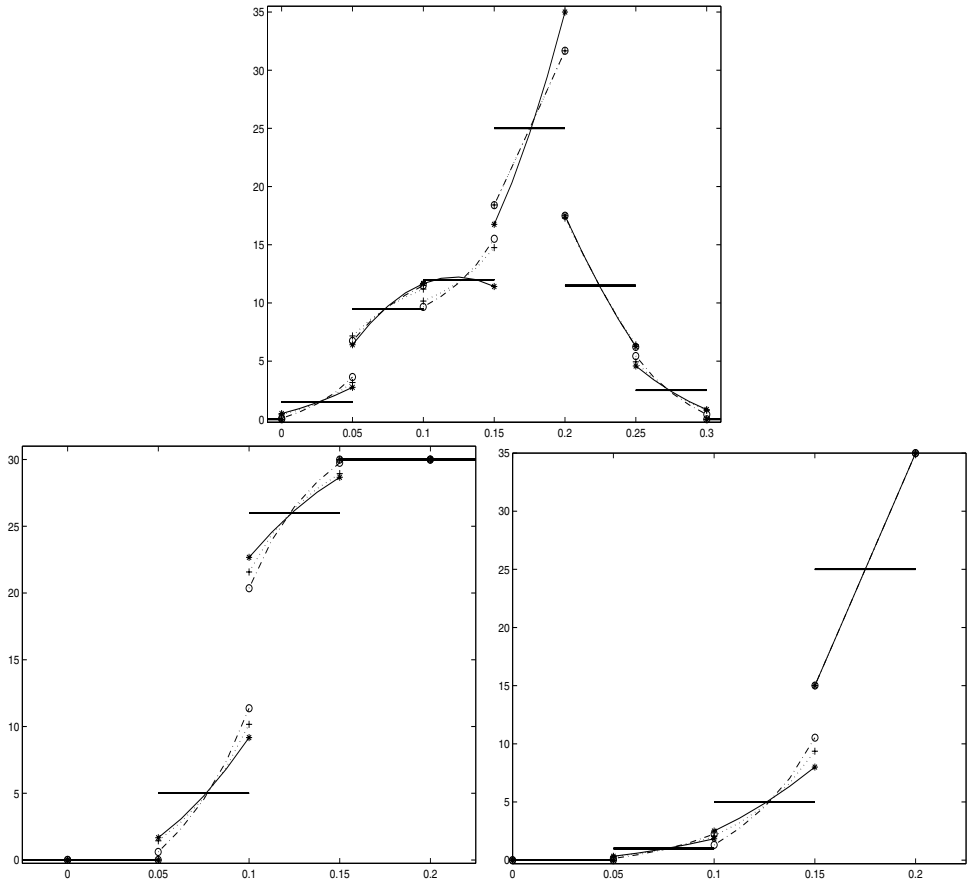


FIG. 3.1. *Piecewise smooth reconstructions: Local extrema (top), jump discontinuity (bottom left), discontinuity in derivative (bottom right). PHM, dotted line; PowerPHM, dash-dotted line; and PowerENO, solid line.*

4. Numerical experiments. The procedure to design a PowerPHM scheme for scalar hyperbolic conservation laws reads as follows. We start with an approximate Riemann solver that computes the numerical flux (in an explicit way) at every interface from neighboring cell averages, following Godunov's idea to get a spatial discretization in conservation form. In order to get high order accuracy in space we use the so-called *flux formulation* by Shu and Osher [20], based on the following simple lemma.

LEMMA 4.1. [illegible] $g(x)$ [illegible]

$$f(u(x)) = \frac{1}{h}\int_{x-\frac{h}{2}}^{x+\frac{h}{2}} g(\psi)d\psi, \tag{4.1}$$

[illegible]

$$f(u(x))_x = \frac{g(x+\frac{h}{2}) - g(x-\frac{h}{2})}{h}. \tag{4.2}$$

This lemma implies that for approximating the numerical flux $f_{j+\frac{1}{2}}$ with a high order of accuracy, it is enough to reconstruct $g_{j+\frac{1}{2}}$ up to the same order of accuracy.

TABLE 4.1
Errors and numerical orders of $u_t + u_x = 0$, $u_0(x) = \frac{1}{2}\left(\frac{1}{2} + \sin(2\pi x)\right)$, $0 \le x \le 1$ *at* $t = 1$.

	N	L_∞-error	L_∞ order	L_1-error	L_1 order
PHM	20	7.11E-02	-	2.98E-02	-
	40	2.56E-02	1.47	7.97E-03	1.90
	80	9.01E-03	1.51	1.80E-03	2.15
	160	2.96E-03	1.61	5.45E-04	1.72
PowerPHM	20	5.35E-02	-	1.92E-02	-
	40	1.63E-02	1.72	3.65E-03	2.40
	80	4.73E-03	1.78	9.56E-04	1.93
	160	1.25E-03	1.93	1.82E-04	2.39

In our case, we use our piecewise hyperbolic power reconstruction procedure to approximate $g_{j+\frac{1}{2}}$ up to third order of accuracy. We also use the third order accurate TVD Runge–Kutta method to evolve in time, introduced in [20], in order to be consistent with the spatial accuracy. The resulting scheme is a convex combination of three Euler explicit time-stepping schemes in conservation form. We call this scheme PowerPHM. This Euler explicit time-stepping procedure is total variation stable under a suitable CFL restriction $\lambda = \frac{\Delta t}{h} \le \lambda_0$, where λ_0 is inversely proportional to $\max|f'(u)|$ as usual.

4.1. Accuracy test. To study the accuracy of our method, we consider the scalar linear equation $u_t + u_x = 0$ with $u(x,0) = u_0(x)$ periodic in $0 \le x \le 1$,

$$u_0(x) = \frac{1}{2}\left(\frac{1}{2} + \sin(2\pi x)\right).$$

We have computed the solution at $t = 1$ with CFL $= 0.8$ and the mesh refinement sequence $N = 20, 40, 80, 160$ grid points for the PHM and the PowerPHM, respectively. We show in Table 4.1 the corresponding L_∞- and L_1-errors that compare with the exact solution. We observe smaller L_∞- and L_1-errors for the PowerPHM due to its better resolution of local extrema. The new method shows improved numerical orders of accuracy compared with the $O(h^{\frac{3}{2}})$ global accuracy of the PHM procedure.

4.2. Linear advection. For the linear advection test, PowerPHM has been based on Godunov's first order sheme for scalar conservation laws. We use the label PowerPHM-GOD to refer to this scheme.

We solve the linear equation $u_t + u_x = 0$ for two periodic initial data defined in $0 \le x \le 1$:

$$u_0(x) = \begin{cases} 1, & 0.25 \le x \le 0.75, \\ 0 & \text{otherwise} \end{cases} \quad \text{and} \quad u_1(x) = \begin{cases} \sin\left(\pi \frac{x-0.2}{0.6}\right), & 0.2 \le x \le 0.8, \\ 0 & \text{otherwise.} \end{cases}$$

We compute the approximate solution for PHM and PowerPHM using a grid of 100 points with a CFL factor of 0.8. We display in Figure 4.1 two output times that correspond to 10 and 200 periods of both signals. We observe a very dissipative behavior of both methods at large times.

We observe that PowerPHM smears jump discontinuities less and better resolves local extrema and discontinuities in the first derivative than PHM.

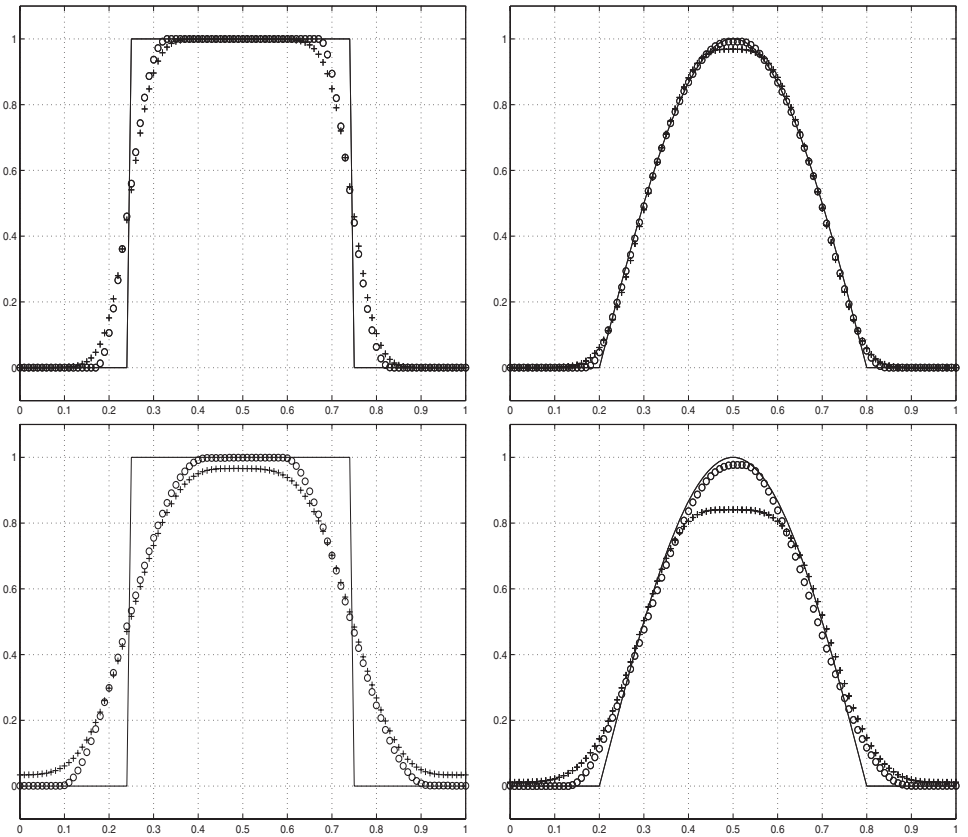


FIG. 4.1. *Numerical solution of linear advection with* 100 *grid points, CFL*=0.8 *until time* 10 *(top) and time* 200 *(bottom), for PHM-GOD scheme "+" and PowerPHM-GOD schemes "o."*

4.3. Euler equations. We consider the one-dimensional Euler equations of gas dynamics

$$\begin{aligned} \rho_t + (\rho u)_x &= 0, \\ (\rho u)_t + \left(P + \frac{(\rho u)^2}{\rho} \right)_x &= 0, \\ E_t + (u(E+P))_x &= 0, \end{aligned}$$

where ρ is the density, u is the velocity, P is the pressure, and $E = \frac{1}{2}\rho u^2 + \rho\epsilon$ is the total energy, ϵ being the specific internal energy per unit volume. We compute the pressure by means of the ideal gas equation of state $P = (\gamma - 1)\rho\epsilon$, where γ is the adiabatic exponent. We use $\gamma = 1.4$ in our experiments. We consider initial value problems by means of an initial profile of the density, velocity, and pressure defined through the domain.

We have chosen three significant test problems of compressible gas dynamics where strong dynamics are present. In order to get a more complete picture of the behavior of the new scheme we compare the results with the PHM scheme and the fifth order parabolic Weighted PowerENO method.

The PHM, PowerPHM, and Weighted PowerENO schemes we have implemented for systems of conservation laws are based on the first order Marquina's flux formula

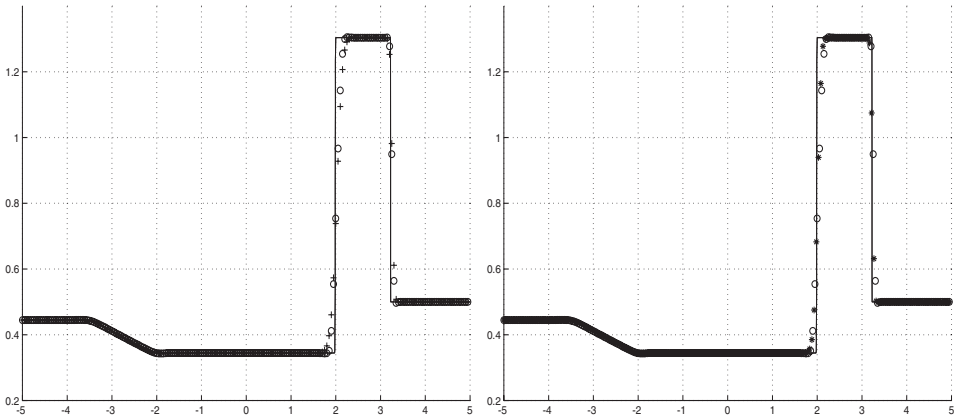


FIG. 4.2. *Numerical solution of the density profile for the Lax shock tube problem at* $t = 1.3$*,* $CFL = 0.8$*, 200 points. Left: The PHM-MFF scheme "+" and PowerPHM-MFF scheme "o." Right: PowerPHM-MFF scheme "o" and Weighted PowerENO "*."*

(MFF) [2], applying the corresponding reconstruction procedure to the local characteristic fluxes and variables, and using the third order accurate TVD Runge–Kutta method for the evolution in time.

We use the labels PHM-MFF, PowerPHM-MFF, and Weighted PowerENO-MFF to refer to these schemes.

4.3.1. Lax shock tube problem. We consider the following Riemann problem due to Lax [8]:

$$u_0(x) = \begin{cases} (0.445, 0.698, 3.528) & -5 \le x < 0, \\ (0.5, 0, 0.571) & 0 \le x \le 5. \end{cases}$$

We have displayed in Figures 4.2 and 4.3 the density profile of the Lax shock tube problem at time $t = 1.3$ and its zoomed regions, respectively. We obtain correct position and speed of discontinuities. We can observe a slight reduction in the smearing of the contact wave for our PowerPHM in comparison with PHM. Comparable results of PowerPHM with the Weighted PowerENO method are displayed. The interesting feature of our PowerPHM is the "sharpening" of jump discontinuities with the same order of spatial accuracy as the classical PHM.

4.3.2. Shock entropy wave interaction. We consider the one-dimensional wave interaction introduced in [20] for testing stability and accuracy of high order methods. The initial data are

$$u_0(x) = \begin{cases} (3.85714, 2.62936, 10.33333) & -5 \le x < -4, \\ (1 + 0.2\sin(5x), 0, 1) & -4 \le x \le 5. \end{cases}$$

We use a uniform grid with 400 points and we evolve until time 1.8, using PHM-MFF, PowerPHM-MFF, and Weighted PowerENO-MFF. We display in Figure 4.4 the zoomed area of the perturbed region behind the shock wave where we observe an improved resolution of our PowerPHM scheme compared with PHM.

The fifth order Weighted PowerENO method better resolves the smooth fine structure developed in this test.

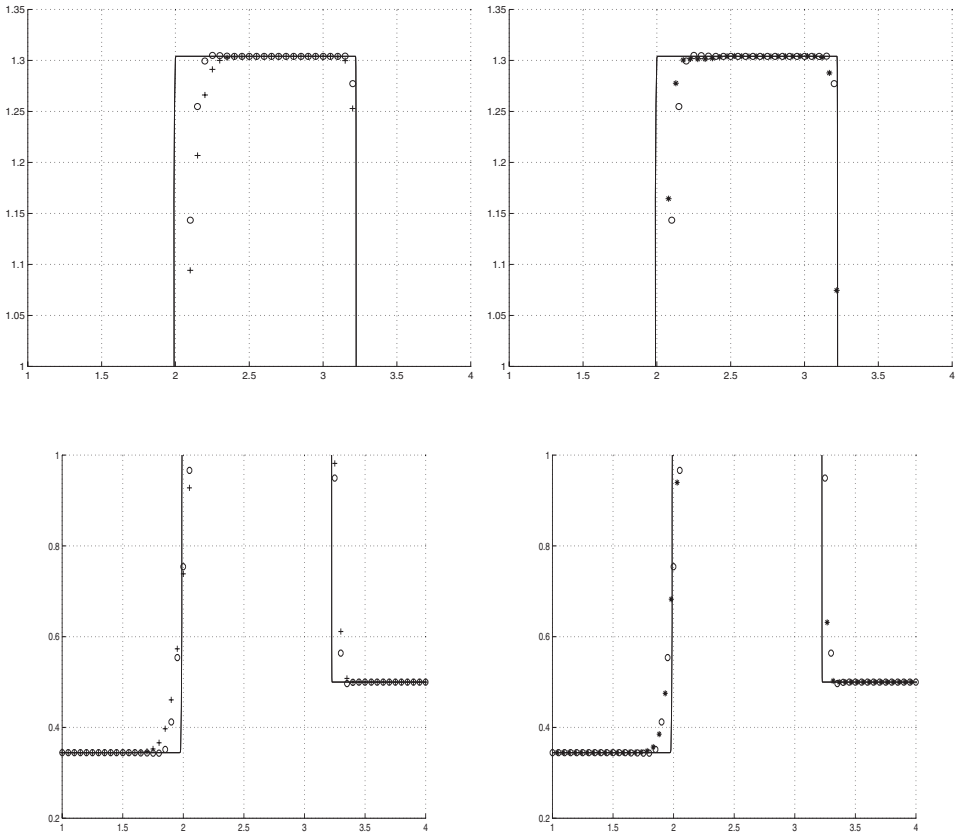


FIG. 4.3. *Zoomed regions of the density profile of Lax shock tube problem.*

4.3.3. Interaction of blast waves. This one-dimensional interacting shock wave problem was suggested in [23] as a very challenging numerical test.

The initial data to simulate the interaction of two blast waves consists in three constant states:

$$u_0(x) = \begin{cases} (\rho_L, v_L, P_L) = (1, 0, 10^3), & 0 \le x < 0.1, \\ (\rho_M, v_M, P_M) = (1, 0, 10^{-2}), & 0.1 \le x < 0.9, \\ (\rho_R, v_R, P_R) = (1, 0, 10^2), & 0.9 \le x \le 1. \end{cases}$$

Reflecting boundary conditions are applied on both ends of the interval. The density field of the solution to this problem after the collision contains interaction of rarefaction waves, contact discontinuities, and shocks.

Figures 4.5 and 4.6 show the density component computed with 400 grid points evolved until times 0.028 and 0.038, respectively, with CFL factor 0.5.

The three schemes behave robustly along the evolution of this strong test. The local extrema are better resolved for the fifth order accurate Weighted PowerENO scheme. We observe an improved result for the PowerPHM scheme compared with the PHM scheme.

4.3.4. Computational cost and efficiency. In this section we analyze the efficiency and computational cost of the reconstruction procedures under study for the one-dimensional shock test problems.

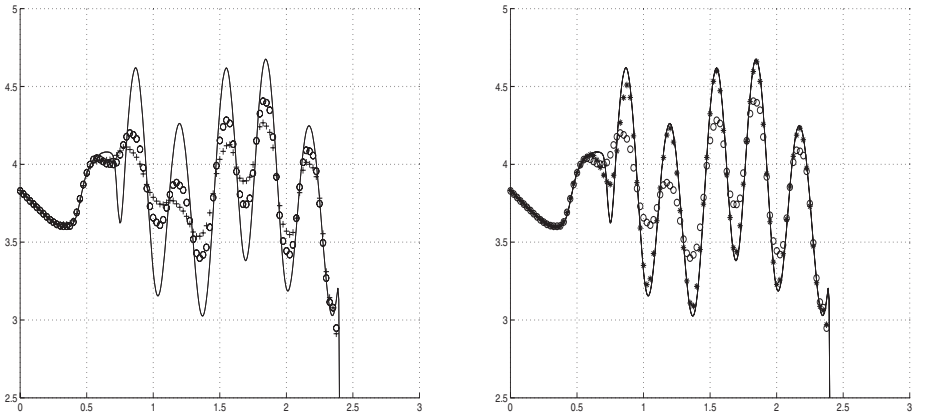


FIG. 4.4. *Zoomed region of the numerical solution of the density profile for the shock-entropy wave interaction with PHM-MFF "+" and PowerPHM-MFF "o" (left), PowerPHM-MFF and Weighted PowerENO-MFF (right) using a grid of* 400 *points with* $CFL = 0.8$ *until time* 1.8.

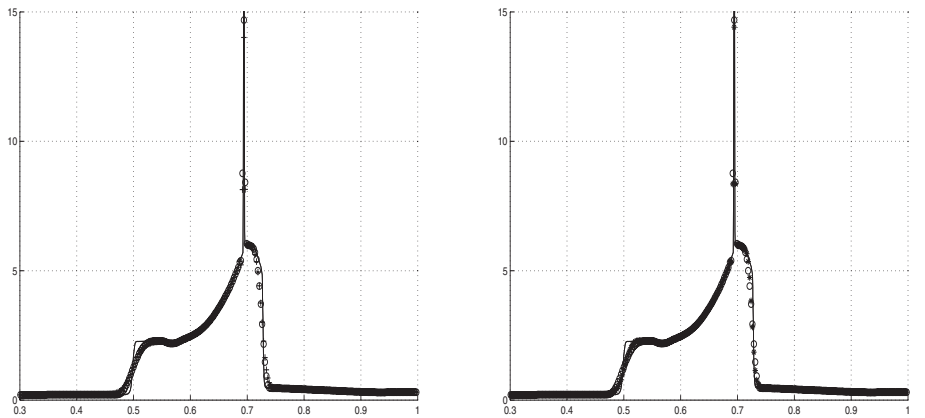


FIG. 4.5. *Centered region of the density profile for the interaction of blast waves test at* $t = 0.028$, $CFL = 0.5$, 400 *points. Left: The PHM-MFF scheme "+" and PowerPHM-MFF scheme "o." Right: PowerPHM-MFF scheme "o" and Weighted PowerENO "*."*

In Tables 4.2, 4.3, and 4.4 we display the L_1-error, as an overall error when discontinuities are present, of the solution computed with respect to a reference solution for a mesh refinement sequence together with the CPU time for each of the computations.

The reference solution for the Lax shock tube problem is the exact solution of the Riemann problem [18]. The reference solution for the shock entropy test and the two interacting blast waves problem were computed with the Weighted PowerENO scheme with 12800 grid points. The computing times are expressed in seconds and are obtained for a fixed number of iterations for the three schemes. The ratio $\frac{\Delta t}{\Delta x}$ is 0.1 for the Lax shock tube problem and the shock entropy interaction test and 0.01 for the interaction of blast waves.

For each of the test problems and schemes, the errors reduce under mesh refinement. The errors give a rough measure of the accuracy of the methods since the L_1-error gives the same weight to the points belonging to the smooth regions and the

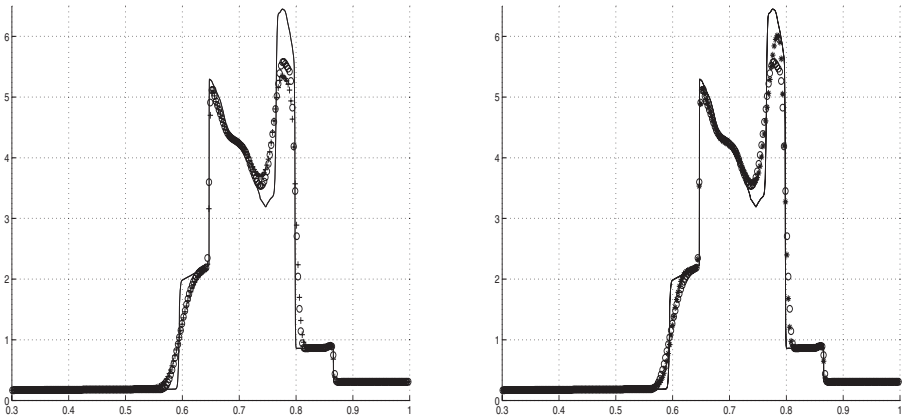


FIG. 4.6. *Centered region of the density profile for the interaction of blast waves test at* $t = 0.038$, $CFL = 0.5$, *400 points. Left: The PHM-MFF scheme "+" and PowerPHM-MFF scheme "o." Right: PowerPHM-MFF scheme "o" and Weighted PowerENO "*."*

TABLE 4.2
L_1*-errors and timings for the Lax shock tube problem.* $CFL = 0.1$, $t = 1.3$.

	PHM		PowerPHM		WPowerENO	
N	L_1-error	Time	L_1-error	Time	L_1-error	Time
100	2.12 E-02	0.51	1.81 E-02	0.52	1.78 E-02	0.89
200	1.20 E-02	2.02	1.01 E-02	2.01	9.61 E-03	3.36
400	6.68 E-03	7.95	5.75 E-03	8.05	5.32 E-03	13.51
800	3.19 E-03	31.56	2.63 E-03	32.08	2.31 E-03	54.31

TABLE 4.3
L_1*-errors and timings for the shock entropy wave interaction.* $CFL = 0.1$, $t = 1.8$.

	PHM		PowerPHM		WPowerENO	
N	L_1-error	Time	L_1-error	Time	L_1-error	Time
100	1.45 E-01	0.68	1.40 E-01	0.70	1.38 E-01	1.20
200	1.02 E-01	2.86	9.88 E-02	2.89	8.68 E-02	4.86
400	6.82 E-02	11.44	5.55 E-02	11.46	4.43 e-02	19.49
800	3.03 E-02	45.93	2.31 E-02	45.68	2.13 E-02	77.37

TABLE 4.4
L_1*-errors and timings for the interaction of blast waves problem.* $CFL = 0.01$, $t = 0.038$.

	PHM		PowerPHM		WPowerENO	
N	L_1-error	Time	L_1-error	Time	L_1-error	Time
100	3.16 E-01	1.44	2.90 E-01	1.48	2.86 E-01	2.56
200	1.95 E-01	5.85	1.72 E-01	5.88	1.53 E-01	10.15
400	1.12 E-01	23.34	9.56 E-02	23.77	8.28 E-02	40.55
800	5.05 E-02	92.41	4.71 E-02	94.84	4.17 E-02	163.19

points of the discontinuity regions. A more complete understanding of the accuracy of the methods can be obtained from direct observation of Figures 4.2, 4.3, 4.4, 4.5, and 4.6. The computational times are similar for both hyperbolic schemes and smaller than the corresponding times for the fifth order parabolic scheme.

4.4. Two-dimensional Riemann problem. We consider the Riemann problem corresponding to configuration 3 defined in [16] in the domain $[0,1] \times [0,1]$

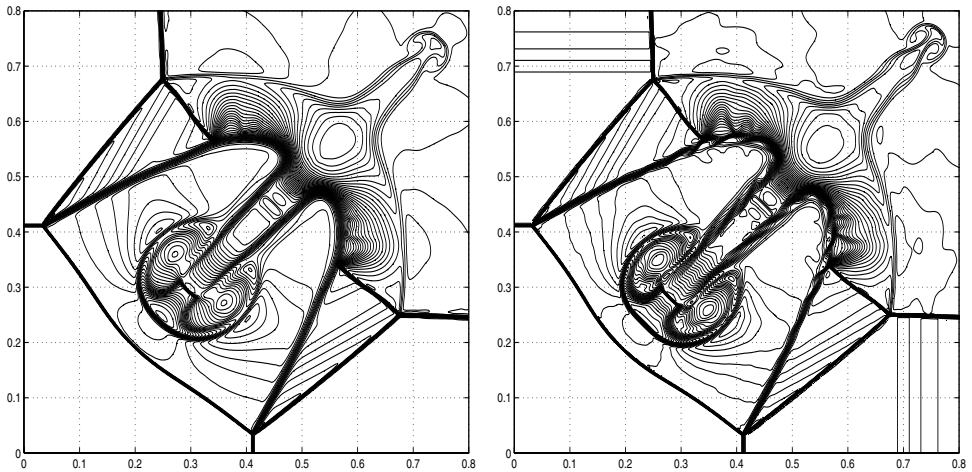


FIG. 4.7. *Zoomed region of the density contour lines of the numerical solution of the two-dimensional Riemann problem using* 400×400 *points,* $CFL = 0.8$*, at time* 0.8 *for the PHM-MFF scheme (left) and PowerPHM-MFF (right).*

localizing the contact point at $(0.75, 0.75)$ instead of the center of the square. We evolve in time until 0.8 with a CFL factor of 0.8 and a grid of 400×400 points. In Figure 4.7 we display the approximations of the density contour profile obtained with the PHM-MFF and PowerPHM-MFF schemes. We can observe a substantial improvement at the contact where the instability is manifest for the PowerPHM-MFF scheme. We also notice a better resolution of the structure appearing at the top right region for the PowerPHM-MFF scheme.

5. Conclusions. In this paper we have analyzed the power limiters to know their suitability to be used as slope-limiters for piecewise hyperbolic reconstructions. We have found that the powereno limiter has the desired qualities for the design of a piecewise hyperbolic reconstruction. We formulate a new PHM based on the powereno limiter (we call it PowerPHM) that behaves stably with reduced numerical viscosity compared to PHM. We show that this new method is LTVB and improves the behavior of PHM at local extrema, jump discontinuities, and discontinuities in derivative. We have shown numerical evidence of the above mentioned features by means of one- and two-dimensional numerical tests with the Euler equations of gas dynamics.

Acknowledgment. The author thanks Professor Antonio Marquina for stimulating discussions and comments on this paper.

REFERENCES

[1] R. ARTEBRANT AND H. J. SCHROLL, *High-resolution Riemann-solver-free methods for conservation laws*, in Hyperbolic Problems: Theory, Numerics, Applications, Springer-Verlag, Berlin, 2003, pp. 305–314.

[2] R. DONAT AND A. MARQUINA, *Capturing shock reflections: An improved flux formula*, J. Comput. Phys., 125 (1996), pp. 42–58.

[3] R. P. FEDKIW, A. MARQUINA, AND B. MERRIMAN, *An isobaric fix for the overheating problem in multimaterial compressible flows*, J. Comput. Phys., 148 (1999), pp. 545–578.

[4] S. K. GODUNOV, *A finite difference method for the computation of discontinuous solutions of the equations of fluid dynamics*, Mat. Sb., 47 (1959), pp. 357–393 (in Russian).

[5] G. H. Hardy, J. E. Littlewood, and G. Polya, *Inequalities*, Cambridge University Press, Cambridge, UK, 1973.
[6] A. Harten, *High resolution schemes for hyperbolic conservation laws*, J. Comput. Phys., 49 (1983), pp. 357–393.
[7] A. Harten, B. Engquist, S. Osher, and S. Chakravarthy, *Uniformly high order accurate essentially nonoscillatory schemes* III, J. Comput. Phys., 71 (1987), pp. 231–303.
[8] P. D. Lax, *Weak solutions of nonlinear hyperbolic equations and their numerical computation*, Comm. Pure Appl. Math., 7 (1954), pp. 159–193.
[9] S. Li and L. Petzold, *Moving mesh methods with upwinding schemes for time-dependent PDEs*, J. Comput. Phys., 131 (1997), pp. 368–377.
[10] R. J. LeVeque *Numerical Methods for Conservation Laws*, Birkhäuser Verlag, Basel, 1990.
[11] A. Marquina, *Local piecewise hyperbolic reconstruction of numerical fluxes for nonlinear scalar conservation laws*, SIAM J. Sci. Comput., 15 (1994), pp. 892–915.
[12] A. Marquina and P. Mulet, *A flux-split algorithm applied to conservative models for multicomponent compressible flows*, J. Comput. Phys., 185 (2003), pp. 120–138.
[13] S. Osher and S. Chakravarthy, *High resolution schemes and the entropy condition*, SIAM J. Numer. Anal., 21 (1984), pp. 955–984.
[14] P. L. Roe, *Some contributions to the modelling of discontinuous flows*, in Large-Scale Computations in Fluid Mechanics, Part 2, Lectures in Appl. Math. 22, B. Enquist, S. Osher, and R. J. Somerville, eds., AMS, Providence, RI, 1985, pp. 163–193.
[15] H. J. Schroll, *High resolution relaxed upwind schemes in gas dynamics*, J. Sci. Comput., 17 (2002), pp. 599–607.
[16] C. W. Schulz-Rinne, J. P. Collins, and H. M. Glaz, *Numerical solution of the Riemann problem for two-dimensional gas dynamics*, SIAM J. Sci. Comput., 14 (1993), pp. 1394–1414.
[17] S. Serna and A. Marquina, *Power ENO Methods: A fifth order accurate weighted power ENO method,* J. Comput. Phys., 194 (2004), pp. 632–658.
[18] J. Smoller, *Shock Waves and Reaction-Diffusion Equations,* Springer-Verlag, New York, 1983.
[19] Y. Stiriba, *A nonlinear flux split method for hyperbolic conservation laws,* J. Comput. Phys., 176 (2002), pp. 20–39.
[20] C. W. Shu and S. J. Osher, *Efficient implementation of essentially nonoscillatory shock capturing schemes* II, J. Comput. Phys., 83 (1989), pp. 32–78.
[21] B. van Leer, *Towards the ultimate conservative difference scheme,* II: *Monotonicity and conservation combined in a second order scheme*, J. Comput. Phys., 14 (1974), pp. 361–370.
[22] B. van Leer, *Towards the ultimate conservative difference scheme,* V: *A second order sequel to Godunov's method*, J. Comput. Phys., 32 (1979), pp. 101–136.
[23] P. Woodward and P. Colella, *The numerical simulation of two-dimensional fluid flow with strong shocks*, J. Comput. Phys., 54 (1984), pp. 115–173.

SIAM J. SCI. COMPUT.
Vol. 28, No. 1, pp. 141–171
© 2006 Society for Industrial and Applied Mathematics

A MULTIGRID METHOD FOR INCOMPRESSIBLE FLOW PROBLEMS USING QUASI DIVERGENCE FREE FUNCTIONS*

GUNAR MATTHIES† AND FRIEDHELM SCHIEWECK‡

Abstract. We consider a finite element method of higher order on a quadrilateral or triangular, hexahedral or tetrahedral mesh for solving the linearized Navier–Stokes equations (Oseen equations) by means of discontinuous elements for the pressure and suitable conforming elements for the velocity such that the global and the local inf-sup condition are satisfied. Our goal is the construction of a coupled multigrid solver for the corresponding algebraic system of equations. The idea of this solver is to switch inside of the multigrid method to a new velocity basis which leads to a reduced system with a much lower number of unknowns as well as a very small number of couplings between them. We call this new basis quasi divergence free since most of the basis functions are discretely divergence free which implies that they do not have any coupling to the pressure. The quasi divergence free basis functions can be constructed locally during the assembling process of the stiffness matrix. We create a multigrid method for solving the reduced problem efficiently. Since most of the velocity basis functions are completely decoupled from the pressure we can construct a smoother with low computational costs. The efficiency of the new multigrid method compared with other known coupled multigrid solvers is demonstrated by numerical experiments for a Stokes problem, the driven cavity problem, and a benchmark problem.

Key words. higher order finite elements, quasi divergence free basis, incompressible flow, Oseen equations, Navier–Stokes equations, multigrid

AMS subject classifications. 65N22, 65N30, 65N55, 76D07, 76D05

DOI. 10.1137/04061814X

1. Introduction.

Many processes in science and engineering include fluid flow coupled with other physical phenomena. For the numerical simulation of such complex applications, it is very important to have a fast solver for flow problems. In the last few years, higher order finite element methods turned out to be very efficient numerical methods concerning the number of unknowns in relation to the achieved accuracy. In this paper, we consider the numerical solution of an incompressible flow problem by means of a higher order finite element method and the construction of a multigrid solver for the corresponding algebraic system of equations.

For simplicity, we take as model problem the stationary d-dimensional , ($d = 2$ or 3) with Dirichlet boundary conditions, i.e., we want to find a velocity $u : \Omega \to \mathbb{R}^d$ and a pressure $p : \Omega \to \mathbb{R}$ such that

$$\begin{aligned} -\operatorname{div} \mathbb{T}(u,p) + (\beta \cdot \nabla)u &= f \quad \text{in } \Omega, \\ \operatorname{div} u &= 0 \quad \text{in } \Omega, \\ u &= g \quad \text{on } \Gamma, \end{aligned} \tag{1.1}$$

where $\mathbb{T}(u,p) \in \mathbb{R}^{d\times d}$ denotes the stress tensor defined via the deformation tensor

*Received by the editors November 3, 2004; accepted for publication (in revised form) September 12, 2005; published electronically March 3, 2006.
http://www.siam.org/journals/sisc/28-1/61814.html

†Fakultät für Mathematik, Ruhr-Universität Bochum, Universitätsstraße 150, D-44780 Bochum, Germany (gunar.matthies@ruhr-uni-bochum.de, http://www.ruhr-uni-bochum.de/jpnum).

‡Institut für Analysis und Numerik, Otto-von-Guericke-Universität Magdeburg, Postfach 4120, D-39016 Magdeburg, Germany (friedhelm.schieweck@mathematik.uni-magdeburge.de, http://www-ian.math.uni-magdeburg.de/∼schiewec).

$\mathbb{D}(u) \in \mathbb{R}^{d\times d}$ for the velocity $u = (u_1, \dots, u_d)$ in the following way:

$$\mathbb{T}(u,p) := -p\mathbb{I} + 2\nu\mathbb{D}(u), \quad \mathbb{I} := (\delta_{i,j}), \quad \mathbb{D}(u) := \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right).$$

The vector function $\beta : \Omega \to \mathbb{R}^d$ is a given convection field with $\operatorname{div}\beta = 0$. Furthermore, $\Omega \subset \mathbb{R}^d$ is a connected, bounded domain with a polygonal ($d = 2$) or polyhedral ($d = 3$) boundary $\Gamma := \partial\Omega$, ν the viscosity, $f : \Omega \to \mathbb{R}^d$ a given body force and $g : \Gamma \to \mathbb{R}^d$ the prescribed boundary data satisfying the solvability condition $\int_\Gamma g \cdot n\, ds = 0$, where n denotes the outer normal unit vector on the boundary Γ. Problem (1.1) arises if a Picard iteration is applied to the stationary Navier–Stokes equations [25, 5] or it corresponds to the Stokes equations if $\beta \equiv 0$.

We use a higher order conforming finite element method on triangles and/or quadrilaterals ($d = 2$) and tetrahedra or hexahedra ($d = 3$), respectively. It is essential for our method to have a discontinuous pressure approximation and a finite element pair for velocity and pressure that satisfies the local (elementwise) and the global inf-sup condition. Let us emphasize that a discontinuous pressure is not a disadvantage. By means of a simple postprocessing procedure one can compute a continuous piecewise polynomial pressure function which maintains the optimal approximation order of the discontinuous pressure solution [21]. In order to get a method of order $r \geq 2$ in the energy norm, we approximate the pressure by discontinuous piecewise P_{r-1}-elements. On quadrilaterals and hexahedra a continuous piecewise Q_r-approximation for the velocity is used. For triangles and tetrahedra we approximate the velocity by continuous functions where the restriction to each element is contained in a suitable enrichment of the polynomial space P_r. Details are given in [4, 7]. For the sake of simplicity, we restrict ourselves to a fixed polynomial degree r for all elements. However, our method can be generalized easily to the case where the degree can vary from element to element.

In the literature, a lot of work has been devoted to the solution of the large algebraic system in saddle point form that corresponds to the discrete problem associated with (1.1). We refer to the overview paper [2] and to [5, 18, 25] for some recent work in the context of computational fluid dynamics. In this paper, we consider the category of so-called [illegible] where the unknown vectors for velocity and pressure are computed simultaneously. Here, our goal is to achieve an improvement for the coupled multigrid solvers in the case of higher order finite element discretizations of (1.1). The problem of the existing multigrid solvers in this category is that they use a block Gauss–Seidel smoother (Vanka smoother) with a relatively large block size which leads to high computational costs. Typically this block size is the number of all degrees of freedom for velocity and pressure that belong to one element. That means the block size becomes large for higher order elements, in particular, in the 3D case [14, 11, 12].

The new idea of this solver is to switch inside of the multigrid method to a new basis of a suitable velocity subspace which leads at the end to a highly reduced number of velocity and pressure unknowns as well as to a very small number of couplings between them. We call this new basis a [illegible] or, for convenience, a [illegible] since most of the basis functions are discretely divergence free. This implies that they do not have any coupling to the pressure. The QDF basis functions can be constructed locally during the assembling process of the stiffness matrix. The discrete Oseen problem is equivalent to a subspace formulation called [illegible] in the

following. This problem is further decomposed by an orthogonal splitting into two subproblems. The first one is a problem for the element bubble part u_b of the velocity solution $u_h = u_b + \widetilde{u}_s$. It can be solved independently from the remaining part $\widetilde{u}_s$ in an elementwise way during the assembling process of the stiffness matrix. The second problem of the decomposition is a [illegible] for the velocity part $\widetilde{u}_s$ and the piecewise constant part $\overline{p}_h$ of the pressure solution $p_h = \overline{p}_h + \widetilde{p}_h$. It has a much smaller dimension than the original discrete Oseen problem. The remaining pressure part $\widetilde{p}_h$ can be computed at the end by an elementwise postprocessing procedure.

In this paper, we create a multigrid method for solving the QDF formulation of the Oseen problem efficiently. The most important ingredient of this method is the construction of a suitable QDF velocity basis where almost all basis functions are discretely divergence free. Only those basis functions which are associated with so-called [illegible] are not discretely divergence free. The number of flux nodes is equal to the number of the $(d-1)$-dimensional faces of the grid, i.e., it is relatively small compared to the total number of velocity unknowns in the reduced Oseen problem. Since each discretely divergence free basis function is completely decoupled from the pressure, this, in principle, allows one to apply a point Gauss–Seidel smoothing to all nonflux nodes. Thus, switching inside of our multigrid method to the QDF velocity basis allows us to construct a smoothing procedure with very low computational costs compared to existing multigrid methods from the literature. For a bounded polynomial degree r, the computational overhead to work with the QDF basis is only proportional to the number of unknowns. The required memory for an efficient implementation of the QDF basis is less than the memory that will become free from the fact that a lot of nonzero matrix entries of the original finite element method are no longer used for our new multigrid method.

The idea to use a discretely divergence free basis for the velocity space is not new. For the 2D situation, we refer in case of continuous pressure elements to [27, 28] and in case of a discontinuous pressure approximation to [8, 9, 23, 4, 3, 24, 26] for elements up to second order and to [22] for higher order finite element approximations. Efficient multigrid solvers based on a discretely divergence free basis of the velocity space are constructed in [3, 24]. However, the main drawback of this approach is that the construction of such a basis in the 3D case is very complicated [10]. Therefore, it seems to be better to renounce a complete discretely divergence free basis and to work instead with a QDF basis which might be the right compromise.

The paper is organized as follows. We introduce some notation and define the finite element spaces in section 2. The construction of the QDF basis of the velocity space is presented in section 3. In section 4 we show how the saddle point problem created from the original basis is decomposed into local subproblems on each element and a global system with a highly reduced dimension. The aspects of an efficient implementation of our new method are discussed in section 5. In section 6 we describe the main components of our multigrid method and present a solution technique for the Navier–Stokes problem. The performance of the new multigrid method compared with other coupled multigrid solvers known from the literature is tested in section 7 by means of numerical experiments for a Stokes problem, the driven cavity problem, and a benchmark problem [20].

2. Preliminaries and notation. For a domain $G \subset \mathbb{R}^d$, the $L^2(G)$ inner product and norm are denoted by $(\cdot,\cdot)_G$ and $\|\cdot\|_{0,G}$, respectively, for scalar functions as well as for vector- or tensor-valued functions. For example, for tensor-valued

functions $A, B : G \to \mathbb{R}^{d\times d}$, we define the inner product as

$$(A,B)_G := \int_G \sum_{i,j=1}^{d} A_{i,j}(x)B_{i,j}(x)dx\,.$$

For a sufficiently smooth $(d-1)$-dimensional manifold $\gamma \subset \overline{\Omega}$, we denote by $\langle\cdot,\cdot\rangle_\gamma$ the inner product in $L^2(\gamma)$ or in $(L^2(\gamma))^d$, respectively. For a domain $G \subset \mathbb{R}^d$, the Sobolev spaces $H^k(G) = W^{k,2}(G)$ are defined in the usual way with the standard norm and semi-norm $\|\cdot\|_{k,G}$ and $|\cdot|_{k,G}$, respectively.

We define the spaces X, X_0 and M, M_0 for velocity and pressure, respectively, as

$$\begin{aligned} X &:= (H^1(\Omega))^d\,, & X_0 &:= (H_0^1(\Omega))^d\,,\\ M &:= L^2(\Omega), & M_0 &:= L_0^2(\Omega) := \{q \in L^2(\Omega) : (q,1)_\Omega = 0\}. \end{aligned}$$

Let $\widetilde{g} \in X$ be an extension of the boundary function $g : \Gamma \to \mathbb{R}^d$ of (1.1) into the domain Ω. Then, the weak formulation of problem (1.1) reads as follows:

$$(u,p) \in (\widetilde{g} + X_0) \times M_0\,,$$

$$\begin{aligned} a(u,v) + b(v,p) &= (f,v)_\Omega \qquad &&\forall\, v \in X_0,\\ b(u,q) &= 0 &&\forall\, q \in M_0, \end{aligned} \tag{2.1}$$

where

$$a(u,v) := 2\nu\,(\mathbb{D}(u),\mathbb{D}(v))_\Omega + ((\beta\cdot\nabla)u,v)_\Omega\,, \qquad b(v,p) := -(\operatorname{div} v,p)_\Omega\,. \tag{2.2}$$

For the discretization of this problem, we decompose the domain $\Omega \subset \mathbb{R}^d$ into elements $K \in \mathcal{T}_h$ which are allowed to be triangles and/or quadrilaterals in the case $d = 2$ and either tetrahedra or hexahedra in the case $d = 3$. We assume that the grid $\mathcal{T}_h$ is regular and shape-regular in the usual sense. The diameter of an element K is denoted by h_K and the global mesh-size of the grid $\mathcal{T}_h$ is defined as $h := \max_{K\in\mathcal{T}_h} h_K$. For an element $K \in \mathcal{T}_h$, we denote by $\mathcal{E}(K)$ the set of all $(d-1)$-dimensional faces of K, by $\mathcal{E}_h := \bigcup_{K\in\mathcal{T}_h} \mathcal{E}(K)$ the set of all faces of the grid $\mathcal{T}_h$, by $\mathcal{E}_h(\Gamma)$ the set of all boundary faces $E \subset \Gamma$ and by $\mathcal{E}_h^0 := \mathcal{E}_h \setminus \mathcal{E}_h(\Gamma)$ the set of all inner faces. Note that a $(d-1)$-dimensional face $E \in \mathcal{E}_h$ can belong to at most two different elements. For a given inner face $E \in \mathcal{E}_h^0$, let $K(E)$ and $K'(E)$ be those two elements of the grid $\mathcal{T}_h$ which have E as their common face. For a boundary face $E \in \mathcal{E}_h(\Gamma)$, let $K(E)$ denote that element such that $E \in \mathcal{E}(K(E))$. We will denote by n_E^K the unit normal vector of a face $E \in \mathcal{E}(K)$ which is directed outward with respect to the element K. To each face $E \in \mathcal{E}_h$ we assign the unit normal vector $n_E := n_E^{K(E)}$.

We denote by $\mathbb{P}_m(G)$ the space of all polynomials on the domain $G \subset \mathbb{R}^d$ with total degree less than or equal to m and by $\mathbb{Q}_m(G)$ the space of those polynomials where the maximum power in each coordinate is less than or equal to m. For an element K, let $F_K : \widehat{K} \to K$ be the one-to-one mapping between the reference element $\widehat{K}$ and K. For sufficient conditions that guarantee the bijectivity of nonaffine reference mappings F_K, we refer to [17].

It is essential for our approach that there are no continuity requirements for the pressure approximation.

The finite element spaces M_h and $M_{h,0}$ for the pressure with $M_h \subset M$ and $M_{h,0} \subset M_0$ are defined as

$$M_h := \{q \in L^2(\Omega) :\ q\big|_K \circ F_K \in \mathbb{P}_{r-1}(\widehat{K}) \quad \forall K \in \mathcal{T}_h\}\,, \qquad M_{h,0} := M_h \cap M_0\,. \tag{2.3}$$

The velocity spaces X and X_0 will be approximated by the finite element spaces $X_h \subset X$ and $X_{h,0} \subset X_0$ of order $r \geq 2$ which are defined as

$$X_h := (S_h)^d\,, \qquad X_{h,0} := X_h \cap X_0\,, \tag{2.4}$$

with the scalar finite element space

$$S_h := \{\varphi \in H^1(\Omega) :\ \varphi\big|_K \circ F_K \in \mathbb{U}_r(\widehat{K}) \quad \forall K \in \mathcal{T}_h\}. \tag{2.5}$$

The space $\mathbb{U}_r(\widehat{K})$ depends on the shape of element K. We assume that it is chosen such that the local inf-sup condition on K is fulfilled. This means that there is a constant $\beta_0 > 0$ which is independent on K such that

$$\inf_{q \in \widetilde{M}_h(K)} \sup_{v \in X_{h,0}(K)} \frac{(\operatorname{div} v, q)_K}{|v|_{1,K}\|q\|_{0,K}} \geq \beta_0 \tag{2.6}$$

holds true for all $K \in \mathcal{T}_h$, where

$$\begin{array}{rcl} \widetilde{M}_h(K) & := & \{q \in M_{h,0} :\ q = 0 \text{ on } \Omega \setminus K\}\,, \\ X_{h,0}(K) & := & \{v \in X_h :\ v = 0 \text{ on } \Omega \setminus K\}\,, \end{array} \tag{2.7}$$

are the local pressure and velocity spaces associated with the element K, respectively. As an example, the choice $\mathbb{U}_r(\widehat{K}) = \mathbb{Q}_r(\widehat{K})$ on quadrilaterals or hexahedra guarantees that assumption (2.6) is satisfied; see [17]. A possible choice for $\mathbb{U}_r(\widehat{K})$ on triangles and tetrahedra is a suitable bubble enrichment of $\mathbb{P}_r(\widehat{K})$. For details, see [4, 7].

We use the [illegible] version of the local spaces where their definition is done via the reference transformation for both the velocity and the pressure. Thus, for a nonaffine reference mapping, the pressure will be approximated by functions which may be no polynomials. Note that also the [illegible] version of the local space is possible where only the velocity is defined using the reference transformation while the pressure will be approximated by piecewise polynomials from $\mathbb{P}_{r-1}(K)$ [7]. Our method will work also for the unmapped pressure definition. In the case of nonaffine reference transformations the mapped and unmapped version differ, but on families of regularly refined meshes we obtain the same optimal error estimates for both versions; see [1, 16].

Let $\widetilde{g}_h \in X_h$ be a suitable approximative extension of the boundary function g in (1.1) with

$$\langle\, \widetilde{g}_h, n \,\rangle_\Gamma = \langle\, g, n \,\rangle_\Gamma = 0. \tag{2.8}$$

Details will be given later on. Then, the discrete Oseen problem reads as follows: [illegible] $(u_h, p_h) \in (\widetilde{g}_h + X_{h,0}) \times M_{h,0}$ [illegible]

$$\begin{array}{rcll} a_h(u_h, v_h) + b(v_h, p_h) & = & f_h(v_h) & \forall\, v_h \in X_{h,0}\,, \\ b(u_h, q_h) & = & 0 & \forall\, q_h \in M_{h,0}\,, \end{array} \tag{2.9}$$

where $a_h(\cdot,\cdot)$ is either the bilinear form $a(\cdot,\cdot)$ in case of the Galerkin discretization or a stabilized version of it (see section 6.4). f_h denotes a bounded linear functional which is $f_h(\cdot) = (f, \cdot)_\Omega$ for the Galerkin discretization or a modified version if a stabilization technique is used. Together with (2.6) we assume that the pair $(X_{h,0}, M_{h,0})$ satisfies also the global inf-sup condition uniformly with respect to the mesh-size h. For example, in the case of quadrilaterals or hexahedra, the choice $\mathbb{U}_r(\widehat{K}) = \mathbb{Q}_r(\widehat{K})$

ensures this condition; see [17] for mapped and [7] for unmapped pressure elements. Thus, for a coercive and continuous bilinear form $a_h(\cdot,\cdot)$, the existence of a unique and stable solution (u_h, p_h) of problem (2.9) is guaranteed (see [7] and Theorem 4.1).

Finally, let us introduce some general notation. We denote by $\mathrm{card}(J)$ the number of elements of a finite set J. The dual space of a given space X will be denoted by X'. For a set $G \subset \mathbb{R}^m$, we denote by $\mathrm{int}(G)$ and $\overline{G}$ the interior and closure of G, respectively.

3. Construction of a quasi divergence free basis. Let φ_j, $j \in J$, denote the usual vector-valued Lagrange basis functions of the finite element space X_h, i.e.,

$$X_h = \mathrm{span}\{\varphi_j : j \in J\}\,, \tag{3.1}$$

where the indices $j \in J$ will be called [illegible] in what follows. Each node $j \in J$ is associated with a nodal point $a_j \in \overline{\Omega}$ of the grid $\mathcal{T}_h$. Let $J(\Gamma)$ and J_0, defined as

$$J(\Gamma) := \{j \in J : a_j \in \Gamma\}\,, \qquad J_0 := J \setminus J(\Gamma)\,, \tag{3.2}$$

denote the set of the boundary nodes and the set of the interior nodes, respectively, such that $X_{h,0} = \mathrm{span}\{\varphi_j : j \in J_0\}$.

Our construction of QDF basis functions ψ_j, $j \in \widetilde{J}$, of $\widetilde{X}_h$ is based on two projection operators which are applied to a subset of the basis functions φ_j of X_h. The first operator P_b maps into the space of the element bubble functions and the second operator P_f maps into a space of specially chosen face bubble functions.

Let us start with the definition of QDF functions and discretely divergence free functions. To this end, we need the discrete pressure subspace $\widetilde{M}_h \subset M_{h,0}$ defined by

$$\widetilde{M}_h := \{q \in M_{h,0} :\ q\big|_K \in L^2_0(K)\ \ \forall K \in \mathcal{T}_h\}\,. \tag{3.3}$$

Definition 3.1. [illegible] $v_h \in X_h$ [illegible] **quasi divergence free function** [illegible] **QDF function** [illegible] $\widetilde{M}_h \subset M_{h,0}$ [illegible]

$$b(v_h, q_h) = 0 \qquad \forall q_h \in \widetilde{M}_h\,. \tag{3.4}$$

[illegible]

$$\widetilde{X}_h := \{v_h \in X_h :\ b(v_h, q_h) = 0\ \ \forall q_h \in \widetilde{M}_h\} \quad \text{[illegible]} \quad \widetilde{X}_{h,0} := \widetilde{X}_h \cap X_0 \tag{3.5}$$

[illegible] **subspaces of the QDF functions** [illegible] X_h [illegible] $X_{h,0}$ [illegible]

$$V_h := \{v_h \in X_h :\ b(v_h, q_h) = 0\ \ \forall q_h \in M_h\} \quad \text{[illegible]} \quad V_{h,0} := V_h \cap X_0 \tag{3.6}$$

[illegible] **subspaces of the discretely divergence free functions** [illegible] X_h [illegible] $X_{h,0}$ [illegible]

In order to clarify the relation between QDF functions and discretely divergence free functions it is useful to introduce the spaces of the piecewise constant pressure functions

$$\overline{M}_h := \{q \in L^2(\Omega) :\ q\big|_K \in \mathbb{P}_0(K)\ \ \forall K \in \mathcal{T}_h\} \quad \text{and} \quad \overline{M}_{h,0} := \overline{M}_h \cap M_0\,. \tag{3.7}$$

Now, from $\widetilde{M}_h \subset M_{h,0} \subset M_h$ it follows that $V_h \subset \widetilde{X}_h$. For the pressure spaces we have the L^2-orthogonal decompositions

$$M_h = \overline{M}_h \oplus \widetilde{M}_h\,, \qquad M_{h,0} = \overline{M}_{h,0} \oplus \widetilde{M}_h, \qquad \text{and} \qquad \widetilde{M}_h = \bigoplus_{K\in\mathcal{T}_h} \widetilde{M}_h(K). \tag{3.8}$$

This implies that a function $v_h \in X_h$ belongs to the discretely divergence free subspace V_h if and only if $v_h \in \widetilde{X}_h$ and

$$b(v_h, q_h) = 0 \qquad \forall q_h \in \overline{M}_h\,. \tag{3.9}$$

It is easy to see that the condition (3.9) is equivalent to the property that, for each element $K \in \mathcal{T}_h$, the integral of the normal flux of v_h across the element boundary ∂K is zero, i.e.,

$$\sum_{E\in\mathcal{E}(K)} \left\langle\, v_h, n_E^K \,\right\rangle_E = 0 \qquad \forall K \in \mathcal{T}_h\,. \tag{3.10}$$

Using the fact that $b(v_h, 1) = 0$ for all functions $v_h \in X_{h,0}$, we get that

$$v_h \in V_{h,0} \qquad \text{if and only if} \qquad v_h \in \widetilde{X}_{h,0} \ \text{ and (3.10) is satisfied.} \tag{3.11}$$

[illegible] 3.2. It is important to emphasize that an essential assumption of our approach is the fact that the pressure space is discontinuous which implies the property (3.8) and the above mentioned relation between the QDF functions and the discretely divergence free functions. Furthermore, the question whether a function $v_h \in X_h$ is a QDF function can be checked elementwise. Therefore, the construction of a QDF basis of $\widetilde{X}_h$ can be done in an elementwise fashion.

3.1. Correction by element bubble functions. For a given element $K \in \mathcal{T}_h$, let

$$J(K) := \{j \in J :\ K \subset \mathrm{supp}(\varphi_j)\} \tag{3.12}$$

be the set of the local velocity nodes associated with K. Then, the local velocity space $X_h(K) \subset X_h$ related to element K is defined as

$$X_h(K) := \mathrm{span}\{\varphi_j : j \in J(K)\}\,. \tag{3.13}$$

For each $K \in \mathcal{T}_h$, we split the set of nodes $J(K)$ into the set $J_b(K)$ of [illegible], where the associated basis functions φ_j are so-called [illegible], and the remaining set $J_s(K)$ of so-called [illegible] living on the [illegible] of the grid $\mathcal{T}_h$, i.e.,

$$J_b(K) := \left\{j \in J(K) :\ \mathrm{supp}(\varphi_j) \subset \overline{K}\right\} \qquad \text{and} \qquad J_s(K) := J(K) \setminus J_b(K). \tag{3.14}$$

The associated local velocity spaces are defined as

$$X_b(K) := \mathrm{span}\{\varphi_j :\ j \in J_b(K)\}\,, \qquad X_s(K) := \mathrm{span}\{\varphi_j :\ j \in J_s(K)\}\,, \tag{3.15}$$

such that $X_h(K) = X_b(K) \oplus X_s(K)$. Note that, due to the global continuity of the basis functions φ_j on Ω, we have $v_h\big|_K \in (H_0^1(K))^d$ for all element bubble functions

$v_h \in X_b(K)$. Moreover, the space $X_b(K)$ coincides with $X_{h,0}(K)$ in (2.7). The global set of nodes can be split as $J = J_b \cup J_s$ with

$$J_b := \bigcup_{K \in \mathcal{T}_h} J_b(K) \qquad \text{and} \qquad J_s := J \setminus J_b = \bigcup_{K \in \mathcal{T}_h} J_s(K) \tag{3.16}$$

and the global velocity space is decomposed into $X_h = X_b \oplus X_s$, where

$$X_b := \operatorname{span}\{\varphi_j : \ j \in J_b\} \qquad \text{and} \qquad X_s := \operatorname{span}\{\varphi_j : \ j \in J_s\}. \tag{3.17}$$

Note that, due to the local support of the basis functions φ_j, $j \in J_b$, we have the following decomposition:

$$X_b = \bigoplus_{K \in \mathcal{T}_h} X_b(K), \tag{3.18}$$

which is orthogonal in L^2 and H^1.

In the following, we will construct basis functions $\widetilde{\varphi}_j$, $j \in \widetilde{J}$, of the subspace of QDF functions $\widetilde{X}_h$. The idea for the construction of $\widetilde{\varphi}_j$ is to subtract from the original basis function φ_j a correction consisting of element bubble functions. To this end, we define the operators

$$P_b : X \to X_b \qquad \text{and} \qquad Q_b : X \to \widetilde{M}_h$$

as follows. For a given function $w \in X = (H^1(\Omega))^d$, the pair $(P_b w, Q_b w)$ is defined to be the solution of the problem:

$(P_b w, Q_b w) \in X_b \times \widetilde{M}_h$

$$\begin{aligned} a_h(P_b w, v_h) \, + \, b(v_h, Q_b w) &= a_h(w, v_h) && \forall \, v_h \in X_b \,, \\ b(P_b w, q_h) &= b(w, q_h) && \forall \, q_h \in \widetilde{M}_h \,. \end{aligned} \tag{3.19}$$

Using the orthogonal splittings (3.18) and (3.8), we see that problem (3.19) is equivalent to $\operatorname{card}(\mathcal{T}_h)$ many local problems, i.e., $(P_b w, Q_b w)$ can be written as a sum

$$(P_b w, Q_b w) = \sum_{K \in \mathcal{T}_h} (P_b^K w, Q_b^K w)$$

with local operators $P_b^K : X \to X_b(K)$ and $Q_b^K : X \to \widetilde{M}_h(K)$ associated with the element $K \in \mathcal{T}_h$ such that, for a given $w \in X$, the pair $(P_b^K w, Q_b^K w)$ is defined as the solution of the following local problem:

$(P_b^K w, Q_b^K w) \in X_b(K) \times \widetilde{M}_h(K)$

$$\begin{aligned} a_h(P_b^K w, v_h) \, + \, b(v_h, Q_b^K w) &= a_h(w, v_h) && \forall \, v_h \in X_b(K) \,, \\ b(P_b^K w, q_h) &= b(w, q_h) && \forall \, q_h \in \widetilde{M}_h(K) \,. \end{aligned} \tag{3.20}$$

Since $X_b(K)$ coincides with the space $X_{h,0}(K)$ from (2.7), the assumption (2.6) implies the inf-sup condition for problem (3.20). Thus, from the standard FEM-theory for mixed problems [4] we get that, for each $K \in \mathcal{T}_h$, the local problem (3.20) has a unique solution which continuously depends on the local norm $\|w\|_{1,K}$. Hence, the global problem (3.19) also has a unique solution which depends continuously on the global norm $\|w\|_{1,\Omega}$.

Now, using the operator P_b, we define the following functions $\widetilde{\varphi}_j$ for the index set J_s of all skeleton nodes:

$$\widetilde{\varphi}_j := (I - P_b)\varphi_j \qquad \forall j \in J_s, \tag{3.21}$$

where I denotes the identity operator. Since $(P_b\varphi_j)\big|_{\partial K} = 0$ for all $K \in \mathcal{T}_h$, we get that $\widetilde{\varphi}_j$ coincides with φ_j on the skeleton of the grid $\mathcal{T}_h$, i.e.,

$$\widetilde{\varphi}_j(x) := \varphi_j(x) \qquad \forall x \in \bigcup_{K\in\mathcal{T}_h} \partial K \quad \forall j \in J_s. \tag{3.22}$$

Therefore, all the $\widetilde{\varphi}_j$ with $j \in J_s$ are linearly independent functions in X_h. Moreover, from the second equation in (3.19), it follows that $\widetilde{\varphi}_j \in \widetilde{X}_h$. Thus, the functions $\widetilde{\varphi}_j$, $j \in J_s$, are the basis functions of a subspace $\widetilde{X}_s \subset \widetilde{X}_h$ defined by

$$\widetilde{X}_s := \operatorname{span}\{\widetilde{\varphi}_j : \ j \in J_s\}\,, \tag{3.23}$$

and the operator $I - P_b : X_s \to \widetilde{X}_s$ is an isomorphism.

However, for $j \in J_b$, the solution of problem (3.19) with $w = \varphi_j$ is $(P_b\varphi_j, Q_b\varphi_j) = (\varphi_j, 0) \in X_b \times \widetilde{M}_h$ such that $\widetilde{\varphi}_j$ defined by (3.21) would be zero. Therefore, in general, we have to construct for each element $K \in \mathcal{T}_h$ an additional set of basis functions $\widetilde{\varphi}_j$, $j \in \widetilde{J}_b(K)$, of the subspace $\widetilde{X}_b(K)$ of the quasi divergence free element bubble functions defined as

$$\widetilde{X}_b(K) := \{v_h \in X_b(K) : \ b(v_h, q_h) = 0 \ \ \forall q_h \in \widetilde{M}_h(K)\}. \tag{3.24}$$

Here "in general" means the case when the polynomial degree r of the velocity space is larger than 2. For example, in the case of quadrilateral elements with Q_2-elements for the velocity and discontinuous P_1-elements for the pressure we have for all elements K that $\dim(\widetilde{X}_b(K)) = \dim(X_b(K)) - \dim(\widetilde{M}_h(K)) = 0$, i.e., no basis functions have to be constructed. For a given element $K \in \mathcal{T}_h$, the problem to determine basis functions $\widetilde{\varphi}_j$ such that

$$\widetilde{X}_b(K) = \operatorname{span}\{\widetilde{\varphi}_j : \ j \in \widetilde{J}_b(K)\} \tag{3.25}$$

is equivalent to an algebraic problem of a small fixed size. This problem consists of the computation of basis vectors of the kernel of the matrix $B^K \in \mathbb{R}^{\dim(\widetilde{M}_h(K))\times\dim(X_b(K))}$ defined as

$$B^K_{i,j} := b(\varphi_j, \chi_i) \qquad \forall i \in \widetilde{I}(K), \ j \in J_b(K), \tag{3.26}$$

where the χ_i, $i \in \widetilde{I}(K)$, denote the basis functions of the local pressure space $\widetilde{M}_h(K)$. Basis vectors of $\ker(B^K)$ could be obtained, for instance, by means of a singular value decomposition of the matrix B^K. In the following, let us assume that the basis functions $\widetilde{\varphi}_j$, $j \in \widetilde{J}_b(K)$, are available for all elements $K \in \mathcal{T}_h$ in the sense that coefficients $c^K_{j,i} \in \mathbb{R}$ are known such that

$$\widetilde{\varphi}_j = \sum_{i\in J_b(K)} c^K_{j,i}\varphi_i \qquad \forall j \in \widetilde{J}_b(K)\,. \tag{3.27}$$

Then, the global space of the quasi divergence free element bubble functions $\widetilde{X}_b$ defined as

$$\widetilde{X}_b := \{v_h \in X_b : \ b(v_h, q_h) = 0 \ \ \forall q_h \in \widetilde{M}_h\}\,, \tag{3.28}$$

can be represented by means of basis functions $\widetilde{\varphi}_j$ as follows:

$$\widetilde{X}_b = \mathrm{span}\{\widetilde{\varphi}_j : \ j \in \widetilde{J}_b\}\,, \qquad \text{where} \qquad \widetilde{J}_b := \bigcup_{K\in\mathcal{T}_h} \widetilde{J}_b(K). \tag{3.29}$$

Since all functions $v_h \in \widetilde{X}_b$ vanish on the element boundaries, the condition (3.10) is satisfied which implies that

$$\widetilde{X}_b = V_b := V_h \cap X_b \qquad \text{and} \qquad \widetilde{X}_b(K) = V_b(K) := V_h \cap X_b(K) \quad \forall K \in \mathcal{T}_h\,, \tag{3.30}$$

i.e., all quasi divergence free element bubble functions are discretely divergence free.

Now we are in the position to give the basis representation of the subspace of the QDF functions $\widetilde{X}_h$:

$$\widetilde{X}_h = \mathrm{span}\{\widetilde{\varphi}_j : \ j \in \widetilde{J}\}, \qquad \text{where} \qquad \widetilde{J} := J_s \cup \widetilde{J}_b. \tag{3.31}$$

Using the space $\widetilde{X}_s$ defined in (3.23), we can decompose $\widetilde{X}_h$ in the direct sum

$$\widetilde{X}_h = \widetilde{X}_s \oplus V_b, \tag{3.32}$$

where $V_b = V_h \cap X_b$ is the space of the discretely divergence free element bubble functions. In fact, by construction it is clear that the space $\widetilde{X}_s \oplus V_b$ is a subspace of $\widetilde{X}_h$ and if we consider the dimensions we obtain

$$\begin{aligned} \dim(\widetilde{X}_s \oplus V_b) &= \mathrm{card}(J_s) + \sum_{K\in\mathcal{T}_h} \Big(\dim(X_b(K)) - \dim(\widetilde{M}_h(K))\Big) \\ &= \dim(X_s) + \dim(X_b) - \sum_{K\in\mathcal{T}_h} \dim(\widetilde{M}_h(K)) = \dim(\widetilde{X}_h). \end{aligned}$$

Similarly we can decompose X_h in the direct sum

$$X_h = \widetilde{X}_s \oplus X_b\,. \tag{3.33}$$

In fact, an arbitrary function $v_h \in X_h$ has a representation as $v_h = v_s + \widetilde{v}_b$ with $v_s \in X_s$ and $\widetilde{v}_b \in X_b$. Using $\widetilde{v}_s := (I - P_b)v_s \in \widetilde{X}_s$ we get the representation $v_h = \widetilde{v}_s + v_b$ with $v_b := \widetilde{v}_b + P_b v_s \in X_b$. This shows that $X_h \subset \widetilde{X}_s \oplus X_b$. By construction we have $\widetilde{X}_s \oplus X_b \subset X_h$.

3.2. Correction by face bubble functions. The aim of this subsection is to perform a further decomposition of the subspace $\widetilde{X}_s$ in the form

$$\widetilde{X}_s = V_s \oplus \widetilde{X}_f,$$

where $V_s \subset V_h$ is a subspace of discretely divergence free functions which should be as large as possible. An advantage of using V_h-subspaces is that velocity components lying in those subspaces do not have any couplings with the pressure in the discrete Oseen problem (2.9).

The property that a QDF basis function $\psi_j \in \widetilde{X}_s$ belongs to V_h is equivalent to the condition that ψ_j satisfies the condition (3.10). The idea to fulfill (3.10) for ψ_j is

to start from the already known QDF basis function $\widetilde{\varphi}_j$ and to modify it in such a way that on each $(d-1)$-dimensional face $E \in \mathcal{E}_h$ the normal flux $\langle \psi_j, n_E \rangle_E$ becomes zero. To this end we modify the normal flux of $\widetilde{\varphi}_j$ across E by subtracting a multiple of a properly chosen face bubble function associated with the face E. This does not change the normal fluxes for all other faces.

In the following, we will describe how to choose these face bubble functions. Recall that each node $j \in J$ is associated with a nodal point $a_j \in \overline{\Omega}$. Then, for each face $E \in \mathcal{E}_h$, the set

$$J_0(E) := \{j \in J_s : \ a_j \in \mathrm{int}(E)\} \tag{3.34}$$

denotes the set of all skeleton nodes that correspond to the interior of the $(d-1)$-dimensional face E. Among all the nodes $j \in J_0(E)$ we choose a special [illegible] E, denoted by $j_f(E)$, that realizes the maximum absolute normal flux of the corresponding basis functions, i.e.,

$$|\langle \varphi_{j_f(E)}, n_E \rangle_E| = \max_{j \in J_0(E)} |\langle \varphi_j, n_E \rangle_E|. \tag{3.35}$$

Note that, due to the fact that $P_b \varphi_j\big|_E = 0$ for all $j \in J_0(E)$, the analogous equation holds true for the QDF basis functions $\widetilde{\varphi}_{j_f(E)}$ and $\widetilde{\varphi}_j$, respectively. To simplify the notation we introduce the abbreviations

$$\varphi_E := \varphi_{j_f(E)}, \qquad \widetilde{\varphi}_E := \widetilde{\varphi}_{j_f(E)} \qquad \forall E \in \mathcal{E}_h. \tag{3.36}$$

The set $J_f(K)$ of all flux nodes of an element $K \in \mathcal{T}_h$ and the set $\widetilde{J}_s(K)$ of the remaining skeleton nodes are defined by

$$J_f(K) := \{j_f(E) : \ E \in \mathcal{E}(K)\} \qquad \text{and} \qquad \widetilde{J}_s(K) := J_s(K) \setminus J_f(K). \tag{3.37}$$

Let J_f and $\widetilde{J}_s$ denote the corresponding global sets of nodes given by

$$J_f := \bigcup_{K \in \mathcal{T}_h} J_f(K) \qquad \text{and} \qquad \widetilde{J}_s := \bigcup_{K \in \mathcal{T}_h} \widetilde{J}_s(K) = J_s \setminus J_f\,, \tag{3.38}$$

and $\widetilde{X}_f$ the subspace of $\widetilde{X}_s$ associated with the flux nodes, i.e.,

$$\widetilde{X}_f := \mathrm{span}\{\widetilde{\varphi}_j : \ j \in J_f\}. \tag{3.39}$$

Then, we introduce the operator $P_f : X \to \widetilde{X}_f$ such that for a vector-valued function $w \in X = (H^1(\Omega))^d$ the discrete function $P_f w \in \widetilde{X}_f$ is defined as

$$P_f w := \sum_{E \in \mathcal{E}_h} \frac{\langle w, n_E \rangle_E}{\langle \widetilde{\varphi}_E, n_E \rangle_E} \widetilde{\varphi}_E. \tag{3.40}$$

It is easy to see that this operator has the property

$$\langle P_f w, n_E \rangle_E = \langle w, n_E \rangle_E \qquad \forall E \in \mathcal{E}_h, \quad \forall w \in (H^1(\Omega))^d. \tag{3.41}$$

Now, starting from the basis functions $\widetilde{\varphi}_j$ of $\widetilde{X}_h$ with $j \in \widetilde{J} = (\widetilde{J}_s \cup J_f) \cup \widetilde{J}_b$, we define new basis functions ψ_j of $\widetilde{X}_h$ as

$$\psi_j := \begin{cases} \widetilde{\varphi}_j - P_f \widetilde{\varphi}_j, & j \in \widetilde{J}_s, \\ \widetilde{\varphi}_j, & j \in J_f \cup \widetilde{J}_b. \end{cases} \tag{3.42}$$

Note that the function $\psi_j = \widetilde{\varphi}_j - P_f\widetilde{\varphi}_j$ would be zero for $j \in J_f$ and it would be equal to $\widetilde{\varphi}_j$ for $j \in \widetilde{J}_b$. One easily verifies that the functions $\psi_j \in \widetilde{X}_s$ with $j \in \widetilde{J}_s$ are linearly independent. Moreover, the property (3.41) implies that $\psi_j \in V_h$ for all $j \in \widetilde{J}_s$. Thus, we get the desired decomposition $\widetilde{X}_s = V_s \oplus \widetilde{X}_f$ with the space

$$V_s := \operatorname{span}\{\psi_j : \ j \in \widetilde{J}_s\} \subset V_h.$$

Using (3.32) with $V_b = \operatorname{span}\{\widetilde{\varphi}_j; \ j \in \widetilde{J}_b\} = \operatorname{span}\{\psi_j; \ j \in \widetilde{J}_b\}$ we finally get the decomposition of the subspace $\widetilde{X}_h$ of the quasi divergence free functions

$$\widetilde{X}_h = \widetilde{X}_s \oplus V_b \qquad \text{with} \qquad \widetilde{X}_s = V_s \oplus \widetilde{X}_f = \operatorname{span}\{\psi_j : \ j \in J_s = \widetilde{J}_s \cup J_f\}\,, \tag{3.43}$$

where both spaces V_s and V_b are subspaces of the space V_h of the discretely divergence free functions. That means that all the basis functions ψ_j with $j \in \widetilde{J}_s \cup \widetilde{J}_b$ do not have any couplings with the pressure in the Oseen problem (2.9).

4. Decomposition of the saddle point problem. In this section, we show how the discrete Oseen problem (2.9) can be split into two subproblems using the decomposition (3.33) of the velocity space $X_h = \widetilde{X}_s \oplus X_b$. At first we explain how the discrete extension $\widetilde{g}_h \in X_h$ of the boundary function $g : \Gamma \to \mathbb{R}^d$ is chosen. Let

$$g_h = \sum_{j \in J(\Gamma)} g_j \varphi_j \in X_h \qquad \text{with} \qquad g_j \in \mathbb{R}$$

be an interpolate of g using only boundary nodes $j \in J(\Gamma)$ and satisfying the solvability condition $\langle\, g_h, n \,\rangle_\Gamma = \langle\, g, n \,\rangle_\Gamma = 0$. Then we choose

$$\widetilde{g}_h := \sum_{j \in J(\Gamma)} g_j \widetilde{\varphi}_j = (I - P_b) g_h \in \widetilde{X}_s, \tag{4.1}$$

where $\widetilde{g}_h \in \widetilde{X}_s$ follows from the fact that $J(\Gamma) \subset J_s$.

It is easy to see that the discrete solution u_h of (2.9) is a solution of the following [illegible] of the discrete Oseen problem:
[illegible] $(u_h, \overline{p}_h) \in (\widetilde{g}_h + \widetilde{X}_{h,0}) \times \overline{M}_{h,0}$ [illegible]

$$\begin{array}{lcll} a_h(u_h, \widetilde{v}_h) \,+\, b(\widetilde{v}_h, \overline{p}_h) & = & f_h(\widetilde{v}_h) & \forall\, \widetilde{v}_h \in \widetilde{X}_{h,0}, \\ b(u_h, \overline{q}_h) & = & 0 & \forall\, \overline{q}_h \in \overline{M}_{h,0}\,. \end{array} \tag{4.2}$$

Now, the splitting (3.32) of $\widetilde{X}_h$ as the direct sum $\widetilde{X}_h = \widetilde{X}_s \oplus V_b$ implies that the discrete solution $u_h \in \widetilde{g}_h + \widetilde{X}_{h,0}$ can be uniquely represented in the form

$$u_h = \widetilde{u}_s + u_b = (\widetilde{g}_h + \widetilde{u}_{s,0}) + u_b \tag{4.3}$$

with $\widetilde{u}_s \in \widetilde{X}_s$, $u_b \in V_b$ and $\widetilde{u}_{s,0} \in \widetilde{X}_{s,0} := \widetilde{X}_s \cap X_0$. From the definition of the operator $P_b : X \to X_b$ in (3.19), it follows by taking the test function $v_h = v_b \in V_b$ and using $b(v_b, p_h) = 0$ that

$$a_h(v_s - P_b v_s, v_b) = 0 \qquad \forall v_s \in X_s\,,\ v_b \in V_b\,.$$

Since the operator $I - P_b : X_s \to \widetilde{X}_s$ is an isomorphism we get the property

$$a_h(\widetilde{v}_s, v_b) = 0 \qquad \forall \widetilde{v}_s \in \widetilde{X}_s\,,\ v_b \in V_b\,, \tag{4.4}$$

which says that the subspaces $\widetilde{X}_s$ and V_b satisfy a certain orthogonality condition. This property is important for the decomposition of the Oseen problem (2.9) into smaller subproblems which will be described in the following.

If we insert the splitting $u_h = \widetilde{u}_s + u_b$ into the first equation of (2.9) and take the test function $v_h = v_b \in V_b \subset X_{h,0}$ we get, by means of $b(v_b, p_h) = 0$ and (4.4), the following problem for u_b:

$u_b \in V_b$:

$$a_h(u_b, v_b) = f_h(v_b) \qquad \forall v_b \in V_b\,. \tag{4.5}$$

That means that the element bubble part u_b of the solution u_h can be computed independently from the remaining part $\widetilde{u}_s$. Moreover, this computation can be done separately on each element $K \in \mathcal{T}_h$ inside of the assembling process of the stiffness matrix. If the basis functions $\psi_j = \widetilde{\varphi}_j$, $j \in \widetilde{J}_b(K)$, of V_b are available, the part of the u_b-problem (4.5) on element K can be calculated by solving the equivalent $\widetilde{n}_b(K) \times \widetilde{n}_b(K)$ linear system of equations where $\widetilde{n}_b(K) := \dim(X_b(K)) - \dim(\widetilde{M}_h(K))$. In the special case that the space X_h corresponds to conforming $\mathbb{Q}_2$-elements and M_h corresponds to discontinuous $\mathbb{P}_1$ elements, nothing is to do for solving (4.5) since $V_b = \{0\}$ and $u_b = 0$. If the basis functions ψ_j of V_b are not available, we can compute the part $u_b^K \in X_b(K)$ on an element K of $u_b = \sum_{K \in \mathcal{T}_h} u_b^K \in X_b$ corresponding to the splitting (3.18) by solving the mixed local problem associated to (4.5):

$(u_b^K, \theta^K) \in X_b(K) \times \widetilde{M}_h(K)$:

$$\begin{array}{lcll} a_h(u_b^K, v_h) + b(v_h, \theta^K) & = & f_h(v_h) & \forall\, v_h \in X_b(K)\,, \\ b(u_b^K, q_h) & = & 0 & \forall\, q_h \in \widetilde{M}_h(K)\,. \end{array} \tag{4.6}$$

Now, we want to derive the problem for the part $\widetilde{u}_s$ of the solution u_h. Due to the decomposition (3.8) of the pressure space $M_{h,0}$ the pressure solution p_h can be uniquely split in the form $p_h = \overline{p}_h + \widetilde{p}_h$ with $\overline{p}_h \in \overline{M}_{h,0}$ and $\widetilde{p}_h \in \widetilde{M}_h$. If we take in (2.9) the test functions $v_h = \widetilde{v}_s \in \widetilde{X}_{s,0} = \widetilde{X}_s \cap X_0 \subset X_{h,0}$ and $q_h = \overline{q}_h \in \overline{M}_{h,0} \subset M_{h,0}$ and if we use the properties $b(\widetilde{v}_s, \widetilde{p}_h) = 0$ and $b(u_b, \overline{q}_h) = 0$ we get the following for $\widetilde{u}_s$ and $\overline{p}_h$:

$(\widetilde{u}_s, \overline{p}_h) \in (\widetilde{g}_h + \widetilde{X}_{s,0}) \times \overline{M}_{h,0}$:

$$\begin{array}{lcll} a_h(\widetilde{u}_s, \widetilde{v}_s) + b(\widetilde{v}_s, \overline{p}_h) & = & f_h(\widetilde{v}_s) - a_h(u_b, \widetilde{v}_s) & \forall\, \widetilde{v}_s \in \widetilde{X}_{s,0}\,, \\ b(\widetilde{u}_s, \overline{q}_h) & = & 0 & \forall\, \overline{q}_h \in \overline{M}_{h,0}\,. \end{array} \tag{4.7}$$

THEOREM 4.1. $a_h(\cdot,\cdot)$ $C_1 = C_1(\nu, \beta)$ $C_2 = C_2(\nu, \beta)$

$$a_h(v_h, v_h) \geq C_1 |v_h|_{1,\Omega}^2 \qquad \forall\, v_h \in X_{h,0}\,, \tag{4.8}$$

$$a_h(v_h, w_h) \leq C_2 |v_h|_{1,\Omega} |w_h|_{1,\Omega} \qquad \forall\, v_h, w_h \in X_h. \tag{4.9}$$

(4.6) (4.7) $(u_b^K, \theta^K) \in X_b(K) \times \widetilde{M}_h(K)$ $K \in \mathcal{T}_h$ $(\widetilde{u}_s, \overline{p}_h) \in (\widetilde{g}_h + \widetilde{X}_{s,0}) \times \overline{M}_{h,0}$ $u_h \in \widetilde{g}_h + \widetilde{X}_{h,0}$ (2.9) (4.2)

$$u_h = \widetilde{u}_s + \sum_{K \in \mathcal{T}_h} u_b^K \in \widetilde{g}_h + X_{h,0}\,. \tag{4.10}$$

[illegible] $p_h \in M_{h,0}$ [illegible] (2.9) [illegible] $p_h = \overline{p}_h + \widetilde{p}_h$ [illegible] $\overline{p}_h \in \overline{M}_{h,0}$ [illegible] (4.7) [illegible] $\widetilde{p}_h = \sum_{K \in \mathcal{T}_h} \widetilde{p}_h^K \in \widetilde{M}_h$ [illegible] $\widetilde{p}_h^K \in \widetilde{M}_h(K)$ [illegible]

Find $\widetilde{p}_h^K \in \widetilde{M}_h(K)$ such that

$$b(v_b, \widetilde{p}_h^K) = f_h(v_b) - a_h(u_h, v_b) \qquad \forall\, v_b \in X_b(K)\,. \tag{4.11}$$

[illegible] $K \in \mathcal{T}_h$ [illegible] (4.11) [illegible] $\widetilde{p}_h^K \in \widetilde{M}_h(K)$ [illegible] $(u_h, \overline{p}_h)$ [illegible] (4.2) [illegible]

[illegible]. Due to the assumption (2.6) and the fact that $X_b(K) = X_{h,0}(K)$, the spaces $(X_b(K), \widetilde{M}_h(K))$ fulfill the inf-sup condition. Moreover, since $X_b(K) \subset X_{h,0}$, the bilinear form $a_h(\cdot,\cdot)$ is coercive on $X_b(K) \times X_b(K)$. Therefore, the mixed problem (4.6) has a unique solution (u_b^K, θ^K) for each $K \in \mathcal{T}_h$.

In the following, we will show that the pair $(\widetilde{X}_{s,0}, \overline{M}_{h,0})$ satisfies the inf-sup condition. We will restrict the presentation to the case of quadrilateral or hexahedral elements where the velocity space is defined by (2.5) with $\mathbb{U}_r(\widehat{K}) = \mathbb{Q}_r(\widehat{K})$. In [17, Lemma 7], it is proved that, for each $q_h \in \overline{M}_{h,0}$, there exists a $\bar{v}_h \in \bar{X}_h$ such that

$$b(\bar{v}_h, q_h) \geq \bar{\beta}_1 |\bar{v}_h|_{1,\Omega} \|q_h\|_{0,\Omega}\,, \tag{4.12}$$

where $\bar{\beta}_1$ is a positive h-independent constant and $\bar{X}_h \subset X_0$ is a space that consists of vector-valued Q_1-functions enriched with special Q_2-functions associated with the element faces. Since the space $\bar{X}_h$ does not contain any bubble functions of X_b we have, due to $r \geq 2$, that $\bar{X}_h \subset X_{s,0}$. Thus, it holds $\bar{v}_h \in X_{s,0}$ which implies that $v_h := \bar{v}_h - P_b \bar{v}_h \in \widetilde{X}_{s,0}$. Since $P_b \bar{v}_h \in X_b$, the functions $\bar{v}_h$ and v_h coincide on each element face $E \in \mathcal{E}_h$. Therefore, for each piecewise constant $q_h \in \overline{M}_{h,0}$, it holds that

$$b(\bar{v}_h, q_h) = -\sum_{K \in \mathcal{T}_h} \left\{ q_h\big|_K \sum_{E \in \mathcal{E}(K)} \left\langle\, \bar{v}_h, n_E^K \,\right\rangle_E \right\} = b(v_h, q_h)\,. \tag{4.13}$$

If we apply Proposition II.1.3 of [4] to the solution $P_b^K w = P_b w\big|_K$ of the mixed problem (3.20), we get by means of (4.8) and (4.9) the estimate

$$|P_b w|_{1,K} \leq C_3 |w|_{1,K} \qquad \forall\, w \in X_h\,,\ \forall\, K \in \mathcal{T}_h\,, \tag{4.14}$$

where $C_3 := (C_2/C_1 + 1)\sqrt{d}/\beta_0 + C_2/C_1$ is a constant independent of K and β_0 the local inf-sup-constant in (2.6). Now, we conclude from (4.14) that

$$|v_h|_{1,\Omega} = |\bar{v}_h - P_b \bar{v}_h|_{1,\Omega} \leq (1 + C_3) |\bar{v}_h|_{1,\Omega}\,,$$

which implies by means of (4.12) and (4.13)

$$b(v_h, q_h) = b(\bar{v}_h, q_h) \geq \bar{\beta}_1 |\bar{v}_h|_{1,\Omega} \|q_h\|_{0,\Omega} \geq \frac{\bar{\beta}_1}{1 + C_3} |v_h|_{1,\Omega} \|q_h\|_{0,\Omega}\,,$$

i.e., the spaces $(\widetilde{X}_{s,0}, \overline{M}_{h,0})$ satisfy the inf-sup condition. In the case of simplicial elements, where the local space $\mathbb{U}_r(\widehat{K})$ of the velocity is a suitable enrichment of the space $\mathbb{P}_r(\widehat{K})$, the inf-sup condition can be proved analogously with an auxiliary space

$\bar{X}_h$ that consists of P_1-functions and special face bubble functions (see (2.3) and (2.32) in sect. II.2 of [7]).

Due to assumptions (4.8) and (4.9), the bilinear form $a_h(\cdot,\cdot)$ is coercive and continuous also on $\widetilde{X}_{s,0} \times \widetilde{X}_{s,0}$. Thus, the inf-sup condition for the spaces $(\widetilde{X}_{s,0}, \overline{M}_{h,0})$ implies that the reduced Oseen problem (4.7) has a unique solution $(\widetilde{u}_s, \overline{p}_h)$.

Now, let $\widetilde{u}_s$ and $u_b = \sum_{K\in\mathcal{T}_h} u_b^K$ be the unique solutions of (4.7) and (4.5), respectively. Further, let $\overline{p}_h$ and $\widetilde{p}_h = \sum_{K\in\mathcal{T}_h} \widetilde{p}_h^K$ be the unique solutions of (4.7) and (4.11), respectively. Note that the unique solvability of (4.11) follows from the assumption (2.6) that the local spaces $X_b(K) = X_{h,0}(K)$ and $\widetilde{M}_h(K)$ fulfill the inf-sup condition. In the following, we will show that $u_h := \widetilde{u}_s + u_b$ and $p_h := \overline{p}_h + \widetilde{p}_h$ are the solutions of the original Oseen problem (2.9). Let v_b be an arbitrary element of V_b. Then, from $a_h(\widetilde{u}_s, v_b) = 0$, see (4.4), $b(v_b, p_h) = 0$ and (4.5) we get

$$a_h(u_h, v_b) + b(v_b, p_h) = f_h(v_b) \qquad \forall\, v_b \in V_b\,. \tag{4.15}$$

Using $a_h(\widetilde{v}_s, u_b) = 0$, $b(\widetilde{v}_s, \widetilde{p}_h) = 0$ and the first equation of (4.7) we obtain

$$a_h(u_h, \widetilde{v}_s) + b(\widetilde{v}_s, p_h) = f_h(\widetilde{v}_s) \qquad \forall\, \widetilde{v}_s \in \widetilde{X}_{s,0}\,. \tag{4.16}$$

The decomposition $\widetilde{X}_h = \widetilde{X}_s \oplus V_b$ and (4.15) and (4.16) yield

$$a_h(u_h, \widetilde{v}_h) + b(\widetilde{v}_h, p_h) = f_h(\widetilde{v}_h) \qquad \forall\, \widetilde{v}_h \in \widetilde{X}_{h,0}\,. \tag{4.17}$$

The equations (4.11) and $b(v_b, \overline{p}_h) = 0$ imply that

$$a_h(u_h, v_b) + b(v_b, p_h) = f_h(v_b) \qquad \forall\, v_b \in X_b(K)\,. \tag{4.18}$$

Due to the decomposition (3.18) of X_b, (4.18) also holds for an arbitrary $v_b \in X_b$. Thus, it follows that u_h and p_h satisfy the first equation of the Oseen problem (2.9). From the second equation of (4.7) we get with $b(u_b, \overline{q}_h) = 0$ the equation

$$b(u_h, \overline{q}_h) = 0 \qquad \forall \overline{q}_h \in \overline{M}_{h,0}\,.$$

By definition we have $\widetilde{u}_s \in \widetilde{X}_s \subset \widetilde{X}_h$ and $u_b \in V_b \subset \widetilde{X}_h$ which implies $u_h \in \widetilde{X}_h$, i.e.,

$$b(u_h, \widetilde{q}_h) = 0 \qquad \forall \widetilde{q}_h \in \widetilde{M}_h\,.$$

Thus, from the decomposition $M_{h,0} = \overline{M}_{h,0} \oplus \widetilde{M}_h$ it follows that u_h fulfills also the second equation of the Oseen problem (2.9). This proves the representations of u_h and p_h since problem (2.9) has a unique solution.

If we take in (2.9) the test functions $v_h \in \widetilde{X}_{h,0}$ and $q_h \in \overline{M}_{h,0}$ we easily see that $(u_h, \overline{p}_h)$ is a solution of (4.2). The inf-sup condition for the pair $(\widetilde{X}_{h,0}, \overline{M}_{h,0})$ follows immediately from the inf-sup condition for $(\widetilde{X}_{s,0}, \overline{M}_{h,0})$. This implies that the problem (4.2) has a unique solution which completes the proof of Theorem 4.1. □

REMARK 4.2. The assumptions (4.8) and (4.9) in Theorem 4.1 are satisfied if $\operatorname{div}\beta = 0$ and the Galerkin discretization is used, i.e., if the bilinear form $a_h(\cdot,\cdot)$ is identical to $a(\cdot,\cdot)$. In the case, where $a_h(\cdot,\cdot)$ is derived from a streamline diffusion stabilization method, these assumptions are discussed, e.g., in [6].

In the following, we analyze the structure of the QDF formulation (4.2) of the Oseen problem by exploiting the fact that the space $\widetilde{X}_h$ can be decomposed in the

direct sum $\widetilde{X}_h = V_b \oplus V_s \oplus \widetilde{X}_f$; see (3.43). The solution u_h of (4.2) can be decomposed as

$$u_h = \widetilde{g}_h + u_{h,0} \qquad \text{with} \qquad u_{h,0} \in \widetilde{X}_{h,0} = \operatorname{span}\{\psi_j : j \in \widetilde{J}_0\}, \quad \widetilde{J}_0 := \widetilde{J} \setminus J(\Gamma),$$

where $\widetilde{J} = \widetilde{J}_b \cup J_s$; see (3.31). By the disjoint decomposition (3.38) of the index set $J_s = \widetilde{J}_s \cup J_f$ the following disjoint decomposition of the index set $\widetilde{J}_0$ is induced:

$$\widetilde{J}_0 = \widetilde{J}_b \cup \widetilde{J}_{s,0} \cup J_{f,0} \qquad \text{with} \qquad \widetilde{J}_{s,0} := \widetilde{J}_s \setminus J(\Gamma) \quad J_{f,0} := J_f \setminus J(\Gamma).$$

Using this decomposition, the coefficient vector $\underline{u}^{h,0} = (u_j^{h,0})$ of $u_{h,0} = \sum_{j \in \mathcal{J}_0} u_j^{h,0} \psi_j$ can be partitioned into the two block vectors

$$\underline{u}^{h,0} = \begin{pmatrix} \underline{u}^{n,0} \\ \underline{u}^{f,0} \end{pmatrix} \qquad \text{with} \qquad \underline{u}^{n,0} := (u_j^{h,0})_{j \in \mathcal{J}_b \cup \mathcal{J}_{s,0}}, \quad \underline{u}^{f,0} := (u_j^{h,0})_{j \in J_{f,0}}.$$

Let $\overline{I}$ be the index set of the elements $K \in \mathcal{T}_h$ such that $\mathcal{T}_h = \{K_i : i \in \overline{I}\}$. Then, the piecewise constant pressure solution $\overline{p}_h$ of (4.7) corresponds to the coefficient vector $\underline{p}$ with

$$\underline{p} := (p_i)_{i \in \overline{I}} \in \mathbb{R}^{\operatorname{card}(\overline{I})}, \qquad \text{where} \qquad \overline{p}_h = \sum_{i \in \overline{I}} p_i \chi_0^{K_i}.$$

Here, $\chi_0^{K_i}$ denotes the function in $\overline{M}_h$ which is 1 on K_i and 0 otherwise. Due to the fact that $\psi_j \in V_h$ for all nonflux nodes $j \in \widetilde{J}_b \cup \widetilde{J}_{s,0}$, the QDF formulation (4.2) corresponds to a block system of the following structure:

$$\begin{pmatrix} A^{nn} & A^{nf} & 0 \\ A^{fn} & A^{ff} & (B^f)^T \\ 0 & B^f & 0 \end{pmatrix} \begin{pmatrix} \underline{u}^{n,0} \\ \underline{u}^{f,0} \\ \underline{p} \end{pmatrix} = \begin{pmatrix} \underline{r}^{n,0} \\ \underline{r}^{f,0} \\ \underline{s} \end{pmatrix}. \tag{4.19}$$

We see that only the QDF basis functions ψ_j for flux nodes $j \in J_{f,0}$ are coupled with the piecewise constant pressure. In contrast to the original basis functions, this is an important computational advantage for the construction of cheap block Gauss–Seidel smoothers since the structure of (4.19) allows, in principle, to apply a point Gauss–Seidel smoother for all nonflux nodes $j \in \widetilde{J}_0 \setminus J_{f,0}$. Moreover, the number of flux nodes is equal to the number of faces $E \in \mathcal{E}_h$ which is small compared to the total number of velocity unknowns in (4.19), in particular, in the case of higher order finite elements. That means that in the algebraic system (4.19) generated by means of the QDF basis functions most of the velocity unknowns are decoupled from the pressure unknowns.

5. Implementation aspects.

5.1. Generation of the QDF basis functions. Within a numerical code it is not necessary to have explicit formulas for the modified QDF basis functions ψ_j. It suffices to know how one can obtain these functions by linear combinations of standard Lagrange basis functions.

First we will have a look at the element bubble correction operator P_b which is defined locally by (3.20). Using the standard bases $\{\varphi_j : j \in J(K)\}$ and $\{\chi_i : i \in \widetilde{I}(K)\}$ for the local velocity and pressure spaces $X_h(K)$ and $\widetilde{M}_h(K)$, respectively,

problem (3.20) for $w = \varphi_j$, $j \in J_s(K)$, is equivalent to the following system of linear equations:

$$\begin{pmatrix} A^K & (B^K)^T \\ B^K & 0 \end{pmatrix} \begin{pmatrix} \underline{c}_j^K \\ \underline{d}_j^K \end{pmatrix} = \begin{pmatrix} \underline{r}_j^K \\ \underline{s}_j^K \end{pmatrix} \tag{5.1}$$

with

$$\begin{aligned} (A^K)_{i,k} &:= a_h(\varphi_k, \varphi_i), \quad i,k \in J_b(K), \qquad B^K \text{ defined by (3.26)}, \\ (\underline{r}_j)_i &:= a_h(\varphi_j, \varphi_i), \quad i \in J_b(K), \qquad (\underline{s}_j)_i := b(\varphi_j, \chi_i), \quad i \in \widetilde{I}(K). \end{aligned}$$

We see that everything which we need to generate the above system is computed during the usual local assembling process of element stiffness matrices. From the solution vector $\underline{c}_j^K = (c_{j,i}^K) \in \mathbb{R}^{\mathrm{card}(J_b(K))}$ of the linear system (5.1) we get the following representation of the local part $P_b^K \varphi_j$ of the projection $P_b \varphi_j$ on element K:

$$P_b^K \varphi_j = \sum_{i \in J_b(K)} c_{j,i}^K \varphi_i, \qquad j \in J_s(K). \tag{5.2}$$

In order to be able to reconstruct the modified local functions $\widetilde{\varphi}_j\big|_K = \varphi_j\big|_K - P_b^K \varphi_j$, $j \in J_s(K)$, we store the coefficients $c_{j,i}^K$. The required storage for this is less than the memory in the original finite element method for storing the matrix entries $a_h(\varphi_j, \varphi_i)$ with $j \in J_s(K)$, $i \in J_b(K)$, which are no longer needed in the new method using the QDF basis.

Now we will describe how the action of the operator P_f defined in (3.40) can be implemented. For a basis function φ_j, which corresponds to the nodal point a_j, we have that $\varphi_j|_E \equiv \widetilde{\varphi}_j|_E \equiv 0$ for all faces $E \in \mathcal{E}_h$ with $a_j \notin \overline{E}$. Hence, the sum in (3.40) for $w = \widetilde{\varphi}_j$ reduces to a small sum over those faces E with

$$E \in \mathcal{E}_h(j) := \{E \in \mathcal{E}_h : a_j \in \overline{E}\}. \tag{5.3}$$

Then, the function $\psi_j = \widetilde{\varphi}_j - P_f \widetilde{\varphi}_j$, $j \in J_s(K) \setminus J_f(K)$, defined in (3.42), can be written as

$$\psi_j = \widetilde{\varphi}_j - \sum_{E \in \mathcal{E}_h(j)} \alpha_{j,E}\, \widetilde{\varphi}_{j_f(E)} \quad \text{with} \quad \alpha_{j,E} := \frac{\langle \varphi_j, n_E \rangle_E}{\langle \varphi_E, n_E \rangle_E}, \tag{5.4}$$

where we have used that φ_j and $\widetilde{\varphi}_j$ coincide on all $E \in \mathcal{E}_h$. For each face $E \in \mathcal{E}_h$, we store the coefficients $\alpha_{j,E}$ for all nodes j belonging to the closure of E, i.e., for all $j \in J(E)$ with

$$J(E) := \{j \in J : a_j \in \overline{E}\}, \tag{5.5}$$

where $\alpha_{j,E}$ is defined by (5.4) if $j \in J_s \setminus J_f$ and by

$$\alpha_{j,E} := \langle \varphi_j, n_E \rangle_E \qquad \text{if} \qquad j = j_f(E) \in J(E) \cap J_f\,.$$

This is useful since for $j = j_f(E) \in J_f$ we have $\psi_j = \widetilde{\varphi}_j$ and

$$\alpha_{j,E} = \langle \varphi_j, n^{K(E)} \rangle_{\partial K(E)} = \langle \psi_j, n^{K(E)} \rangle_{\partial K(E)} = \begin{cases} -b(\psi_j, \chi_0^K) & \text{if} \quad K = K(E) \\ +b(\psi_j, \chi_0^K) & \text{if} \quad K = K'(E), \end{cases}$$

where $K(E)$ and $K'(E)$ denote the two elements associated with the face E (see section 2) and $\chi_0^K \in \overline{M}_h$ is the function which is 1 on K and 0 on $\Omega \setminus K$. In the algebraic system, which is equivalent to the reduced Oseen problem (4.7), we need, for each velocity node $j \in J_s$, the matrix entries $b(\psi_j, \chi_0^K)$ for all $K \in \mathcal{T}_h$. Note that it is a common practice that the functions χ_0^K, which are not in $\overline{M}_{h,0}$, are used as a basis of the larger pressure space $\overline{M}_h$ and that the condition $\overline{p}_h \in \overline{M}_{h,0} = \overline{M}_h \cap L_0^2(\Omega)$ is realized within the discrete solver by subtracting a suitable global constant from the pressure approximation. Now, using that $\psi_j \in V_h$ for all $j \in \widetilde{J}_s = J_s \setminus J_f$, we get

$$b(\psi_j, \chi_0^K) = 0 \qquad \forall j \in J_s \setminus J_f\,,\ K \in \mathcal{T}_h\,,$$

i.e., the matrix entry $b(\psi_j, \chi_0^K)$ can be nonzero only if $j \in J_f(K)$. Therefore, if we store the coefficients $\alpha_{j,E} = \pm b(\psi_j, \chi_0^K)$ for $j = j_f(E) \in J_f(K)$, $E \in \mathcal{E}(K)$, then it is no longer necessary to store any matrix entries associated with the bilinear form $b(\cdot,\cdot)$. The memory required for storing the $\alpha_{j,E}$ is approximately equal to the memory that becomes free from the matrix entries of the bilinear form $b(\cdot,\cdot)$.

Now we consider the linear algebraic system which corresponds to the reduced Oseen problem (4.7) by using the new QDF basis $\{\psi_j : j \in J_s\}$ for $\widetilde{X}_s$ and the basis $\{\chi_0^K : K \in \mathcal{T}_h\}$ for $\overline{M}_h$. Using the stored coefficients $\alpha_{j,E}$ and $c_{j,i}^K$, this system is easily generated by means of linear combinations of rows and columns of the element stiffness matrices of the original finite element method. For example, the entries of the global stiffness matrix $\overline{A} = (\overline{A}_{i,j})$ with $\overline{A}_{i,j} := a_h(\psi_j, \psi_i)$ are assembled in an elementwise fashion as

$$\overline{A}_{i,j} = \sum_{K \in \mathcal{T}_h} \overline{A}_{i,j}^K \qquad \text{with} \qquad \overline{A}_{i,j}^K := a_K(\psi_j, \psi_i)\,,$$

where $a_K(\cdot,\cdot)$ denotes the additive part of $a_h(\cdot,\cdot)$ associated with the element K. The modified element matrix entries $\overline{A}_{i,j}^K$ are computed from the original entries $A_{i,j}^K := a_K(\varphi_j, \varphi_i)$ in the following way. Using the representation (5.4) of the QDF basis functions ψ_j restricted to an element K, we have for nonflux nodes $i, j \in J_s \setminus J_f$,

$$\begin{aligned}\overline{A}_{i,j}^K = a_K(\widetilde{\varphi}_j, \widetilde{\varphi}_i) &- \sum_{E \in \mathcal{E}_h^K(j)} \alpha_{j,E}\, a_K(\widetilde{\varphi}_{j_f(E)}, \widetilde{\varphi}_i) - \sum_{F \in \mathcal{E}_h^K(i)} \alpha_{i,F}\, a_K(\widetilde{\varphi}_j, \widetilde{\varphi}_{j_f(F)}) \\ &+ \sum_{E \in \mathcal{E}_h^K(j)} \sum_{F \in \mathcal{E}_h^K(i)} \alpha_{j,E}\, \alpha_{i,F}\, a_K(\widetilde{\varphi}_{j_f(E)}, \widetilde{\varphi}_{j_f(F)}),\end{aligned}$$

where $\mathcal{E}_h^K(j) := \{E \in \mathcal{E}(K) : a_j \in \overline{E}\}$ denotes the set of those faces of element K that are associated with the nodal point a_j.

In a similar way, one can compute in a previous step the needed intermediate matrix entries $\widetilde{A}_{i,j}^K := a_K(\widetilde{\varphi}_j, \widetilde{\varphi}_i)$ by linear combinations of the entries $A_{i,j}^K = a_K(\varphi_j, \varphi_i)$ of the original element stiffness matrix using the stored coefficients $c_{j,i}^K$.

Remark 5.1. The memory for storing the nonzero matrix entries $\overline{A}_{i,j}$ is the same as for storing the nonzero entries $A_{i,j}$, $i, j \in J_s$, of the original global stiffness matrix which are no longer needed for solving the reduced Oseen problem (4.7). Moreover, the additional work to compute the element matrix entries $\overline{A}_{i,j}^K$ from the original entries $A_{i,j}^K$ is only proportional to the number of velocity unknowns due to the local support of the QDF basis functions.

5.2. Computation of $\widetilde{p}_h$. In the following, we want to discuss how the pressure part $\widetilde{p}_h \in \widetilde{M}_h$ of the pressure solution $p_h = \overline{p}_h + \widetilde{p}_h$ given in Theorem 4.1 can be determined in practice if the parts $\widetilde{u}_s$ and $u_b = \sum_{K\in\mathcal{T}_h} u_b^K$ of the velocity solution $u_h = \widetilde{u}_s + u_b$ have been already computed. From Theorem 4.1 we know that $\widetilde{p}_h$ can be decomposed elementwise as $\widetilde{p}_h = \sum_{K\in\mathcal{T}_h} \widetilde{p}_h^K$, where $\widetilde{p}_h^K \in \widetilde{M}_h(K)$ is the unique solution of the local problem (4.11). There are two ways to solve this problem. The first one is to solve for the coefficient vector $\underline{\widetilde{p}}^K$ of $\widetilde{p}_h^K$ the linear algebraic system that corresponds to (4.11). The structure of this system is

$$\left(B^K\right)^T \underline{\widetilde{p}}^K = \underline{z}^K, \tag{5.6}$$

where

$$\begin{aligned} &\left(B^K\right)^T \in \mathbb{R}^{\dim(X_b(K))\times\dim(\widetilde{M}_h(K))}, \quad \underline{\widetilde{p}}^K \in \mathbb{R}^{\dim(\widetilde{M}_h(K))}, \\ &\underline{z}^K = (z_j^K) \in \mathbb{R}^{\dim(X_b(K))} \quad \text{with} \qquad z_j^K := f_h(\varphi_j) - a_K(u_h,\varphi_j) \qquad \forall j \in J_b(K). \end{aligned}$$

For the definition of the matrix B^K, see (3.26). Since (4.11) has a unique solution (see Theorem 4.1) and (5.6) is equivalent to (4.11) it follows that $\ker((B^K)^T)=\{0\}$. Therefore, the symmetric matrix $B^K(B^K)^T$ is regular and $\underline{\widetilde{p}}^K$ can be computed by solving the linear system

$$B^K(B^K)^T\underline{\widetilde{p}}^K = B^K\underline{z}^K. \tag{5.7}$$

The disadvantage of this approach is that the local element matrices, which are needed to generate the systems (5.7) for all elements $K \in \mathcal{T}_h$, have to be assembled again since they are not stored inside of our new multigrid method for the reduced Oseen problem (4.7).

The second approach for computing $\widetilde{p}_h^K \in \widetilde{M}_h(K)$ uses the local pressure solutions $\theta^K \in \widetilde{M}_h(K)$ of the local Oseen problem (4.6) and $\theta_j^K := Q_b^K\varphi_j \in \widetilde{M}_h(K)$, $j \in J_s(K)$, of problem (3.20) with $w = \varphi_j$. The algebraic systems equivalent to these local Oseen problems have to be solved anyway by means of a direct solver in order to generate the QDF basis functions for the velocity. Thus, the coefficient vectors of θ^K and θ_j^K are available for free. For our second approach, we assume that these coefficient vectors will be stored. The required storage for this is less than the memory that becomes free from the matrix entries $b(\varphi_j,\chi_i)$, $j \in J_s(K)$, $i \in \widetilde{I}(K)$, and the right-hand side components $-b(\widetilde{g}_h,\chi_i)$, $i \in \widetilde{I}(K)$, which are no longer needed for our new method relying on QDF basis functions. Now, let us assume that the local representation of the computed solution $\widetilde{u}_s \in \widetilde{X}_s$ of the reduced Oseen problem (4.7) on an element $K \in \mathcal{T}_h$ is given by means of coefficients $\widetilde{u}_j \in \mathbb{R}$ as follows:

$$\widetilde{u}_s(x) = \sum_{j\in J_s(K)} \widetilde{u}_j\psi_j(x) \qquad \forall\, x \in K.$$

Applying the definition (3.42) of the ψ_j and (5.4), we can represent $\widetilde{u}_s$ in the form

$$\widetilde{u}_s(x) = \sum_{j\in J_s(K)} u_j\widetilde{\varphi}_j(x) \qquad \forall\, x \in K$$

using the basis functions $\widetilde{\varphi}_j \in \widetilde{X}_s$ defined by (3.21) and coefficients $u_j \in \mathbb{R}$ defined as $u_j := \widetilde{u}_j$ if $j \in J_s(K)\setminus J_f(K)$ and $u_j := \widetilde{u}_j - \sum_{i\in J(E)\setminus\{j\}} \alpha_{i,E}\widetilde{u}_i$ if $j = j_f(E) \in J_f(K)$.

From the definition of $\widetilde{\varphi}_j\big|_K = \varphi_j\big|_K - P_b^K\varphi_j$ and $\theta_j^K := Q_b^K\varphi_j$ in (3.20), it follows that

$$b(v_b, \theta_j^K) = a_h(\widetilde{\varphi}_j, v_b) \qquad \forall v_b \in X_b(K).$$

Using the definition of θ^K in (4.6) we obtain that the pressure function $\widetilde{p}_h^K$ defined by

$$\widetilde{p}_h^K := \theta^K - \sum_{j\in J_s(K)} u_j\theta_j^K \tag{5.8}$$

is the unique solution of (4.11). This shows that we can calculate $\widetilde{p}_h^K$ by the cheap computation of the coefficients u_j from the coefficients $\widetilde{u}_j$ of the QDF basis representation of $\widetilde{u}_s$ if we invest the memory to store the coefficients that represent the pressure functions θ^K and θ_j^K.

5.3. Transformation of basis representations. In this subsection, we will describe two operators which allow one to switch from the basis representation of a function in $\widetilde{X}_h = \operatorname{span}\{\psi_j : j \in \widetilde{J}\}$ to the one in $X_h = \operatorname{span}\{\varphi_j : j \in J\}$ and vice versa. Let $N := \operatorname{card}(J)$ and $\widetilde{N} := \operatorname{card}(\widetilde{J})$. By the finite element isomorphism a function $v_h \in \widetilde{X}_h \subset X_h$ is uniquely associated with a coefficient vector $\underline{\widetilde{v}} = (\widetilde{v}_j) \in \mathbb{R}^{\widetilde{\mathcal{N}}}$ such that $v_h = \sum_{j\in\widetilde{\mathcal{P}}} \widetilde{v}_j\psi_j$. Thus, using (3.42) and (5.4), we have

$$v_h = \sum_{j\in\widetilde{\mathcal{P}}_s} \widetilde{v}_j\big(\widetilde{\varphi}_j - P_f\varphi_j\big) + \sum_{j\in\widetilde{\mathcal{P}}_b\cup J_f} \widetilde{v}_j\widetilde{\varphi}_j = \sum_{j\in\mathcal{P}} v_j^*\widetilde{\varphi}_j$$

with coefficients v_j^* defined by

$$v_j^* := \begin{cases} \widetilde{v}_j & \text{if} \quad j \in \widetilde{J}_s \cup \widetilde{J}_b, \\ \widetilde{v}_j - \displaystyle\sum_{i\in J(E)\setminus\{j\}} \alpha_{i,E}\widetilde{v}_i & \text{if} \quad j = j_f(E) \in J_f. \end{cases} \tag{5.9}$$

Now, from $\widetilde{J} = J_s \cup \widetilde{J}_b$ and the representation (3.27) of $\widetilde{\varphi}_j$ for $j \in \widetilde{J}_b$ we get

$$v_h = \sum_{j\in\mathcal{P}} v_j^*\widetilde{\varphi}_j = \sum_{j\in J_s} v_j^*\big(\varphi_j - P_b\varphi_j\big) + \sum_{K\in\mathcal{T}_h} \sum_{j\in\tilde{J}_b(K)} v_j^* \sum_{i\in J_b(K)} c_{j,i}^K\varphi_i.$$

Using the local representation (5.2) of $P_b\varphi_j$ for $j \in J_s$, we obtain the following basis representation of $v_h = \sum_{j\in\widetilde{\mathcal{P}}} \widetilde{v}_j\psi_j \in \widetilde{X}_h$ by means of the original basis $\{\varphi_j : j \in J\}$ of X_h:

$$v_h = \sum_{j\in J} v_j\varphi_j \qquad \text{with} \qquad v_j := \begin{cases} v_j^* & \text{if} \quad j \in J_s, \\ \displaystyle\sum_{i\in\mathcal{P}_b(K)} c_{i,j}^K v_i^* - \sum_{i\in J_s(K)} c_{i,j}^K v_i^* & \text{if} \quad j \in J_b(K). \end{cases} \tag{5.10}$$

Obviously, the mapping that assigns the coefficient vector $\underline{v}$ of the original basis representation to the coefficient vector $\underline{\widetilde{v}}$ of the QDF basis representation of a function $v_h \in \widetilde{X}_h$ is linear, i.e., there is a linear operator $\Pi : \mathbb{R}^{\mathcal{N}} \to \mathbb{R}^N$ such that

$$v_h = \sum_{j\in\widetilde{\mathcal{P}}} (\underline{\widetilde{v}})_j\psi_j = \sum_{j\in J} (\Pi\underline{\widetilde{v}})_j\varphi_j \ \in \ \widetilde{X}_h \qquad \forall\, \underline{\widetilde{v}} \in \mathbb{R}^{\widetilde{\mathcal{N}}}. \tag{5.11}$$

At the end of this section we want to construct for a given function $v_h = \sum_{j\in J} v_j\varphi_j \in X_h$ a suitable QDF approximation $\widetilde{v}_h = \sum_{j\in\mathcal{F}} \widetilde{v}_j\psi_j \in \widetilde{X}_h \subset X_h$ which will be used later on during the prolongation in the multigrid method. Using the decomposition $X_h = X_s \oplus X_b$ we can write any function $v_h \in X_h$ uniquely in the form

$$v_h = v_s + v_b, \qquad \text{with} \qquad v_s = \sum_{j\in J_s} v_j\varphi_j \in X_s, \quad v_b = \sum_{j\in J_b} v_j\varphi_j \in X_b.$$

Since the projector $I - P_b$ maps from X_s into the space $\widetilde{X}_s \subset \widetilde{X}_h$, a natural approximation of the part v_s of v_h is the function $\widetilde{v}_s$ with

$$\widetilde{v}_s := (I - P_b)v_s = \sum_{j\in J_s} v_j\widetilde{\varphi}_j \in \widetilde{X}_s.$$

Therefore, we have

$$v_h = \widetilde{v}_s + w_b \qquad \text{with} \qquad w_b := v_b + P_b v_s \ \in \ X_b,$$

which is, due to the decomposition $X_h = \widetilde{X}_s \oplus X_b$ (see (3.33)), the unique splitting of v_h into a function $\widetilde{v}_s \in \widetilde{X}_s$ and a function $w_b \in X_b$. Thus, the decomposition $\widetilde{X}_h = \widetilde{X}_s \oplus V_b$ (see (3.32)) suggests to define the approximation $\widetilde{v}_h \in \widetilde{X}_h$ of v_h as

$$\widetilde{v}_h := \widetilde{v}_s + \widetilde{w}_b \qquad \text{with} \qquad \widetilde{w}_b := L_b w_b \in V_b\,, \tag{5.12}$$

where $L_b : X_b \to V_b$ is a suitable projection operator. Using the elementwise decomposition of X_b (see (3.18)), we have

$$w_b = \sum_{K\in\mathcal{T}_h} w_b^K \qquad \text{with} \qquad w_b^K = \sum_{j\in J_b(K)} v_j\varphi_j + \sum_{i\in J_s(K)} v_i P_b^K\varphi_i = \sum_{j\in J_b(K)} w_j^K\varphi_j,$$

where

$$w_j^K := v_j \ + \sum_{i\in J_s(K)} v_i c_{i,j}^K \qquad \forall\, j \in J_b(K)\,. \tag{5.13}$$

We define L_b locally as

$$L_b w_b = \sum_{K\in\mathcal{T}_h} L_b^K w_b^K,$$

where $L_b^K : X_b(K) \to V_b(K)$ is a suitable projection operator as, for example, the L^2 projection. The local coefficient vector $\underline{\widetilde{w}}^K := (\widetilde{w}_j^K)$ of the projection $\widetilde{w}_b^K = L_b^K w_b^K$ can be obtained from the coefficient vector $\underline{w}^K := (w_j^K)_{j\in J_b(K)}$ by matrix-vector multiplication with a small local matrix, say $D^K = (D_{i,j}^K) \in \mathbb{R}^{\mathrm{card}(\mathcal{F}_b(K))\times\mathrm{card}(J_b(K))}$, i.e.,

$$\widetilde{w}_b^K = \sum_{j\in\mathcal{F}_b(K)} \widetilde{w}_j^K\psi_j \qquad \text{with} \qquad \widetilde{w}_j^K = \sum_{i\in J_b(K)} D_{j,i}^K w_i^K\,.$$

In an efficient implementation, the matrices D^K should be computed and stored for each element $K \in \mathcal{T}_h$ during the assembling process of the element stiffness matrices

before starting the multigrid solver. The required memory is less than the one for storing the matrix entries of the bubble functions $a_h(\varphi_j, \varphi_i)$, $i, j \in J_b(K)$, in the original finite element method which is no longer needed in our new method.

Finally, a straightforward calculation yields the following basis representation for the approximation $\widetilde{v}_h \in \widetilde{X}_h$ of a given function $v_h = \sum_{j \in J} v_j \varphi_j \in X_h$:

(5.14)
$$\widetilde{v}_h = \sum_{j \in \mathcal{F}} \widetilde{v}_j \psi_j \qquad \text{with} \qquad \widetilde{v}_j := \begin{cases} v_j & \text{if} \quad j \in J_s \setminus J_f, \\ \displaystyle\sum_{i \in J_b(K)} D^K_{j,i} w^K_i & \text{if} \quad j \in \widetilde{J}_b(K), \\ \displaystyle v_j + \sum_{i \in J(E) \setminus \{j\}} v_i \, \alpha_{i,E} & \text{if} \quad j = j_f(E) \in J_f, \end{cases}$$

where the w^K_i are defined by (5.13).

Obviously, the resulting coefficient vector $\underline{\widetilde{v}} = (\widetilde{v}_j) \in \mathbb{R}^{\mathcal{F}}$ of $\widetilde{v}_h \in \widetilde{X}_h$ in the QDF basis depends linearly on the coefficient vector $\underline{v} = (v_j) \in \mathbb{R}^N$ of $v_h \in X_h$. Hence, there is a linear operator $\widetilde{\Pi} : \mathbb{R}^N \to \mathbb{R}^{\mathcal{F}}$ such that

$$\widetilde{v}_h = \sum_{j \in \mathcal{F}} (\widetilde{\Pi} \underline{v})_j \psi_j \ \in \ \widetilde{X}_h \qquad \forall \, \underline{v} \in \mathbb{R}^N, \tag{5.15}$$

is the projection of v_h into the subspace $\widetilde{X}_h$. It is easy to verify that a given $v_h \in \widetilde{X}_h$ implies $\widetilde{v}_h = v_h$.

As a summary we can say that both operators Π and $\widetilde{\Pi}$ as well as all other tools that are necessary to work with the QDF basis can be implemented efficiently with a computational work which is only proportional to the number of unknowns. This fact is important to ensure optimal computational costs for our proposed multigrid solver. The required memory to store the additionally needed coefficients $c^K_{j,i}$, $\alpha_{j,E}$, and $D^K_{i,j}$ is less than the memory that will become free from the fact that a lot of nonzero matrix entries of the original finite element method are no longer used in the new multigrid solver based on the QDF basis functions.

6. Multigrid method.

6.1. The multigrid cycle. In the following, we will describe the modifications of the usual multigrid cycle that are made in our approach. First we introduce some notations that use the grid level ℓ instead of the discretization parameter h. Let the final grid $\mathcal{T}_h$ be the finest grid $\mathcal{T}^{(L)} := \mathcal{T}_{h_L}$ of a sequence of uniformly refined grids $\mathcal{T}^{(\ell)} := \mathcal{T}_{h_\ell}$, $\ell = 0, 1, \ldots, L$, and let $X^{(\ell)}$, $\widetilde{X}^{(\ell)}$, $\widetilde{X}^{(\ell)}_s$, $X^{(\ell)}_b$, $V^{(\ell)}_b$, $\widetilde{M}^{(\ell)}$, and $\overline{M}^{(\ell)}$ denote the corresponding spaces $X_{h,0}$, $\widetilde{X}_{h,0}$, $\widetilde{X}_{s,0}$, X_b, V_b, $\widetilde{M}_h$, and $\overline{M}_{h,0}$ associated with the grid $\mathcal{T}_h = \mathcal{T}^{(\ell)}$.

The aim of one multigrid cycle on the finest level $\ell = L$ is to compute an improved approximation of the solution of the [illegible] ℓ:

[illegible] $(\widetilde{u}^{(\ell)}_s, \overline{p}^{(\ell)}) \in \widetilde{X}^{(\ell)}_s \times \overline{M}^{(\ell)}$ [illegible]

$$\begin{aligned} a_h(\widetilde{u}^{(\ell)}_s, \widetilde{v}_s) + b(\widetilde{v}_s, \overline{p}^{(\ell)}) &= \langle \, d^{(\ell)}, \widetilde{v}_s \, \rangle \qquad \forall \, \widetilde{v}_s \in \widetilde{X}^{(\ell)}_s \,, \\ b(\widetilde{u}^{(\ell)}_s, \overline{q}) &= \langle \, e^{(\ell)}, \overline{q} \, \rangle \qquad \forall \, \overline{q} \in \overline{M}^{(\ell)} \,. \end{aligned} \tag{6.1}$$

Here, $\widetilde{u}_s^{(L)}$ denotes the solution part $\widetilde{u}_{s,0}$ in the decomposition $u_h = (\widetilde{g}_h + \widetilde{u}_{s,0}) + u_b$, see (4.3) and (4.7), and the linear functionals are defined by

$$\begin{aligned} \langle\, d^{(L)}, \widetilde{v}_s \,\rangle &:= f_h(\widetilde{v}_s) - a_h(\widetilde{g}_h, \widetilde{v}_s) - a_h(u_b, \widetilde{v}_s)\,, \\ \langle\, e^{(L)}, \overline{q} \,\rangle &:= -b(\widetilde{g}_h, \overline{q})\,. \end{aligned} \tag{6.2}$$

The smoothing procedure applied to (6.1) will be described in section 6.2.

For the coarse grid correction, we consider two different approaches. The first one is a so-called [illegible] presented in [13, sect. 4.3]. Here, the coarse space correction is computed in a lowest order space based on the finest grid $\mathcal{T}^{(L)}$. This problem is then solved by a usual multigrid approach which uses the lowest order spaces on all grids $\mathcal{T}^{(\ell)}$, $\ell \le L$.

In the second approach, the coarse space corrections are computed in the spaces $\widetilde{X}^{(\ell)} \times \overline{M}^{(\ell)}$ with $\ell < L$. Here, our first modification compared to a usual multigrid cycle is that the coarse grid problems on levels $\ell < L$ are not reduced Oseen problems of type (6.1) but QDF-formulations of discrete Oseen problems of the type:

[illegible] $(\widetilde{u}^{(\ell)}, \overline{p}^{(\ell)}) \in \widetilde{X}^{(\ell)} \times \overline{M}^{(\ell)}$ [illegible]

$$\begin{aligned} a_h(\widetilde{u}^{(\ell)}, \widetilde{v}) \,+\, b(\widetilde{v}, \overline{p}^{(\ell)}) &= \langle\, \widetilde{d}^{(\ell)}, \widetilde{v} \,\rangle && \forall\, \widetilde{v} \in \widetilde{X}^{(\ell)}\,, \\ b(\widetilde{u}^{(\ell)}, \overline{q}) &= \langle\, e^{(\ell)}, \overline{q} \,\rangle && \forall\, \overline{q} \in \overline{M}^{(\ell)}\,, \end{aligned} \tag{6.3}$$

where the linear functionals $\langle\, \widetilde{d}^{(\ell)}, \cdot \,\rangle$ and $\langle\, e^{(\ell)}, \cdot \,\rangle$ are defined via defect restriction from the next finer grid level. The reason that we have to solve in the larger space $\widetilde{X}^{(\ell)} = \widetilde{X}_s^{(\ell)} \oplus V_b^{(\ell)}$ is that on one hand the restricted coarse grid defects $\langle\, \widetilde{d}^{(\ell)}, \widetilde{v} \,\rangle$ do not vanish for coarse quasi divergence free bubble functions $\widetilde{v} \in V_b^{(\ell)}$ and on the other hand that the subspaces $\widetilde{X}_s^{(\ell)}$, for $\ell = 0, 1, \ldots, L$, do not satisfy the approximation property in contrast to the spaces $\widetilde{X}^{(\ell)}$.

Problem (6.3) on a coarse grid level $\ell < L$ can be decomposed into two subproblems. Let $u_b^{(\ell)} \in V_b^{(\ell)}$ be the solution of the mixed problem:

[illegible] $(u_b^{(\ell)}, \theta^{(\ell)}) \in X_b^{(\ell)} \times \widetilde{M}^{(\ell)}$ [illegible]

$$\begin{aligned} a_h(u_b^{(\ell)}, v_b) \,+\, b(v_b, \theta^{(\ell)}) &= \langle\, \widetilde{d}^{(\ell)}, v_b \,\rangle && \forall\, v_b \in X_b^{(\ell)}\,, \\ b(u_b^{(\ell)}, \widetilde{q}) &= 0 && \forall\, \widetilde{q} \in \widetilde{M}^{(\ell)}\,. \end{aligned} \tag{6.4}$$

Due to the structure of the functions in $X_b^{(\ell)}$, this problem decomposes into local problems over each element of the grid. Instead of solving (6.4) on each newly entered coarser level $\ell < L$, one can compute and store during the assembling process of the stiffness matrix on level ℓ, for each element $K \in \mathcal{T}^{(\ell)}$, a local matrix that assigns to a given local vector of bubble defects $\langle\, \widetilde{d}^{(\ell)}, \varphi_i \,\rangle$ with $i \in J_b(K)$ the corresponding local vector of nodal values of the solution $u_b^{(\ell)}$ restricted on K.

Now, one can easily show that the solution $(\widetilde{u}^{(\ell)}, \overline{p}^{(\ell)})$ of (6.3) can be computed as

$$\widetilde{u}^{(\ell)} = \widetilde{u}_s^{(\ell)} + u_b^{(\ell)},$$

where $(\widetilde{u}_s^{(\ell)}, \overline{p}^{(\ell)}) \in \widetilde{X}_s^{(\ell)} \times \overline{M}^{(\ell)}$ is the solution of a reduced Oseen problem of type (6.1) with the linear functional

$$\langle\, d^{(\ell)}, \widetilde{v}_s \,\rangle := \langle\, \widetilde{d}^{(\ell)}, \widetilde{v}_s \,\rangle - a_h(u_b^{(\ell)}, \widetilde{v}_s) \qquad \forall\, \widetilde{v}_s \in \widetilde{X}_s^{(\ell)}\,. \tag{6.5}$$

6.1. In the case of a polynomial degree of $r = 2$, we have $V_b^{(\ell)} = \{0\}$, i.e., $u_b^{(\ell)} = 0$ and $\widetilde{X}^{(\ell)} = \widetilde{X}_s^{(\ell)}$ such that (6.3) is already a reduced Oseen problem of type (6.1).

Due to the decomposition of each coarse grid problem (6.3), we have to apply the smoothing procedure and the defect restriction only to each reduced Oseen problem of type (6.1). A further modification compared to a usual multigrid cycle concerns the choice of the fine grid correction. Let $(\widetilde{u}_s^{(\ell+1)}, \overline{p}^{(\ell+1)}) \in \widetilde{X}_s^{(\ell+1)} \times \overline{M}^{(\ell+1)}$ be a multigrid iterate after the smoothing step applied to the reduced Oseen problem (6.1) on level $\ell+1$. For the corresponding coarse grid problem (6.3), let $u_b^{(\ell)}$ denote the velocity solution of (6.4) and $(\widetilde{u}_s^{(\ell)}, \overline{p}^{(\ell)}) \in \widetilde{X}_s^{(\ell)} \times \overline{M}^{(\ell)}$ the corresponding coarse grid correction computed by applying some multigrid cycles to the reduced Oseen problem (6.1) on level ℓ. As fine grid correction for improving the iterate $\widetilde{u}_s^{(\ell+1)}$ we need a function $\widetilde{v}_s^{(\ell+1)} \in \widetilde{X}_s^{(\ell+1)}$. However, the corresponding coarse grid correction from (6.3), which is the function $\widetilde{u}^{(\ell)} = u_b^{(\ell)} + \widetilde{u}_s^{(\ell)}$, is contained in the space $\widetilde{X}^{(\ell)}$. That means, we need a prolongation operator $\widetilde{P}^{(\ell)} : \widetilde{X}^{(\ell)} \to \widetilde{X}^{(\ell+1)}$ such that the fine grid correction $\widetilde{v}_s^{(\ell+1)} = \widetilde{P}^{(\ell)} \widetilde{u}^{(\ell)}$ is in the subspace $\widetilde{X}_s^{(\ell+1)}$ with zero bubble part. This is realized if we choose simply the operator $L_b^K = 0$ in the definition of the operator $\widetilde{\Pi}$; see section 5.3. As the fine grid correction for the pressure approximation $\overline{p}^{(\ell+1)}$ we take the usual prolongation of $\overline{p}^{(\ell)}$.

6.2. Smoother. We apply the smoother on each grid level ℓ only to the corresponding reduced Oseen problem of type (6.1). One smoothing step consists essentially of a loop over all elements where some block solves are performed for each element $K \in \mathcal{T}^{(\ell)}$. In order to simplify the notation we omit the level ℓ and introduce, for a nodal point a_m at the boundary of an element $K \in \mathcal{T}_h = \mathcal{T}^{(\ell)}$, the set $J_n(a_m)$ of all nonflux nodes which are connected to a_m, i.e.,

$$J_n(a_m) := \{ j \in J \setminus J_f : a_j = a_m \}.$$

We use a block Gauss–Seidel smoother which can be regarded as a successive subspace correction method. Therefore, we define, for an arbitrary index set $I \subset \widetilde{J}$, the velocity subspace

$$\widetilde{X}_I := \operatorname{span}\{\psi_j : j \in I\} \cap \widetilde{X}_{h,0}$$

and the local pressure subspaces

$$\overline{M}_K := \operatorname{span}\{\chi_0^K\} \subset \overline{M}_h \qquad \forall K \in \mathcal{T}_h.$$

Let $(u^{\mathrm{old}}, p^{\mathrm{old}}) \in \widetilde{X}_s^{(\ell)} \times \overline{M}^{(\ell)}$ be an approximation of the solution $(\widetilde{u}_s^{(\ell)}, \overline{p}^{(\ell)})$ of the reduced Oseen problem of type (6.1) on some grid level ℓ with right-hand side functionals $\langle d, \cdot \rangle$ and $\langle e, \cdot \rangle$ defined by defect restriction and formula (6.5). Then, one smoothing step $(u^{\mathrm{old}}, p^{\mathrm{old}}) \mapsto (u, p)$ is defined as follows:

$(u, p) := (u^{\mathrm{old}}, p^{\mathrm{old}})$
for $K \in \mathcal{T}_h$ **do**
(i) Find $(v, q) \in \widetilde{X}_{J_f(K)} \times \overline{M}_K$ such that

(6.6)
$$\begin{aligned} a_h(v, w) + b(w, q) &= \langle d, w \rangle - a_h(u, w) - b(w, p) && \forall w \in \widetilde{X}_{J_f(K)}, \\ b(v, \mu) &= \langle e, \mu \rangle - b(u, \mu) && \forall \mu \in \overline{M}_K, \end{aligned}$$

update $(u, p) := (u, p) + (v, q)$

(ii) **for** all nodal points $a_m \in \partial K \setminus \Gamma$ **do**

Find $v \in \widetilde{X}_{J_n(a_m)}$ such that

$$a_h(v,w) = \langle\, d, w \,\rangle - a_h(u,w) \qquad \forall w \in \widetilde{X}_{J_n(a_m)} \tag{6.7}$$

update $u := u + v$

end do

end do

L_0^2 projection $p := p - |\Omega|^{-1}(p,1)_\Omega$.

In (6.6) we solve a system associated with all flux nodes of element K and the one node for the pressure constant. This is a small saddle-point problem of the dimension $\mathrm{card}\big(\mathcal{E}(K)\big) + 1$. For all nodal points $a_m \in \partial K$, we have $\widetilde{X}_{J_n(a_m)} \subset V_h$ due to $\widetilde{X}_s = V_s \oplus \widetilde{X}_f$, i.e., in the problem (6.1), the functions of the subspace $\widetilde{X}_{J_n(a_m)}$ are decoupled from the pressure. Therefore, the system (6.7) does not contain pressure terms. The size of the linear system associated with (6.7) is equal to $\mathrm{card}\,(J_n(a_m))$. It is either $(d-1)$, if a_m is connected to a flux node of K in the sense that a face $E \in \mathcal{E}(K)$ exists with $a_m = a_{j_f(E)}$, or it is d otherwise. For the special case of the Q_2-element in 2D, the different blocks of degrees of freedom associated with the systems (6.6) and (6.7) are shown in Figure 6.1. Since all local systems are of small size they can be solved very efficiently by means of a direct solver.

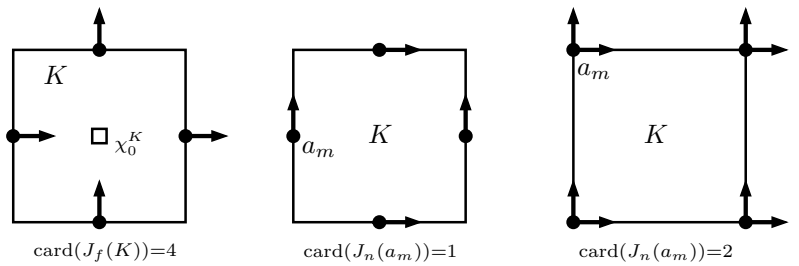


FIG. 6.1. *Blocks of degrees of freedom for the Q_2-element in 2D. Left: one 5×5-block from* (6.6)*, middle: four 1×1-blocks from* (6.7)*, right: four 2×2-blocks from* (6.7).

6.3. Prolongation and restriction. In the following, we describe the transfer operators between the fine and coarse grid velocity spaces. For the pressure spaces, the transfer operators are chosen in the usual way. Let the operators Π and $\widetilde{\Pi}$ of section 5.3 in the situation $X_h = X^{(\ell)}$ and $\widetilde{X}_h = \widetilde{X}^{(\ell)}$, $\ell = 0, \ldots, L$, be denoted by $\Pi^{(\ell)} : \mathbb{R}^{\widetilde{N}^{(\ell)}} \to \mathbb{R}^{N^{(\ell)}}$ and $\widetilde{\Pi}^{(\ell)} : \mathbb{R}^{N^{(\ell)}} \to \mathbb{R}^{\widetilde{N}^{(\ell)}}$, where $\widetilde{N}^{(\ell)} := \dim(\widetilde{X}^{(\ell)})$ and $N^{(\ell)} := \dim(X^{(\ell)})$. Moreover, let us assume that the local projection operator $L_b^K : X_b(K) \to V_b(K)$, used in the definition of the operator $\widetilde{\Pi}$, is defined by $L_b^K w := 0$ for all $w \in X_b(K)$. This is necessary since no prolongated correction may be added to the already computed fine grid bubble parts.

Now, we will define the prolongation operator $\widetilde{P}^{(\ell)} : \widetilde{X}^{(\ell)} \to \widetilde{X}^{(\ell+1)}$, i.e., we will describe how a fine grid QDF approximation $\widetilde{u}^{(\ell+1)} \in \widetilde{X}^{(\ell+1)}$ of a given coarse grid QDF function $\widetilde{u}^{(\ell)} \in \widetilde{X}^{(\ell)}$ is computed; schematically, this is shown in Figure 6.2. The function $\widetilde{u}^{(\ell)} \in \widetilde{X}^{(\ell)}$ is uniquely associated with the coefficient vector $\underline{\widetilde{u}}^{(\ell)} \in \mathbb{R}^{\widetilde{N}^{(\ell)}}$ in the QDF basis. From this vector we compute the coefficient vector $\underline{u}^{(\ell)} := \Pi^{(\ell)} \underline{\widetilde{u}}^{(\ell)}$

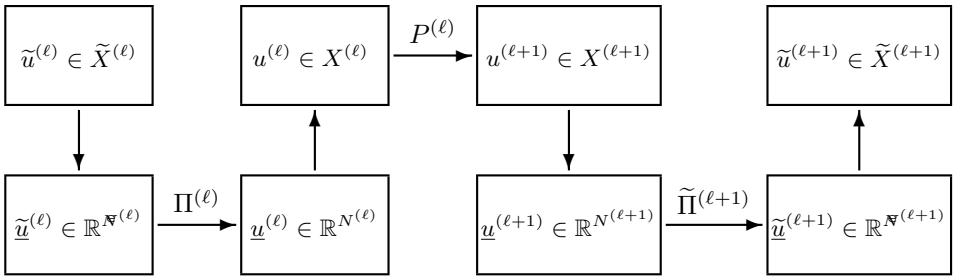


FIG. 6.2. *Computation of the prolongation $\widetilde{u}^{(\ell+1)}$ of a given coarse grid QDF function $\widetilde{u}^{(\ell)}$.*

that corresponds to the representation of $u^{(\ell)} = \widetilde{u}^{(\ell)}$ in the original finite element basis of $X^{(\ell)}$; see (5.11). Now, we apply the prolongation operator $P^{(\ell)} : X^{(\ell)} \to X^{(\ell+1)}$ that would be used for a multigrid method with the original spaces $X^{(\ell)}$, $\ell = 0, \dots, L$. This operator yields a fine grid approximation $u^{(\ell+1)} := P^{(\ell)} u^{(\ell)}$ of $\widetilde{u}^{(\ell)}$ in the original fine grid space $X^{(\ell+1)}$ which again corresponds to a coefficient vector $\underline{u}^{(\ell+1)} \in \mathbb{R}^{N^{(\ell+1)}}$. Finally, we apply our operator $\widetilde{\Pi}^{(\ell+1)}$ in order to compute the coefficient vector $\underline{\widetilde{u}}^{(\ell+1)} := \widetilde{\Pi}^{(\ell+1)} \underline{u}^{(\ell+1)}$ of the projection $\widetilde{u}^{(\ell+1)}$ of $u^{(\ell+1)}$ into the subspace $\widetilde{X}^{(\ell+1)}$ of the QDF functions in $X^{(\ell+1)}$. Thus, the definition of the prolongation $\widetilde{u}^{(\ell+1)} = \widetilde{P}^{(\ell)} \widetilde{u}^{(\ell)}$ of the coarse grid QDF function $\widetilde{u}^{(\ell)}$ is completed.

For the defect restriction operator $\widetilde{R}^{(\ell)} : (\widetilde{X}^{(\ell+1)})' \to (\widetilde{X}^{(\ell)})'$, we choose the adjoint of the above defined prolongation operator $\widetilde{P}^{(\ell)} : \widetilde{X}^{(\ell)} \to \widetilde{X}^{(\ell+1)}$. We will omit the technical details here.

6.4. Solving the Navier–Stokes problem. For solving the stationary Navier–Stokes equations, we have to find a velocity field u and pressure field p which solve the Oseen equations (1.1) with the convection field $\beta = u$. A finite element discretization on the finest mesh $\mathcal{T} = \mathcal{T}^{(L)}$ leads to a nonlinear system of equations for nodal vectors U and P of the following form:

$$\begin{pmatrix} A(U) & B_1(U)^T \\ B & 0 \end{pmatrix} \begin{pmatrix} U \\ P \end{pmatrix} = \begin{pmatrix} F(U) \\ G \end{pmatrix}. \tag{6.8}$$

Here, the matrix $A(U)$ is associated with the bilinear form a_h, matrix B with the bilinear form b, matrix $B_1(U)$ with a bilinear form b_1, and the vector $F(U)$ with the linear form f_h, where a_h, b_1, and f_h use a convection field β which is the finite element function related to the nodal vector U. For the Galerkin discretization, we have $a_h = a$, $b_1 = b$, and $f_h(\cdot) = (f, \cdot)_\Omega$. As a stabilized discretization in case of a high Reynolds number, we choose the reduced streamline diffusion method given in [6], i.e., we choose $a_h = a_{\mathrm{SD}}$, $b_1 = b_{\mathrm{SD}}$, and $f_h = f_{\mathrm{SD}}$ with

$$\begin{aligned} a_{\mathrm{SD}}(w, v) &:= a(w, v) + \textstyle\sum_{K \in \mathcal{T}} \delta_K \left(-2\nu \operatorname{div} \mathbb{D}(w) + (\beta \cdot \nabla) w, (\beta \cdot \nabla) v \right)_K, \\ b_{\mathrm{SD}}(v, q) &:= b(v, q) + \textstyle\sum_{K \in \mathcal{T}} \delta_K \left(\nabla q, (\beta \cdot \nabla) v \right)_K, \\ f_{\mathrm{SD}}(v) &:= (f, v)_\Omega + \textstyle\sum_{K \in \mathcal{T}} \delta_K \left(f, (\beta \cdot \nabla) v \right)_K. \end{aligned} \tag{6.9}$$

The parameter δ_K is chosen as $\delta_K = \delta_0 h_K^2$ with a constant δ_0; see [6].

For solving (6.8), we use a Picard-type linearization combined with a defect correction, i.e., a nonlinear iteration $(U^n, P^n) \to (U^{n+1}, P^{n+1})$ is defined as follows:

(i) solve approximately

$$\begin{pmatrix} A(U^n) & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} V^n \\ Q^n \end{pmatrix} = \begin{pmatrix} D^n \\ E^n \end{pmatrix}, \tag{6.10}$$

where

$$\begin{pmatrix} D^n \\ E^n \end{pmatrix} := \begin{pmatrix} F(U^n) \\ G \end{pmatrix} - \begin{pmatrix} A(U^n) & B_1(U^n)^T \\ B & 0 \end{pmatrix} \begin{pmatrix} U^n \\ P^n \end{pmatrix},$$

(ii) update $(U^{n+1}, P^{n+1}) := (U^n, P^n) + \omega\,(V^n, Q^n)\,,$

where ω is a relaxation parameter.

The main part of this iteration is the approximate solution of the discrete Oseen problem (6.10) which is done by applying several cycles of the previously described multigrid method. Numerical experiments have shown that it is sufficient to stop the multigrid iteration for solving (6.10) if the norm of the initial residual has been damped by a factor of 10.

7. Numerical tests. In this section, we compare by means of numerical tests the two new multigrid methods for the QDF basis described in section 6.1, which we denote by and , with the corresponding two multigrid methods using the Vanka smoother for the usual basis which we denote by and .

We consider three different 2D-problems: a Stokes problem, the driven cavity problem, and a benchmark flow problem.

All numerical computations were carried out on a Linux PC (Pentium IV, 2.8 GHz) using the code MooNMD [15]. The presented times are given in seconds.

If not mentioned otherwise, all multigrid approaches use an $F(1,1)$-cycle. The F-cycle combines the good convergence properties of a W-cycle with the low costs of a V-cycle. For all multigrid methods we choose a damping parameter of 0.9 within the smoother.

7.1. Stokes problem. We consider the Stokes problem in the unit square $\Omega = (0,1)^2$ with the prescribed solution

$$u(x,y) := \begin{pmatrix} \sin^2(k\pi x)\sin(2k\pi y) \\ \sin(2k\pi x)\sin^2(k\pi y) \end{pmatrix}, \qquad p(x,y) := \sin(k\pi x) - \sin(k\pi y), \qquad k = 5,$$

the data $f := -\operatorname{div} \mathbb{T}(u,p)$, $\nu = 1$, and $g \equiv 0$. Note that the given flow field describes an array of $k \times k$ vortices.

The coarsest grid $\mathcal{T}^{(0)}$ is a decomposition of the unit square into four congruent squares. Our results are presented for the finest grid $\mathcal{T}_h = \mathcal{T}^{(7)}$ generated by seven uniform refinement steps. This grid consists of 256×256 squares. We use Q_2-elements for the approximation of the velocity and discontinuous P_1-elements to approximate the pressure. On the grid $\mathcal{T}^{(7)}$, this leads in the case of using the usual finite element basis to $526,338$ degrees of freedom (d.o.f.) for the velocity and $196,608$ d.o.f. for the pressure while the QDF basis needs only $395,266$ d.o.f. for the velocity and $65,536$ d.o.f. for the piecewise constant pressure within the reduced Stokes problem.

In order to show the independence of the solution process from the starting vector, we made for each of the four multigrid methods 99 runs with randomly chosen starting vectors.

Table 7.1 gives for all considered multigrid methods the total run time (t_{total}), the time for solving the linear system (t_{solve}), the number of multigrid cycles needed

TABLE 7.1
Results for different multigrid approaches; minimum and maximum of the mentioned quantities are given.

	usual		MDML		QDF		QDF-MDML	
	min	max	min	max	min	max	min	max
t_{total}	104.8	105.8	83.0	92.1	44.9	45.8	64.1	69.3
t_{solve}	96.0	96.9	73.6	82.0	36.3	36.9	55.0	59.9
# cycles	14	14	12	12	20	20	10	10
t_{cycle}	6.86	6.92	6.13	6.84	1.81	1.84	5.50	5.99
aver. rate	0.0833	0.0840	0.0578	0.0588	0.194	0.195	0.0319	0.0322
memory	330		326		292		296	

to achieve a residual less than 10^{-11} ( ), the averaged time per cycle (t_{cycle}), the average convergence rate per cycle ( ), and the needed memory in MByte ( ). Note that due to a better performance, a $F(2,2)$-cycle was used for QDF-MDML. Table 7.1 shows that the convergence of all considered multigrid methods is independent from the starting vector of the iterative solution process. Although the average convergence rate for the QDF method is the worst one, the very small cycle time results at the end in the shortest total run time. Compared to the QDF method, the QDF-MDML method needs more computation time since the local systems within the smoother for the lowest order element Q_1^{rot}/P_0 are larger than for the QDF basis. In contrast to this, the MDML method performs faster than the usual multigrid method because the local systems for the Q_1^{rot}/P_0-element are much smaller than for the Q_2/P_1-element.

7.2. Driven cavity problem. Our second example is the driven cavity problem in the unit square with $\nu = 1/3000$. Since the standard Galerkin discretization for this Reynolds number lacks of stability, we use the reduced streamline diffusion finite element method as a stabilization technique, [6]. Furthermore, we restrict our tests to the MDML and the QDF-MDML multigrid approaches since they have shown to be superior to the other approaches. For the underlying Q_1^{rot}/P_0-element on the lower multigrid levels, an upwind discretization is used which is very robust and stabilizes the whole multigrid method. All calculations were carried out on $\mathcal{T}^{(6)}$ as finest mesh which consists of 128×128 squares.

Instead of solving the linear systems themselves by multigrid methods we applied a preconditioned flexible GMRES method [19] as solver where we have chosen either the MDML or the QDF-MDML multigrid method as preconditioner.

In Table 7.2 we present the total run time t_{total}, the solver time t_{solve}, and the time t_{mod} which is needed to modify the matrices and right-hand sides according to the QDF basis. Moreover, we show the ratio between the corresponding times for

TABLE 7.2
Total time and solver time for the driven cavity problem.

FE pair	MDML		QDF-MDML	
	t_{total}	t_{solve}	t_{total}/ratio	$t_{\text{solve}} + t_{\text{mod}}$/ratio
Q_2/P_1	963	839	497/ 1.94	336 + 23/ 2.34
Q_3/P_2	3,205	2,803	1,139/ 2.81	555 + 121/ 4.15
Q_4/P_3	10,431	8,846	3,377/ 3.09	885 + 542/ 6.20

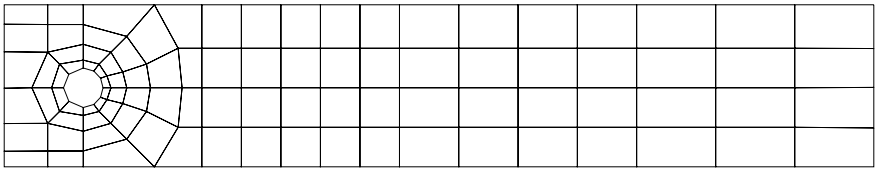


FIG. 7.1. *Coarsest mesh for the benchmark problem.*

the multigrid methods MDML and QDF-MDML. If we look at the solver time we see that the ratio increases from around 2 for the Q_2/P_1-element up to 6 for the Q_4/P_3-element. That means that the advantage of using the new multigrid method based on the QDF basis gets bigger with increasing polynomial degree of the finite elements. In contrast, the ratio of the total run time increases rather slowly for increasing polynomial degree. This is caused by the fact that the time used for assembling the matrices and right-hand sides, which does not belong to the solver time, is relatively large for higher order discretizations; see Table 7.2.

7.3. Flow around a circle. Our final example is the benchmark problem [illegible]; see [20]. We consider the Navier–Stokes equations with $\nu = 10^{-3}$. The coarsest mesh $\mathcal{T}_h^{(0)}$ for our computations is shown in Figure 7.1. A parabolic inflow and outflow profile is prescribed at the left and right boundary of the channel while $g \equiv 0$ on all other boundary parts.

A benchmark quantity to be computed is the lift coefficient c_l related to the circle; see [20] for a detailed definition. For the computation of c_l, we have used in our numerical tests the equivalent formula given, e.g., in [14].

We will now compare the performance of the preconditioned flexible GMRES method with the MDML and the QDF-MDML multigrid method as preconditioner for the Q_r/P_{r-1}-element with increasing r. Starting from the coarse mesh $\mathcal{T}_h^{(0)}$ shown in Figure 7.1, all finer meshes are obtained by uniform refinement with boundary adaption at the circle. In the discretization, the curved boundary of the circle was taken into account by means of isoparametric finite elements. Since the problem corresponds to a moderate physical Reynolds number of 20 we have used the standard Galerkin discretization which works well here.

TABLE 7.3
Total run time and error $e(c_l)$ of benchmark quantity c_l.

FE pair	level	total run time			$e(c_l)$
		MDML	QDF-MDML	ratio	
Q_2/P_1	2	20.0	8.9	2.25	8.71-06
Q_2/P_1	3	62.6	35.7	1.75	7.36-07
Q_2/P_1	4	252.7	185.6	1.36	6.81-08
Q_3/P_2	2	54.9	21.5	2.55	-8.76-08
Q_3/P_2	3	190.0	85.3	2.23	-1.17-08
Q_3/P_2	4	894.8	358.4	2.50	-4.84-10
Q_4/P_3	2	175.0	59.1	2.96	-2.50-08
Q_4/P_3	3	600.5	229.1	2.62	2.84-10
Q_4/P_3	4	2962.8	1015.6	2.92	1.02-10

Table 7.3 shows the error $e(c_l)$ of the calculated quantity c_l with respect to the reference parameter

$$c_{l,\mathrm{ref}} = 0.010618937712,$$

which was obtained in [14]. We see that higher order methods are able to produce more accurate results on coarser meshes in less time than lower order methods on even finer meshes. The ratio between the total run times needed for the MDML method and the QDF-MDML method ranges between 1.5 and 3. Note again that for higher order discretizations the time for assembling matrices and right-hand sides can't be neglected.

REFERENCES

[1] D. N. Arnold, D. Boffi, and R. S. Falk, *Approximation by quadrilateral finite elements*, Math. Comp., 71 (2002), pp. 909–922.

[2] M. Benzi, G. H. Golub, and J. Liesen, *Numerical solution of saddle point problems*, Acta Numer., 14 (2005), pp. 1–137.

[3] S. C. Brenner, *A nonconforming multigrid method for the stationary Stokes equations*, Math. Comp., 55 (1990), pp. 411–437.

[4] F. Brezzi and M. Fortin, *Mixed and Hybrid Finite Element Methods*, in Springer Series in Computational Mathematics, 15, Springer-Verlag, New York, 1991.

[5] H. C. Elman, D. J. Silvester, and A. J. Wathen, *Performance and analysis of saddle point preconditioners for the discrete steady-state Navier-Stokes equations*, Numer. Math., 90 (2002), pp. 665–688.

[6] T. Gelhard, G. Lube, M. A. Olshanskii, and J.-H. Starcke, *Stabilized finite element schemes with LBB-stable elements for incompressible flows*, J. Comput. Appl. Math., 177 (2005), pp. 243–267.

[7] V. Girault and P.-A. Raviart, *Finite Element Methods for Navier–Stokes Equations*, in Springer Series in Computational Mathematics, 5, Springer-Verlag, Berlin, 1986.

[8] D. F. Griffiths, *Finite elements for incompressible flow*, Math. Methods Appl. Sci., 1 (1979), pp. 16–31.

[9] D. F. Griffiths, *An approximately divergence-free* 9*-node velocity element (with variations) for incompressible flows*, Internat. J. Numer. Methods Fluids, 1 (1981), pp. 323–346.

[10] F. Hecht, *Construction of a basis at free divergence in finite element and application to the Navier-Stokes equations*, in Numerical Solutions of Nonlinear Problems, (Rocquencourt, 1983), INRIA, Rocquencourt, 1984, pp. 284–297.

[11] V. John, *Higher order finite element methods and multigrid solvers in a benchmark problem for the* 3*D Navier-Stokes equations*, Internat. J. Numer. Methods Fluids, 40 (2002), pp. 775–798.

[12] V. John, *Large eddy simulation of turbulent incompressible flows. Analytical and numerical results for a class of LES models*, Lecture Notes in Computational Science and Engineering, 34, Springer-Verlag, Berlin, 2004.

[13] V. John, P. Knobloch, G. Matthies, and L. Tobiska, *Non-nested multi-level solvers for finite element discretisations of mixed problems*, Computing, 68 (2002), pp. 313–341.

[14] V. John and G. Matthies, *Higher-order finite element discretizations in a benchmark problem for incompressible flows*, Internat. J. Numer. Methods Fluids, 37 (2001), pp. 885–903.

[15] V. John and G. Matthies, *MooNMD—A program package based on mapped finite element methods*, Comput. Vis. Sci., 6 (2004), pp. 163–169.

[16] G. Matthies, *Mapped finite elements on hexahedra. Necessary and sufficient conditions for optimal interpolation errors*, Numer. Algorithms, 27 (2001), pp. 317–327.

[17] G. Matthies and L. Tobiska, *The inf-sup condition for the mapped* Q_k*-*P_{k-1}^{disc} *element in arbitrary space dimensions*, Computing, 69 (2002), pp. 119–139.

[18] M. A. Olshanskii and A. Reusken, *Navier–Stokes equations in rotation form: A robust multigrid solver for the velocity problem*, SIAM J. Sci. Comput., 23 (2002), pp. 1683–1706.

[19] Y. Saad, *A flexible inner-outer preconditioned GMRES algorithm*, SIAM J. Sci. Comput., 14 (1993), pp. 461–469.

[20] M. Schäfer and S. Turek, *The benchmark problem "Flow around a cylinder"*, in Flow Simulation with High-Performance Computers II, E. H. Hirschel, ed., Notes on Numerical Fluid Mechanics 52, Vieweg, Braunschweig, Germany, 1996, pp. 547–566.

[21] F. Schieweck, *A General Transfer Operator for Arbitrary Finite Element Spaces*, preprint 25/00, Otto-von-Guericke-Universität Magdeburg, Fakultät für Mathematik, 2000.

[22] F. Schieweck, *Construction of Higher Order Discretely Divergence Free Finite Elements for Incompressible Flow*, preprint 34/02, Otto-von-Guericke-Universität Magdeburg, Fakultät für Mathematik, 2002.

[23] F. Thomasset, *Implementation of finite element methods for Navier-Stokes equations*, Springer Series in Computational Physics, Springer-Verlag, New York-Berlin, 1981.

[24] S. Turek, *Multigrid techniques for a divergence-free finite element discretization*, East-West J. Numer. Math., 2 (1994), pp. 229–255.

[25] S. Turek, *Efficient solvers for incompressible flow problems. An algorithmic and computational approach*, Lecture Notes in Computational Science and Engineering, 6, Springer-Verlag, Berlin, 1999.

[26] X. Ye and G. Anderson, *The derivation of minimal support basis functions for the discrete divergence operator*, J. Comput. Appl. Math., 61 (1995), pp. 105–116.

[27] X. Ye and C. A. Hall, *The construction of an optimal weakly divergence-free macroelement*, Internat. J. Numer. Methods Engrg., 36 (1993), pp. 2245–2262.

[28] X. Ye and C. A. Hall, *A discrete divergence-free basis for finite element methods*, Numer. Algorithms, 16 (1997), pp. 365–380.

SIAM J. SCI. COMPUT.
Vol. 28, No. 1, pp. 172–205
© 2006 Society for Industrial and Applied Mathematics

NUMERICAL MODELING OF ELASTIC WAVES ACROSS IMPERFECT CONTACTS.*

BRUNO LOMBARD† AND JOËL PIRAUX†

Abstract. A numerical method is described for studying how elastic waves interact with imperfect contacts such as fractures or glue layers existing between elastic solids. These contacts have been classicaly modeled by interfaces, using a simple rheological model consisting of a combination of normal and tangential linear springs and masses. The jump conditions satisfied by the elastic fields along the interfaces are called the "spring-mass conditions." By tuning the stiffness and mass values, it is possible to model various degrees of contact, from perfect bonding to stress-free surfaces. The conservation laws satisfied outside the interfaces are integrated using classical finite-difference schemes. The key problem arising here is how to discretize the spring-mass conditions and how to insert them into a finite-difference scheme: this was the aim of the present paper. For this purpose, we adapted an interface method previously developed for use with perfect contacts [*J. Comput. Phys.*, 195 (2004), pp. 90–116]. This numerical method also describes closely the geometry of arbitrarily shaped interfaces on a uniform Cartesian grid, at negligible extra computational cost. Comparisons with original analytical solutions show the efficiency of this approach.

Key words. elastic waves, interface methods, spring-mass jump conditions, discontinuous coefficients, imperfect contact, hyperbolic conservation laws.

AMS subject classifications. 35L40, 65M06

DOI. 10.1137/05062740X

1. Introduction.

Here we propose to study the propagation of mechanical waves in an elastic medium divided into several subdomains. The wavelengths are assumed to be much larger than the thickness of the contact zones between subdomains, or [illegible] [18]. Each interphase is replaced by a zero-thickness [illegible], where elastic fields satisfy jump conditions. In elastodynamics, the contacts between elastic media are usually assumed to be perfect [1]. They can therefore be modeled by perfect jump conditions, such as perfectly bonded, perfectly lubricated, or stress-free conditions. For example, perfectly bonded conditions will mean that both elastic displacements and normal elastic stresses are continuous across the interface at each time step.

In practice, contacts are often imperfect because of the presence of microcracks or interstitial media in the interphase. Take, for example, fractures in the earth, which are filled with air or liquid, and where jumps occur in the elastic displacements and elastic stresses. The simplest imperfect conditions are the [illegible] (which are sometimes called "linear slip displacements"): these conditions are realistic in the case of incident waves with very small amplitudes [17]. The spring-mass conditions have been extensively studied, both theoretically and experimentally [18, 20, 21]. This approach has been applied in various disciplines, such as nondestructive evaluation of materials [2, 22] and geophysics [17].

However, very few studies have dealt so far with the numerical simulation of wave propagation across imperfect contacts described by spring-mass conditions. To our knowledge, only three approaches have been proposed for this purpose. First, Gu, Nihei, and Myer developed a boundary integral method which can be applied to

*Received by the editors March 23, 2005; accepted for publication (in revised form) September 29, 2005; published electronically March 3, 2006.
http://www.siam.org/journals/sisc/28-1/62740.html
†Laboratoire de Mécanique et d'Acoustique, 31 chemin Joseph Aiguier, 13402 Marseille, France (lombard@lma.cnrs-mrs.fr, piraux@lma.cnrs-mrs.fr).

arbitrarily shaped interfaces [7], but this method requires knowing the Green functions on both sides of the interface, which complicates the study of realistic heterogeneous media. Second, a finite-element method has been proposed by Haney and Sneider [8]. The jump conditions are incorporated automatically here into the numerical scheme, via the variational formulation, and attempts have been made to perform numerical analysis. The main drawback of this approach is that it requires adapting the mesh to the interface, which increases the computational effort. Third, other authors have approached this problem by implicitly accounting for the boundary conditions by using an equivalent medium and then deriving new finite-difference formulas [3, 4]. This approach can be applied to arbitrarily shaped interfaces on a uniform Cartesian grid, but the accuracy is low, and this method involves explicitly changing the numerical scheme near the interface. Note that the inertial effects were not investigated in any of these three cases, although they may be important factors [18].

The aim of this paper is to describe a procedure for incorporating the spring-mass conditions into existing finite-difference schemes, on a regular Cartesian grid. The geometry of arbitrarily shaped interfaces is properly taken into account, reducing the unwanted diffraction classically induced by the Cartesian grid. Last, the extra computational cost is low. For this purpose, we adapted the [illegible] (ESIM) previously developed for dealing with perfect contacts in one dimension [15] and two dimensions [12]. A study has also dealt with imperfect contacts in one dimension [11]. In the present study, this approach is extended to two-dimensional configurations. The focus here is on the description of imperfect contacts; to avoid additional complications, we take media with simple constitutive laws (elastic and isotropic media), but the procedure should be suitable for dealing with more realistic media. The inertial effects are taken into account.

This paper is organized as follows. In section 2, the problem is stated in terms of the configuration, the conservation laws, and the spring-mass conditions. In section 3, a numerical strategy is described for integrating the conservation laws on the whole computational domain: the same scheme is used throughout the domain, but near an interface [illegible] of the solution are used, which implicitly account for the spring-mass conditions. Section 4 is the core part of this paper: it describes in detail how to compute the modified values. Numerical experiments are described in section 5, and comparisons with original analytical solutions show the efficiency of the method. Note that although no rigorous mathematical proof of the validity of the algorithms was obtained, the results of the numerical experiments performed were extremely satisfactory.

2. Problem statement.

2.1. Configuration. Let us consider two isotropic elastic media Ω_0 and Ω_1 separated by a stationary interface Γ (Figure 2.1). We study a two-dimensional configuration with plane strains and adopt Cartesian coordinates x and y pointing rightward and upward, respectively. The interface is described by a parametric description $(x(\tau), y(\tau))$. The unit tangential vector $\boldsymbol{t}$ and the unit normal vector $\boldsymbol{n}$ are

$$\boldsymbol{t} = \frac{1}{\sqrt{x'^2 + y'^2}} \begin{pmatrix} x' \\ y' \end{pmatrix}, \qquad \boldsymbol{n} = \frac{1}{\sqrt{x'^2 + y'^2}} \begin{pmatrix} -y' \\ x' \end{pmatrix}, \tag{2.1}$$

where $x' = \frac{dx}{d\tau}$ and $y' = \frac{dy}{d\tau}$. Γ is assumed to be sufficiently smooth to ensure that $x(\tau)$, $y(\tau)$ and their successive spatial derivatives are continuous all along Γ, up to a given order of derivation.

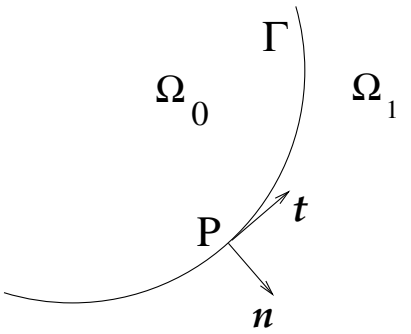


FIG. 2.1. *Interface* Γ *between two elastic media* Ω_0 *et* Ω_1.

The physical parameters are the density ρ, the elastic speed of the compressional P-waves c_p, and the elastic speed of the shear SV-waves c_s. For the sake of simplification, these parameters are taken to be piecewise constant; however, they may be discontinuous across Γ

$$(\rho,\, c_p,\, c_s) = \begin{cases} (\rho_0,\, c_{p0},\, c_{s0}) & \text{if} \quad (x,\, y) \in \Omega_0, \\ (\rho_1,\, c_{p1},\, c_{s1}) & \text{if} \quad (x,\, y) \in \Omega_1. \end{cases} \tag{2.2}$$

The elastic fields are the two components of the elastic velocity $\boldsymbol{v}(v_1, v_2)$ and the three independent components of the elastic stress tensor $\boldsymbol{\sigma}(\sigma_{11}, \sigma_{12}, \sigma_{22})$. The projections, normal and tangential to the interface, of the elastic displacement $\boldsymbol{u}(u_1, u_2)$, those of the velocity $\boldsymbol{v}$, and those of the normal stress $\boldsymbol{\sigma.n}$ are denoted by

$$\left| \begin{array}{l} u_N = \boldsymbol{u.n}, \\ u_T = \boldsymbol{u.t}, \end{array} \right. \quad \left| \begin{array}{l} v_N = \boldsymbol{v.n}, \\ v_T = \boldsymbol{v.t}, \end{array} \right. \quad \left| \begin{array}{l} \sigma_N = (\boldsymbol{\sigma.n}).\boldsymbol{n}, \\ \sigma_T = (\boldsymbol{\sigma.n}).\boldsymbol{t}. \end{array} \right. \tag{2.3}$$

Let P be a point on Γ (Figure 2.1), and t be the time. Given a function $f(x,\, y,\, t)$, the limit values of f at P on both sides of Γ are written

$$f_l(P,\, t) = \lim_{M \to P, M \in \Omega_l} f(M,\, t), \tag{2.4}$$

where $l = 0, 1$. The jump of f across Γ, from Ω_0 to Ω_1, is denoted by

$$[f(P,\, t)] = f_1(P,\, t) - f_0(P,\, t). \tag{2.5}$$

2.2. Conservation laws. To study the propagation of small perturbations in Ω_i $(i = 0, 1)$, we use a velocity-stress formulation of elastodynamic equations. Setting

$$\boldsymbol{U} = {}^T(v_1, v_2, \sigma_{11}, \sigma_{12}, \sigma_{22}), \tag{2.6}$$

the linearization of mechanics equations gives a first-order linear hyperbolic system in each subdomain

$$\frac{\partial}{\partial t}\boldsymbol{U} + \boldsymbol{A}_l \frac{\partial}{\partial x}\boldsymbol{U} + \boldsymbol{B}_l \frac{\partial}{\partial y}\boldsymbol{U} = \boldsymbol{0}, \qquad l = 0, 1, \tag{2.7}$$

which is satisfied outside Γ. The 5×5 piecewise-constant matrices $\boldsymbol{A}_l$ and $\boldsymbol{B}_l$ $(l=0,1)$ are [1]

$$\boldsymbol{A}_l = -\begin{pmatrix} 0 & 0 & \dfrac{1}{\rho} & 0 & 0 \\ 0 & 0 & 0 & \dfrac{1}{\rho} & 0 \\ \rho\, c_p^2 & 0 & 0 & 0 & 0 \\ 0 & \rho\, c_s^2 & 0 & 0 & 0 \\ \rho\left(c_p^2 - 2\, c_s^2\right) & 0 & 0 & 0 & 0 \end{pmatrix},$$

$$\boldsymbol{B}_l = -\begin{pmatrix} 0 & 0 & 0 & \dfrac{1}{\rho} & 0 \\ 0 & 0 & 0 & 0 & \dfrac{1}{\rho} \\ 0 & \rho\left(c_p^2 - 2\, c_s^2\right) & 0 & 0 & 0 \\ \rho\, c_s^2 & 0 & 0 & 0 & 0 \\ 0 & \rho\, c_p^2 & 0 & 0 & 0 \end{pmatrix}. \tag{2.8}$$

2.3. Spring-mass conditions. An incident wave at the interface generates four other waves: a reflected P-wave, a reflected SV-wave, a transmitted P-wave, and a transmitted SV-wave. To pose the problem suitably, it is necessary to define four independent jump conditions satisfied by the fields along Γ (Figure 2.1). [illegible] are generally used for this purpose, namely,

$$\begin{aligned} &[u_N(P,\,t)] = 0, \qquad [\sigma_N(P,\,t)] = 0, \\ &[u_T(P,\,t)] = 0, \qquad [\sigma_T(P,\,t)] = 0, \end{aligned} \tag{2.9}$$

corresponding to a perfectly bonded contact between the two solids in question.

To describe an imperfect contact, one can generalize (2.9) into [illegible]. With the notations defined in (2.4), the spring-mass conditions are

$$\boxed{\begin{aligned} &[u_N(P,\,t)] = \frac{1}{K_N}\,\sigma_{N0}(P,\,t), \qquad [\sigma_N(P,\,t)] = M_N\,\frac{\partial^2}{\partial\, t^2}\,u_{N0}(P,\,t), \\ &[u_T(P,\,t)] = \frac{1}{K_T}\,\sigma_{T0}(P,\,t), \qquad [\sigma_T(P,\,t)] = M_T\,\frac{\partial^2}{\partial\, t^2}\,u_{T0}(P,\,t), \end{aligned}} \tag{2.10}$$

where $K_N > 0$, $K_T > 0$, $M_N \geq 0$, $M_T \geq 0$ are called the normal stiffness, the tangential stiffness, the normal mass, and the tangential mass of the interface, respectively. The conditions (2.10) are called "spring-mass conditions" because of analogies with the equations governing the dynamics of a spring-mass system (Figure 2.2). One basic underlying assumption made here is that the elastic stresses do not affect the nature of the contact and hence that K_N, K_T, M_N, and M_T do not depend on t or on the fields. They can vary with space and hence with the parameter τ.

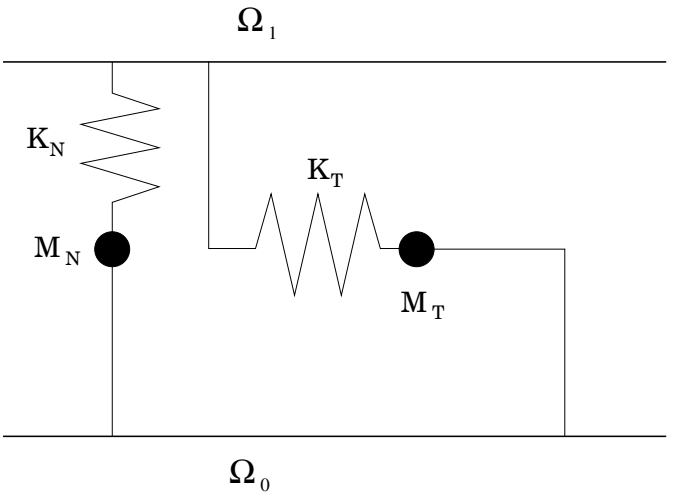


FIG. 2.2. *Spring-mass rheological model of the contact.*

The spring-mass conditions provide an easy way of describing a wide range of contacts between solids, from perfect contact to disconnected media. For $K_N \to +\infty$, $K_T \to +\infty$, $M_N = 0$, and $M_T = 0$, we obtain the perfectly bonded conditions (2.9). For $K_N \to +\infty$, $K_T \to 0$, $M_N = 0$, and $M_T = 0$, we obtain $\sigma_T(P, t) \to 0$, which amounts to a perfect slip with no friction. Lastly, for $K_N \to 0$, $K_T \to 0$, $M_N = 0$, and $M_T = 0$, we obtain $\sigma_N(P, t) \to 0$ and $\sigma_T(P, t) \to 0$, hence $\boldsymbol{\sigma}.\boldsymbol{n}(P, t) \to \mathbf{0}$: the media Ω_0 and Ω_1 tend to have stress-free boundaries, which means that no waves are transmitted from one medium to the other.

The spring-mass conditions entail an important property: the plane waves reflected and transmitted by a plane interface with conditions (2.10) are frequency dependent [21]. These waves therefore show a distorted profile that is quite different from the profile of the incident wave, even below the critical angle. In addition, even if the incident wave is spatially bounded in the direction of propagation, the reflected and transmitted waves are not spatially bounded: a "coda" follows each of these waves. Phenomena of this kind, which do not occur with perfect conditions, are observed experimentally (see, e.g., [17]).

By choosing appropriate values of K_N, K_T, M_N, and M_T, it is possible to model realistic configurations. The spring-mass conditions (2.10) can also be obtained quite rigorously in some cases; the values of K_N, K_T, M_N, and M_T will therefore depend on the physical and geometrical properties of the interphase. Take a plane elastic layer sandwiched between two homogeneous isotropic half-spaces. If the thickness of the intermediate layer is much smaller than the wavelength, the spring-mass conditions can be deduced from an asymptotic analysis of the wave propagation behavior inside the layer [18].

The spring-mass conditions model (2.10) has some limitations. First, if $K_N < +\infty$ and $M_N \neq 0$, or if $K_T < +\infty$ and $M_T \neq 0$, these conditions are asymmetrical. In Appendix A, we establish that the influence of this asymmetry is either null (in the case of reflected waves) or negligible (in that of transmitted waves) in the one-dimensional context. We have checked numerically that this is also the case in the two-dimensional context. Setting up symmetrical conditions would considerably complicate the jump conditions, in return for absolutely negligible effects.

The second drawback follows from the first equation given by (2.10): there is no reason why a negative jump of u_N should not occur, with an absolute value greater than the real thickness of the interphase. Since a penetration of both faces on the interphase is not physically realistic, the conditions (2.10) are valid only in the case of very small perturbations. With larger perturbations, finer modeling procedures are required, based on nonlinear contact laws [23]. Jump conditions of this kind, which are currently under study, require more complex numerical methods.

3. Time-marching.

3.1. Numerical scheme. To integrate the hyperbolic system (2.7), we introduce a uniform lattice of grid points: $(x_i, y_j, t_n) = (i\,\Delta x, j\,\Delta y, n\,\Delta t)$, where $\Delta x = \Delta y$ are the spatial mesh sizes and Δt is the time step. The approximation $\boldsymbol{U}_{i,j}^n$ of $\boldsymbol{U}(x_i, y_j, t_n)$ is computed using explicit two-step, spatially centered finite-difference schemes. The time-stepping of these schemes is written symbolically

$$\boldsymbol{U}_{i,j}^{n+1} = \boldsymbol{H}_{i,j}\left(\boldsymbol{U}_{i+\tilde{i},\,j+\tilde{j}}^{n},\ (\tilde{i},\,\tilde{j}) \in \Sigma\right). \tag{3.1}$$

$\boldsymbol{H}_{i,j}$ is a discrete operator and Σ is the stencil of the scheme. The subscripts in $\boldsymbol{H}_{i,j}$ refer to the physical parameters at $(x_i,\, y_j)$. See [9] for a review of the huge body of literature on numerical methods for conservation laws.

The interface Γ is immersed in the regular meshing, so that one can distinguish between two sets of grid points: the [illegible], where the stencil of the scheme involves a single medium, e.g., Ω_0 or Ω_1, and the [illegible], where the stencil of the scheme crosses Γ. The distribution of irregular points along Γ obviously depends on the geometry of Γ and on the stencil of the scheme. At regular points, the scheme (3.1) is applied classically, as in homogeneous media. At irregular points, however, the scheme (3.1) is modified to take the spring-mass conditions (2.10) into account. This modification is carried out using an interface method, and the main aim of the present article is to describe this procedure, which will be presented in detail in subsequent sections.

In the numerical experiments described in section 5, we use a second-order scheme: the wave propagation algorithm (WPALG), originally developed by LeVeque in the field of computational fluid dynamics [10]. Its stencil is ($\tilde{i} = -2 \ldots 2$, $\tilde{j} = -2 \ldots 2$) and $(\tilde{i},\, \tilde{j}) \neq (\pm 2, \pm 2)$. WPALG is a useful tool for dealing with linear elastic wave propagation, for at least three reasons. First, it involves the use of nonlinear flux limiters that prevent numerical dispersion. Second, this scheme reduces the numerical anisotropy introduced by the Cartesian grid. Third, WPALG is stable in two dimensions up to CFL = 1. For the convergence measurements performed in section 5 (test 1), we also used the standard second-order Lax–Wendroff scheme.

Note that other schemes can be used: in particular, we have successfully combined the interface method with staggered schemes, such as [19]. It should therefore be possible to adapt most solvers for use with the interface method described in the forthcoming discussion.

3.2. Interface method. From now on, we will focus on the time-stepping procedure near the interface. To take into account the spring-mass conditions satisfied along Γ, the scheme (3.1) is also applied at irregular points, but some of the numerical values used for the time-stepping procedure are changed. The use of so-called modified values deduced from the jump conditions is the key feature of the interface method developed in our previous studies: the "explicit simplified interface method"

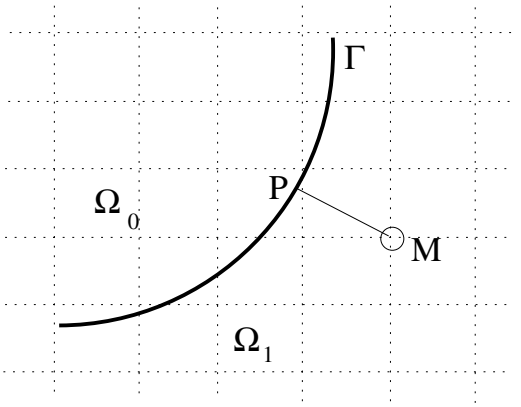


FIG. 3.1. *Zoom on an irregular point M; orthogonal projection P of M onto Γ.*

TABLE 3.1
n with indices used throughout the text ($l = 0, 1$).

	Value	Number of
n_v	$5(k+1)(k+2)/2$	components of $\boldsymbol{U}_l^k$
n_c	$2(k+1)(k+2)$	jump conditions
n_m	$k(k-1)/2$	compatibility conditions
n_q	variable	grid points to estimate $\boldsymbol{U}_l^k$

(ESIM) [11, 12, 15]. At time t_n, the general method used to compute modified values is as follows. On each side of Γ, one defines a smooth extension $\boldsymbol{U}^*(x, y, t_n)$ of the exact solution on the other side. The extension $\boldsymbol{U}^*$ is built satisfying the same jump conditions as the exact solution $\boldsymbol{U}$. At any irregular point, the modified value is a numerical estimate of $\boldsymbol{U}^*$ at this point.

Let us introduce some notation. Take an irregular point M with coordinates (x_I, y_J), belonging to Ω_1. (The following discussion can easily be adapted to the case where $M(x_I, y_J) \in \Omega_0$.) Let $P(x_P, y_P)$ be a point on Γ near M, for example, the closest orthogonal projection of M onto Γ (Figure 3.1). The vector containing the limit values of the exact solution $\boldsymbol{U}(x, y, t_n)$ and those of its spatial derivatives at P up to the kth order is denoted by

$$\boldsymbol{U}_l^k = \lim_{M \to P, M \in \Omega_l} {}^T\left({}^T\boldsymbol{U}, \ldots, \frac{\partial^\alpha}{\partial x^{\alpha-\beta}\, \partial y^\beta}\, {}^T\boldsymbol{U}, \ldots, \frac{\partial^k}{\partial y^k}\, {}^T\boldsymbol{U} \right), \tag{3.2}$$

where $l = 0$ or 1, $\alpha = 0, \ldots, k$, and $\beta = 0, \ldots, \alpha$. This vector has $n_v = 5(k+1)(k+2)/2$ components. Throughout the text, many n with indices have been used; to avoid any confusion, they are summed up in Table 3.1. To obtain concise expressions for the

kth order Taylor expansions at P of quantities at $(x_i,\, y_j)$, we define the $5 \times n_v$ matrix

$$\mathbf{\Pi}^k_{i,j} = \left(\boldsymbol{I}_5, \ldots, \frac{1}{\beta\,!\,(\alpha-\beta)\,!}\,(x_i - x_P)^{\alpha-\beta}(y_j - y_P)^{\beta}\boldsymbol{I}_5, \ldots, \frac{(y_j - y_P)^k}{k\,!}\,\boldsymbol{I}_5 \right), \tag{3.3}$$

where $\boldsymbol{I}_5$ is the 5×5 identity matrix, $\alpha = 0, \ldots, k$, and $\beta = 0, \ldots, \alpha$. The modified value $\boldsymbol{U}^*_{I,J}$ is then defined as a numerical estimate of the smooth extension

$$\boldsymbol{U}^*(x_I, y_J, t_n) = \mathbf{\Pi}^k_{I,J}\,\boldsymbol{U}^k_0. \tag{3.4}$$

Note that $\boldsymbol{U}^k_0$ is the limit value of the solution and its spatial derivatives on the other side of Γ with respect to M.

The modified values at all the irregular points surrounding Γ are computed in a similar way at t_n. The time-stepping at an irregular point (i, j) is written symbolically

$$\boldsymbol{U}^{n+1}_{i,j} = \boldsymbol{H}_{i,j}\left(\hat{\boldsymbol{U}}^n_{i+\tilde{i},\, j+\tilde{j}},\, (\tilde{i},\, \tilde{j}) \in \Sigma \right), \tag{3.5}$$

where

$$\hat{\boldsymbol{U}}_{i+\tilde{i},\, j+\tilde{j}} = \begin{cases} \boldsymbol{U}^n_{i+\tilde{i},\, j+\tilde{j}} \text{ if } (x_{i+\tilde{i}},\, y_{j+\tilde{j}}) \in \Omega_1, \\ \\ \boldsymbol{U}^*_{i+\tilde{i},\, j+\tilde{j}} \text{ if } (x_{i+\tilde{i}},\, y_{j+\tilde{j}}) \in \Omega_0. \end{cases} \tag{3.6}$$

The time-stepping procedure is completed over the whole computational domain by applying (3.1) at the regular points. It only remains now to calculate $\boldsymbol{U}^*$'s, since $\boldsymbol{U}^k_0$ in (3.4) is unknown.

4. Calculating modified values.

4.1. Differentiation of the spring-mass conditions. In the first step toward calculating the modified value (3.4), we look for the jump conditions satisfied by $\boldsymbol{U}^k_l$ (3.2) for any k. These conditions are deduced from the spring-mass conditions (2.10) satisfied by $u_{N,T}$ and $\sigma_{N,T}$. Before describing the procedure, let us introduce a new notation. The vector containing the limit values of the $(k+1)$th spatial derivatives of $\boldsymbol{U}(x, y, t)$ at P is denoted by

$$\overline{\boldsymbol{U}}^{k+1}_l = \lim_{M \to P,\, M \in \Omega_l} {}^T\left(\frac{\partial^{k+1}}{\partial\, x^{k+1}}\,{}^T\boldsymbol{U}, \ldots, \frac{\partial^{k+1}}{\partial\, x^{k+1-\alpha}\,\partial\, y^{\alpha}}\,{}^T\boldsymbol{U}, \ldots, \frac{\partial^{k+1}}{\partial\, y^{k+1}}\,{}^T\boldsymbol{U} \right), \tag{4.1}$$

where $l = 0$ or 1, $\alpha = 0, \ldots, k+1$. This vector has $5(k+2)$ components. Once again, the point P considered and the instant t are omitted.

For $k = 0$, the two equations in (2.10) that deal with the jump in the elastic displacement are differentiated in terms of t. The geometry of Γ and stiffness and mass values do not depend on t; since $\boldsymbol{v} = \frac{\partial \boldsymbol{u}}{\partial\, t}$, we obtain

$$\begin{aligned} [v_N(P,\, t)] &= \frac{1}{K_N}\,\frac{\partial}{\partial\, t}\,\sigma_{N0}(P,\, t), \qquad [\sigma_N(P,\, t)] = M_N\,\frac{\partial}{\partial\, t}\,v_{N0}(P,\, t), \\ [v_T(P,\, t)] &= \frac{1}{K_T}\,\frac{\partial}{\partial\, t}\,\sigma_{T0}(P,\, t), \qquad [\sigma_T(P,\, t)] = M_T\,\frac{\partial}{\partial\, t}\,v_{T0}(P,\, t). \end{aligned} \tag{4.2}$$

The time derivatives in (4.2) are replaced by spatial derivatives thanks to the conservation laws (2.7). We sum up the relations thus obtained in matrix terms

$$\boldsymbol{C}_1^0\,\boldsymbol{U}_1^0 = \boldsymbol{C}_0^0\,\boldsymbol{U}_0^0 + \boldsymbol{E}_0^0\,\overline{\boldsymbol{U}}_0^1. \tag{4.3}$$

$\boldsymbol{C}_l^0$ $(l = 0, 1)$ are 4×5 matrices; $\boldsymbol{U}_l^0$ are the vectors (3.2) for $k = 0$ (i.e., the limit values of (2.6)); $\boldsymbol{E}_0^0$ is a 4×10 matrix; last, $\overline{\boldsymbol{U}}_0^1$ is the vector (4.1) for $l = 0$ and $k = 0$. Matrices $\boldsymbol{C}_l^0$ describe the perfectly bonded conditions (2.9). Matrix $\boldsymbol{E}_0^0$ describes the correction induced by the springs and masses in (2.10). Both matrices $\boldsymbol{C}_l^0$ and $\boldsymbol{E}_0^0$ depend on τ, but they are independent of t; they are dealt with in greater detail in Appendix B.

To compute the conditions satisfied up to $k = 1$, we differentiate (4.3) in terms of t and τ. First, the differentiation of (4.3) in terms of t yields

$$\boldsymbol{C}_1^0\,\frac{\partial}{\partial t}\,\boldsymbol{U}_1^0 = \boldsymbol{C}_0^0\,\frac{\partial}{\partial t}\,\boldsymbol{U}_0^0 + \boldsymbol{E}_0^0\,\frac{\partial}{\partial t}\,\overline{\boldsymbol{U}}_0^1. \tag{4.4}$$

The time derivatives in (4.4) are replaced by spatial derivatives, using the conservation laws (2.7); with the notations (3.2) and (4.1), one readily obtains $(l = 0, 1)$

$$\frac{\partial}{\partial t}\,\boldsymbol{U}_l^0 = \begin{pmatrix} \mathbf{0} & -\boldsymbol{A}_l & -\boldsymbol{B}_l \end{pmatrix} \boldsymbol{U}_l^1, \qquad \frac{\partial}{\partial t}\,\overline{\boldsymbol{U}}_0^1 = \begin{pmatrix} -\boldsymbol{A}_0 & -\boldsymbol{B}_0 & \mathbf{0} \\ \mathbf{0} & -\boldsymbol{A}_0 & -\boldsymbol{B}_0 \end{pmatrix} \overline{\boldsymbol{U}}_0^2,$$

which are injected into (4.4). Second, the differentiation of (4.3) in terms of τ yields

$$\left(\frac{d}{d\tau}\,\boldsymbol{C}_1^0\right) \boldsymbol{U}_1^0 + \boldsymbol{C}_1^0\,\frac{\partial}{\partial \tau}\,\boldsymbol{U}_1^0 = \left(\frac{d}{d\tau}\,\boldsymbol{C}_0^0\right) \boldsymbol{U}_0^0 + \boldsymbol{C}_0^0\,\frac{\partial}{\partial \tau}\,\boldsymbol{U}_0^0 + \left(\frac{d}{d\tau}\,\boldsymbol{E}_0^0\right) \overline{\boldsymbol{U}}_0^1 + \boldsymbol{E}_0^0\,\frac{\partial}{\partial \tau}\,\overline{\boldsymbol{U}}_0^1. \tag{4.5}$$

Since $\boldsymbol{U}_l^0$ $(l = 0, 1)$ and $\overline{\boldsymbol{U}}_0^1$ depend on $x(\tau)$ and $y(\tau)$, the chain-rule gives

$$\begin{aligned} l = 0, 1, \qquad \frac{\partial}{\partial \tau}\,\boldsymbol{U}_l^0 &= x'\,\frac{\partial}{\partial x}\,\boldsymbol{U}_l^0 + y'\,\frac{\partial}{\partial y}\,\boldsymbol{U}_l^0 \\ &= \begin{pmatrix} \mathbf{0} & x'\boldsymbol{I}_5 & y'\boldsymbol{I}_5 \end{pmatrix} \boldsymbol{U}_l^1 \end{aligned}$$

and

$$\frac{\partial}{\partial \tau}\,\overline{\boldsymbol{U}}_0^1 = \begin{pmatrix} x'\boldsymbol{I}_5 & y'\boldsymbol{I}_5 & \mathbf{0} \\ \mathbf{0} & x'\boldsymbol{I}_5 & y'\boldsymbol{I}_5 \end{pmatrix} \overline{\boldsymbol{U}}_0^2.$$

Due to the normalization of vectors $\boldsymbol{n}$ and $\boldsymbol{t}$ in (2.1) and (2.3), special care must be taken with the differentiation procedure $\frac{d}{d\tau}\,\boldsymbol{E}_0^0$ in (4.5) (see Appendix B). From the previous discussion, one builds three 12×15 matrices $\boldsymbol{C}_0^1$, $\boldsymbol{C}_1^1$ and $\boldsymbol{D}_0^1$, and one 12×15 matrix $\boldsymbol{E}_0^1$, so that

$$\boldsymbol{C}_1^1\,\boldsymbol{U}_1^1 = \left(\boldsymbol{C}_0^1 + \boldsymbol{D}_0^1\right) \boldsymbol{U}_0^1 + \boldsymbol{E}_0^1\,\overline{\boldsymbol{U}}_0^2. \tag{4.6}$$

Matrices $\boldsymbol{C}_l^1$ describe the influence of perfectly bonded conditions. Matrices $\boldsymbol{D}_0^1$ and $\boldsymbol{E}_0^1$ describe the changes introduced by the springs and masses.

By iterating a similar procedure $(k-1)$ times, one can find matrices such that

$$\boxed{\boldsymbol{C}_1^k\,\boldsymbol{U}_1^k = \left(\boldsymbol{C}_0^k + \boldsymbol{D}_0^k\right)\,\boldsymbol{U}_0^k + \boldsymbol{E}_0^k\,\overline{\boldsymbol{U}}_0^{k+1},} \tag{4.7}$$

where $\boldsymbol{C}_0^k$, $\boldsymbol{C}_1^k$, and $\boldsymbol{D}_0^k$ are $n_c \times n_v$ matrices with $n_c = 2(k+1)(k+2)$ and $n_v = 5(k+1)(k+2)/2$; $\boldsymbol{E}_0^k$ is a $n_c \times 5(k+2)$ matrix. (In practice and as shown in section 4.4, the last matrix is never computed.) This is a tedious procedure, however, even with low values of k (e.g., $k=1$). This task can be carried out automatically by developing appropriate formal calculus tools; the simulations shown in section 5 were obtained in this way.

4.2. Compatibility conditions. Some components of the spatial derivatives of $\boldsymbol{U}$ are not independent. We set

$$\alpha_1 = \frac{\lambda + 2\,\mu}{4\,(\lambda+\mu)} = \frac{c_p^2}{4\left(c_p^2 - c_s^2\right)}, \qquad \alpha_2 = \frac{-\lambda}{4\,(\lambda+\mu)} = \frac{2\,c_s^2 - c_p^2}{4\left(c_p^2 - c_s^2\right)}, \tag{4.8}$$

where λ and μ are the Lamé coefficients. Then, the standard plane elasticity [illegible] of Saint-Venant [6, 13] are, in terms of stresses,

$$\alpha_2\,\frac{\partial^2\,\sigma_{11}}{\partial\,x^2} + \alpha_1\,\frac{\partial^2\,\sigma_{22}}{\partial\,x^2} - \frac{\partial^2\,\sigma_{12}}{\partial\,x\,\partial\,y} + \alpha_1\,\frac{\partial^2\,\sigma_{11}}{\partial\,y^2} + \alpha_2\,\frac{\partial^2\,\sigma_{22}}{\partial\,y^2} = 0. \tag{4.9}$$

Equation (4.9) is differentiated $(k-2)$-times in terms of x and y, giving the $n_m = k(k-1)/2$ relations

$$\begin{aligned} &\alpha_2\,\frac{\partial^k\,\sigma_{11}}{\partial\,x^{k-j}\,\partial\,y^j} + \alpha_1\,\frac{\partial^k\,\sigma_{22}}{\partial\,x^{k-j}\,\partial\,y^j} - \frac{\partial^k\,\sigma_{12}}{\partial\,x^{k-j-1}\,\partial\,y^{j+1}} \\ &+ \alpha_1\,\frac{\partial^k\,\sigma_{11}}{\partial\,x^{k-j-2}\,\partial\,y^{j+2}} + \alpha_2\,\frac{\partial^k\,\sigma_{22}}{\partial\,x^{k-j-2}\,\partial\,y^{j+2}} = 0, \quad k \geq 2,\ j = 0, \ldots, k-2. \end{aligned} \tag{4.10}$$

Conditions (4.10) are satisfied at each point in Ω_l $(l = 0, 1)$, especially at P (see Figure 2.1). One can therefore express the limit values $\boldsymbol{U}_l^k$ in terms of a vector $\boldsymbol{V}_l^k$ with $n_v - n_m$ independent components

$$\boxed{\boldsymbol{U}_l^k = \boldsymbol{G}_l^k\,\boldsymbol{V}_l^k, \quad l = 0, 1,} \tag{4.11}$$

where $\boldsymbol{G}_l^k$ is a $n_v \times (n_v - n_m)$ matrix deduced from (4.10). The relation (4.11) is useful to reduce the number of components in (4.7), as seen in the next subsection. An algorithm has been proposed in [12] for computing the nonnull components of $\boldsymbol{G}_l^k$

but there was a mistake for $k \geq 3$. The correct algorithm is ($l = 0,\ 1$)

$$
\begin{array}{l}
\alpha = 0, \quad \beta = 0, \\
\text{for } \gamma = 0, \ldots, k, \text{ for } \delta = 0, \ldots, \gamma \\
\quad \text{if } \delta = 0 \text{ then for } \varepsilon = 1, \ldots, 5 \\
\quad\quad \alpha = \alpha + 1, \quad \beta = \beta + 1, \quad \boldsymbol{G}_l^k[\alpha, \beta] = 1 \\
\quad \text{if } \gamma \neq 0 \text{ and } \delta \neq 0 \text{ and } \gamma \neq \delta \text{ then} \\
\quad\quad \text{if } \gamma = 2 \text{ then } \nu = 0,\ \eta = 0, \\
\quad\quad \text{else if } \delta = 1 \text{ then } \nu = 0,\ \eta = 1, \\
\quad\quad \text{else if } \delta = \gamma - 1 \text{ then } \nu = 1,\ \eta = 0, \\
\quad\quad \text{else } \nu = 1,\ \eta = 1, \\
\quad\quad \alpha = \alpha + 1, \quad \beta = \beta + 1, \qquad \boldsymbol{G}_l^k[\alpha, \beta] = 1 \\
\quad\quad \alpha = \alpha + 1, \quad \beta = \beta + 1, \qquad \boldsymbol{G}_l^k[\alpha, \beta] = 1 \\
\quad\quad \alpha = \alpha + 1, \quad \beta = \beta + 1, \qquad \boldsymbol{G}_l^k[\alpha, \beta] = 1 \\
\quad\quad \alpha = \alpha + 1, \quad \beta = \beta - 5 + \nu, \quad \boldsymbol{G}_l^k[\alpha, \beta] = \alpha_2 \\
\quad\quad\quad\quad\quad\ \beta = \beta + 2 - \nu, \quad \boldsymbol{G}_l^k[\alpha, \beta] = \alpha_1 \\
\quad\quad\quad\quad\quad\ \beta = \beta + 7, \qquad \boldsymbol{G}_l^k[\alpha, \beta] = \alpha_1 \\
\quad\quad\quad\quad\quad\ \beta = \beta + 2 - \eta, \quad \boldsymbol{G}_l^k[\alpha, \beta] = \alpha_2 \\
\quad\quad \alpha = \alpha + 1, \quad \beta = \beta - 5 + \eta, \quad \boldsymbol{G}_l^k[\alpha, \beta] = 1 \\
\quad \text{if } \gamma \neq 0 \text{ and } \gamma = \delta \text{ then for } \varepsilon = 1, \ldots, 5 \\
\quad\quad \alpha = \alpha + 1, \quad \beta = \beta + 1, \quad \boldsymbol{G}_l^k[\alpha, \beta] = 1.
\end{array}
\tag{4.12}
$$

4.3. Solution of underdetermined systems. It is now required to express $\boldsymbol{V}_1^k$ in terms of $\boldsymbol{V}_0^k$. For this purpose, we inject the compatibility conditions (4.11) into the jump conditions (4.7). Setting

$$
\boldsymbol{S}_1^k = \boldsymbol{C}_1^k \, \boldsymbol{G}_1^k, \qquad \boldsymbol{S}_0^k = \left(\boldsymbol{C}_0^k + \boldsymbol{D}_0^k \right) \boldsymbol{G}_0^k,
\tag{4.13}
$$

we obtain the underdetermined system

$$
\boldsymbol{S}_1^k \, \boldsymbol{V}_1^k = \boldsymbol{S}_0^k \, \boldsymbol{V}_0^k + \boldsymbol{E}_0^k \, \overline{\boldsymbol{U}}_0^{k+1}.
\tag{4.14}
$$

To find the full range of solutions of (4.14), a singular value decomposition (SVD) of $\boldsymbol{S}_1^k$ [16] is computed. Setting $(\boldsymbol{S}_1^k)^{-1}$ to denote the $(n_v - n_m) \times n_c$ generalized inverse of $\boldsymbol{S}_1^k$ and $\boldsymbol{R}_{s_1}^k$ to denote the $(n_v - n_m) \times (n_v - n_m - n_c)$ matrix associated with the kernel of $\boldsymbol{S}_1^k$, we obtain

$$
\boxed{\boldsymbol{V}_1^k = \left(\left(\boldsymbol{S}_1^k \right)^{-1} \boldsymbol{S}_0^k \,|\, \boldsymbol{R}_{s_1}^k \right) \begin{pmatrix} \boldsymbol{V}_0^k \\ \\ \boldsymbol{\Lambda}^k \end{pmatrix} + \left(\boldsymbol{S}_1^k \right)^{-1} \boldsymbol{E}_0^k \, \overline{\boldsymbol{U}}_0^{k+1}},
\tag{4.15}
$$

where $\boldsymbol{\Lambda}_k$ is a $(n_v - n_m - n_c)$ vector of Lagrange multipliers. A similar procedure can be used to express $\boldsymbol{V}_0^k$ in terms of $\boldsymbol{V}_1^k$.

4.4. Numerical estimates of limit fields. Consider a set $\mathcal{B}$ of n_q grid points surrounding P; for practical purposes, we define $\mathcal{B}$ as the set of grid points whose distance from P is less than a given distance q. We successively examine each grid point in $\mathcal{B}$, writing the kth order Taylor expansions of $\boldsymbol{U}(x_i, y_j, t_n)$ at P. If $(x_i,\, y_j)$ is included in Ω_0, we deduce from (3.3) and (4.11) that

$$
\begin{aligned}
(x_i,\, y_j) \in \mathcal{B} \cap \Omega_0, \quad \boldsymbol{U}(x_i, y_j, t_n) &= \boldsymbol{\Pi}_{i,j}^k\, \boldsymbol{U}_0^k + \boldsymbol{O}(\Delta\, x^{k+1}) \\
&= \boldsymbol{\Pi}_{i,j}^k\, \boldsymbol{G}_0^k\, \boldsymbol{V}_0^k + \boldsymbol{O}(\Delta\, x^{k+1}) \\
&= \boldsymbol{\Pi}_{i,j}^k\, \boldsymbol{G}_0^k\, (\mathbf{1} \,|\, \mathbf{0}) \begin{pmatrix} \boldsymbol{V}_0^k \\ \\ \boldsymbol{\Lambda}^k \end{pmatrix} + \boldsymbol{O}(\Delta\, x^{k+1}),
\end{aligned}
\tag{4.16}
$$

where $\mathbf{1}$ is the identity matrix $(n_v - n_m) \times (n_v - n_m)$ and $\mathbf{0}$ is the null matrix $(n_v - n_m) \times (n_v - n_m - n_c)$. If $(x_i,\, y_j)$ is included in Ω_1, we deduce from (3.3), (4.11), and (4.15) that

$$
\begin{aligned}
(x_i,\, y_j) \in \mathcal{B} \cap \Omega_1, \quad \boldsymbol{U}(x_i, y_j, t_n) &= \boldsymbol{\Pi}_{i,j}^k\, \boldsymbol{U}_1^k + \boldsymbol{O}(\Delta\, x^{k+1}) \\
&= \boldsymbol{\Pi}_{i,j}^k\, \boldsymbol{G}_1^k\, \boldsymbol{V}_1^k + \boldsymbol{O}(\Delta\, x^{k+1}) \\
&= \boldsymbol{\Pi}_{i,j}^k\, \boldsymbol{G}_1^k \begin{pmatrix} \left(\boldsymbol{S}_1^k\right)^{-1} \boldsymbol{S}_0^k \,|\, \boldsymbol{R}_{s_1}^k \end{pmatrix} \begin{pmatrix} \boldsymbol{V}_0^k \\ \\ \boldsymbol{\Lambda}^k \end{pmatrix} \\
&\quad + \boldsymbol{\Pi}_{i,j}^k\, \boldsymbol{G}_1^k \left(\boldsymbol{S}_1^k\right)^{-1} \boldsymbol{E}_0^k\, \overline{\boldsymbol{U}}_0^{k+1} + \boldsymbol{O}(\Delta\, x^{k+1}).
\end{aligned}
\tag{4.17}
$$

Relations (4.16) and (4.17) are summed up in matrix terms as

$$
(\boldsymbol{\mathcal{U}}^n)_{\mathcal{B}} = \boldsymbol{M} \begin{pmatrix} \boldsymbol{V}_0^k \\ \\ \boldsymbol{\Lambda}^k \end{pmatrix} + \boldsymbol{N}\, \overline{\boldsymbol{U}}_0^{k+1} + \begin{pmatrix} \boldsymbol{O}(\Delta\, x^{k+1}) \\ \vdots \\ \boldsymbol{O}(\Delta\, x^{k+1}) \end{pmatrix},
\tag{4.18}
$$

where $(\boldsymbol{\mathcal{U}}^n)_{\mathcal{B}}$ refers to the set of exact values $\boldsymbol{U}(x_i, y_j, t_n)$ at the grid points of $\mathcal{B}$, $\boldsymbol{M}$ is a $5\, n_q \times (2\, n_v - 2\, n_m - n_c)$ matrix, and $\boldsymbol{N}$ is a $5\, n_q \times 5(k+2)$ matrix. The value of n_q depends on $\mathcal{B}$; this set is chosen so that (4.18) is overdetermined. In view of Table 3.1 and Table 4.1, this means that

$$
\begin{aligned}
n_q &\geq (2\, n_v - 2\, n_m - n_c)/5 \\
&\geq 2\left(k^2 + 5\, k + 3\right)/5.
\end{aligned}
$$

To ensure this inequality,

$$
q = (k + 0.2)\, \Delta\, x
\tag{4.19}
$$

can be used as a radius to obtain $\mathcal{B}$. From now on, numerical values and exact values are used indiscriminately in (4.18). We also remove $\boldsymbol{N}\, \overline{\boldsymbol{U}}_0^{k+1}$ and the remainder terms. The least-squares inverse $\boldsymbol{M}^{-1}$ of $\boldsymbol{M}$ is computed using classical techniques, such as

TABLE 4.1
Matrices used for computing the modified values ($l = 0, 1$).

Matrix	Size	Comment
$\boldsymbol{C}_l^k$	$n_c \times n_v$	jump conditions: perfect contact (4.7)
$\boldsymbol{D}_l^k$	$n_c \times n_v$	jump conditions: imperfect contact (4.7)
$\boldsymbol{G}_l^k$	$n_v \times (n_v - n_m)$	compatibility conditions (4.11)
$\boldsymbol{S}_l^k$	$n_c \times (n_v - n_m)$	(4.13)
$\boldsymbol{R}_{s_l}^k$	$(n_v - n_m) \times (n_v - n_c - n_m)$	kernel of $\boldsymbol{S}_l^k$ (4.15)
$\boldsymbol{\Pi}_{i,j}^k$	$5 \times n_v$	Taylor expansions (3.3)
$\mathbf{1}$	$(n_v - n_m) \times (n_v - n_m)$	identity (4.16)
$\mathbf{0}$	$(n_v - n_m) \times (n_v - n_c - n_m)$	null (4.16)
$\boldsymbol{M}$	$5\,n_q \times (2\,n_v - 2\,n_v - n_c)$	(4.18)
$\overline{\boldsymbol{M}^{-1}}$	$(n_v - n_q) \times 5\,n_q$	restriction of $\boldsymbol{M}^{-1}$ (4.21)

LU or SVD [16]. The set of Lagrange multipliers $\boldsymbol{\Lambda}^k$ is not useful for computing modified values; the restriction $\overline{\boldsymbol{M}^{-1}}$ of $\boldsymbol{M}^{-1}$ is therefore defined by keeping only the $(n_v - n_m)$ first lines of $\boldsymbol{M}^{-1}$. This gives the least-squares numerical estimate

$$\boldsymbol{V}_0^k = \overline{\boldsymbol{M}^{-1}}\,(\boldsymbol{\mathcal{U}}^n)_{\mathcal{B}}\,. \tag{4.20}$$

4.5. Modified value. We now have all the tools required to be able to compute the modified value at $M(x_I, y_J)$, as defined in section 3.2. The modified value at (x_I, y_J) is deduced from (3.4), (4.11), and (4.20),

$$\boxed{\boldsymbol{U}_{I,J}^* = \boldsymbol{\Pi}_{I,J}^k\,\boldsymbol{G}_0^k\,\overline{\boldsymbol{M}^{-1}}\,(\boldsymbol{\mathcal{U}}^n)_{\mathcal{B}}\,.} \tag{4.21}$$

A similar procedure is applied at each irregular point surrounding Γ. All the matrices required to compute the modified values are recalled in Table 4.1 with their sizes.

4.6. Comments about the algorithm. This algorithm can easily be adapted to existing codes without having to change the scheme. At each time step, one only needs to compute a few "modified values" (4.21), which do not depend on the scheme selected.

Since the interface is stationary and the nature of the contacts does not vary with time, most of the algorithm can be built up during a preprocessing step, consisting in determining irregular points, deriving interface conditions (4.7), solving underdetermined systems (4.15), and computing the matrices involved in (4.21). All these quantities are stored for future use. At each time step, we need only to carry out one matrix-vector multiplication (4.21) at each irregular point, before running time-stepping procedure. The matrices involved here are small: $5 \times 5\,n_q$ components, usually with $n_q \approx 20$. In addition, the number of irregular points is much smaller than the number of grid points. The extra computational cost is therefore very low. The CPU time measurements are likely to be approximately the same as those presented in [12].

The main aim of the interface method is to incorporate the jump conditions into finite-difference schemes. But it also gives a finer resolution of the geometries than the poor stair-step description induced by the Cartesian grid. The differentiations of spring-mass conditions involve successive derivatives of $x^{'}$ and $y^{'}$ (see section 4.1). In addition, the Taylor expansions in (4.16), (4.17), and (4.21) give information about the position of P inside the mesh.

In section 3.1, we stated that the interface method can be adapted to staggered grid schemes. This adaptation requires the algebra in sections 3 and 4 to be completely rewritten (although this is quite straightforward). To give an idea of the changes involved, let us focus on the staggered scheme [19] with a grid $(x_i,\, y_j,\, t_n)$ for stresses and a grid $(x_{i+\frac{1}{2}},\, y_{j+\frac{1}{2}},\, t_{n+\frac{1}{2}})$ for velocities. Two different sets of irregular points now need to be stored. The jump conditions for velocities and stresses are not correlated (see (4.3) and Appendix B); two independant systems of jump conditions (4.7) are therefore written. Since the jump conditions satisfied by the velocities yield as many equations as unknowns, they do not require an SVD. The compatibility conditions (4.10) are applied only to the jump conditions satisfied by the stresses, that constitute an underdetermined system. The modified values calculated in sections 4.4 and 4.5 are also split in two parts: at time $t_{n+\frac{1}{2}}$, the modified values of the stresses are computed (based on numerical values of $\boldsymbol{\sigma}$ at t_n) before being inserted into the time-marching procedure performed on the velocities. In the same way, at time t_{n+1}, the modified values of the velocities are computed (based on numerical values of $\boldsymbol{v}$ at $t_{n+\frac{1}{2}}$) before they are used to perform the time-marching procedure on the stresses.

4.7. Open questions. Four questions relating to numerical analysis remain to be solved. First, what are the effects of the term $\boldsymbol{N}\,\overline{\boldsymbol{U}}_0^{k+1}$ which was neglected in (4.18)? In one dimension [11], we established that this term is of the same order as the remainder terms. The proof in one dimension was based on performing explicit calculations on MAPLE; this approach seems difficult to apply in the two-dimensional context, because it depends on the geometry of Γ near P.

Second, what is the local truncation error at the irregular points obtained by combining the interface method with the scheme? In the one-dimensional context [11, 15], we established that the following was true: if r is the order of the scheme, the local truncation error at irregular points is still equal to r if $k \geq r$ (which means $k \geq 2$ in the case of second-order schemes such as WPALG), when this scheme is combined with the interface method. Extending this result to the two-dimensional context would require answering the previous question about $\boldsymbol{N}\,\overline{\boldsymbol{U}}_0^{k+1}$, and bounding the matrix $\overline{\boldsymbol{M}^{-1}}$. In practice, the convergence measurements indicate that this property is false in the two-dimensional context (at least for $r = 2$): we need $k = 3$ to ensure second-order accuracy (see section 5.2).

Third, what happens in the extreme case of a homogeneous medium (that is, $\rho_0 = \rho_1$, $c_{p0} = c_{p1}$, $c_{s0} = c_{s1}$, $K_N \to +\infty$, $K_T \to +\infty$, $M_N = 0$, and $M_T = 0$)? The ideal would obviously be that $\boldsymbol{U}_{i,j}^{*} = \boldsymbol{U}_{i,j}^{n}$, in order to recover the scheme in homogeneous medium. In one-dimensional configurations [11], we have shown that it is the case, given simple conditions about the set $\mathcal{B}$ used to estimate the modified values. In a two-dimensional context, these conditions cannot be satisfied because estimating the modified values requires least-squares computations. However, it would be interesting to estimate the difference between $\boldsymbol{U}_{i,j}^{*}$ and $\boldsymbol{U}_{i,j}^{n}$, which might lead to an optimal choice of $\mathcal{B}$.

Fourth and last, the stability analysis still remains to be performed. With a nonlinear scheme (such as WPALG), this seems out of reach so far, but an analysis could

perhaps be carried out with linear schemes (such as the Lax–Wendroff scheme). In the absence of theoretical rules, we have numerically considered many geometrical configurations, physical parameters values, stiffness, and mass values. With a wide range of parameters, no instabilities are usually observed up to the CFL limit, even after very long integration times (a few thousand time-steps). However, instabilities are observed in two cases. First, instabilities can increase when the physical parameters differ considerably between the two sides of an interface. This problem was previously mentioned in [12] in connection with perfect contacts. In practice, it is not too penalizing when dealing with realistic media: even quite large differences between the impedance values, such as those encountered in the case of Plexiglas-aluminium interfaces, yield stable computations. Second, the instabilities can increase at low values of $K_{N,T}$, when the two media tend to be disconnected. This may be due to the small denominators in (2.10). In this case, increasing the order k provides an efficient means of improving the stability limit (see subsection 5.4).

5. Numerical experiments.

5.1. Configurations. Five numerical experiments are performed here. Except for Test 3, the physical parameters are the same on both sides of Γ: this enables us to underscore both the effects of the spring-mass conditions and the accuracy of the interface method, since all wave reflections and changes observed will result from the spring-mass conditions, described numerically by the interface method.

Three geometrical configurations are studied: a plane interface, a circular interface, and a noncanonical object described by cubic splines. Analytical solutions can be obtained for the first two configurations when the spring-mass conditions are constant. The analytical solution of the problem with a plane wave impinging on a plane interface with spring-mass conditions can be quite easily calculated using Fourier analysis; this procedure will therefore not be described here. We have not found articles dealing with the more intricate case of a circular interface; since this analytical solution is useful to validate the algorithm, it has been described in Appendix C. Except for the convergence measurements in Test 1, all the computations were performed with the WPALG.

Apart from Test 5, the computations are all initialized by a plane right-going P-wave

$$\boldsymbol{U}(x,y,t_0) = \varepsilon \begin{pmatrix} -\dfrac{\cos\theta}{c_{p1}} \\ -\dfrac{\sin\theta}{c_{p1}} \\ \dfrac{\lambda_1 + 2\,\mu_1\,\cos^2\theta}{c_{p1}^2} \\ \dfrac{2\,\mu_1\,\sin\theta\,\cos\theta}{c_{p1}^2} \\ \dfrac{\lambda_1 + 2\,\mu_1\,\sin^2\theta}{c_{p1}^2} \end{pmatrix} f\left(t_0 - \frac{x\cos\theta + y\sin\theta}{c_{p1}}\right), \tag{5.1}$$

where ε is an amplitude factor, $\mu_1 = \rho_1\, c_{s1}^2$ and $\lambda_1 = \rho_1(c_{p1}^2 - 2\,c_{s1}^2)$ are the Lamé coefficients, θ is the angle between the direction of propagation and the x-axis, and

t_0 is the initial instant. Function f is a C^2 spatially bounded sinusoid

$$f(\xi) = \begin{cases} \sin(\omega_c\,\xi) - \dfrac{1}{2}\,\sin(2\,\omega_c\,\xi) & \text{if } 0 < \xi < \dfrac{1}{f_c}, \\ 0 \text{ else,} \end{cases} \tag{5.2}$$

where f_c is the central frequency, and $\omega_c = 2\,\pi\,f_c$.

We will conclude this section by making some comments on the figures. σ_{11} is shown with a green-red palette for P-waves and a magenta-yellow palette for SV-waves. (In the electronic version of this article, the plates are color.) The distinction between these waves depends on numerical estimates of div $\boldsymbol{v}$ and **curl** $\boldsymbol{v}$. Most of the snapshots shown are accompanied by a slice; the position of a slice is denoted by a horizontal line on the corresponding snapshot. On each slice, the exact solution and the numerical solution are indicated by a solid line and by points, respectively.

5.2. Test 1: Plane interface between identical media. Here we take the case of a $L_x \times L_y = 400 \times 400$ m^2 domain with a plane inclined interface Γ. The points ($x = 168$ m, $y = 17$ m) and ($x = 241$ m, $y = 253$ m) belong to Γ, and hence the angle between Γ and the x-axis is roughly equal to 72.8 degrees. The physical parameters are identical on both sides of Γ

$$\begin{cases} \rho_0 = \rho_1 = 1200 \text{ kg/m}^3, \\ c_{p0} = c_{p1} = 2800 \text{ m/s}, \\ c_{s0} = c_{s1} = 1400 \text{ m/s}, \end{cases}$$

which corresponds to realistic values in the case of Plexiglas. The spring-mass conditions are constant along Γ; the stiffness and mass values are

$$\begin{cases} K_N = 10^9 \text{ kg/s}^2, \quad K_T = 10^7 \text{ kg/s}^2, \\ M_N = 2000 \text{ kg/m}^2, \quad M_T = 1000 \text{ kg/m}^2. \end{cases}$$

The incident P-wave (5.1) is defined by $\theta = 40$ degrees, $f_c = 49.9$ Hz, $t_0 = 0.1$ s, and $\varepsilon = 2\,10^{-3}$. The numerical experiments are performed with $N_x \times N_y = 400 \times 400$ grid points, which amounts to 56 and 28 grid points per central wavelength for P-waves and SV-waves, respectively. We take CFL $= 0.9$, and $k = 2$ for the extrapolations involved in the interface method. To initialize the computation, one also needs to compute reflected and transmitted P- and SV-waves, via discrete Fourier transforms (DFT). To obtain the required level of accuracy, we perform the DFTs on 32768 points, with a sampling frequency of 0.018125 Hz. At each time step, the exact solution is imposed at the boundaries of the computational domain.

Figure 5.1 shows the solution at time steps $t_i = t_0 + 75\,i\,\Delta\,t$ $(i = 0, 1, 2)$. The slices are performed from

- $i = 0$: ($x = 70$ m, $y = 26$ m) to ($x = 360$ m, $y = 26$ m),
- $i = 1$: ($x = 156$ m, $y = 162$ m) to ($x = 207$ m, $y = 162$ m),
- $i = 2$: ($x = 246$ m, $y = 198$ m) to ($x = 282$ m, $y = 198$ m).

Looking from the left to the right on the first slice ($i = 0$), one can successively observe the reflected SV-wave, the transmitted SV-wave, and the transmitted P-wave. As mentioned in section 2.3, these waves do not have the same sinusoidal profile as the incident P-wave. A "coda" occurs after the reflected and transmitted P and SV-waves: as in [11], these fields are not spatially bounded in the direction of the propagation.

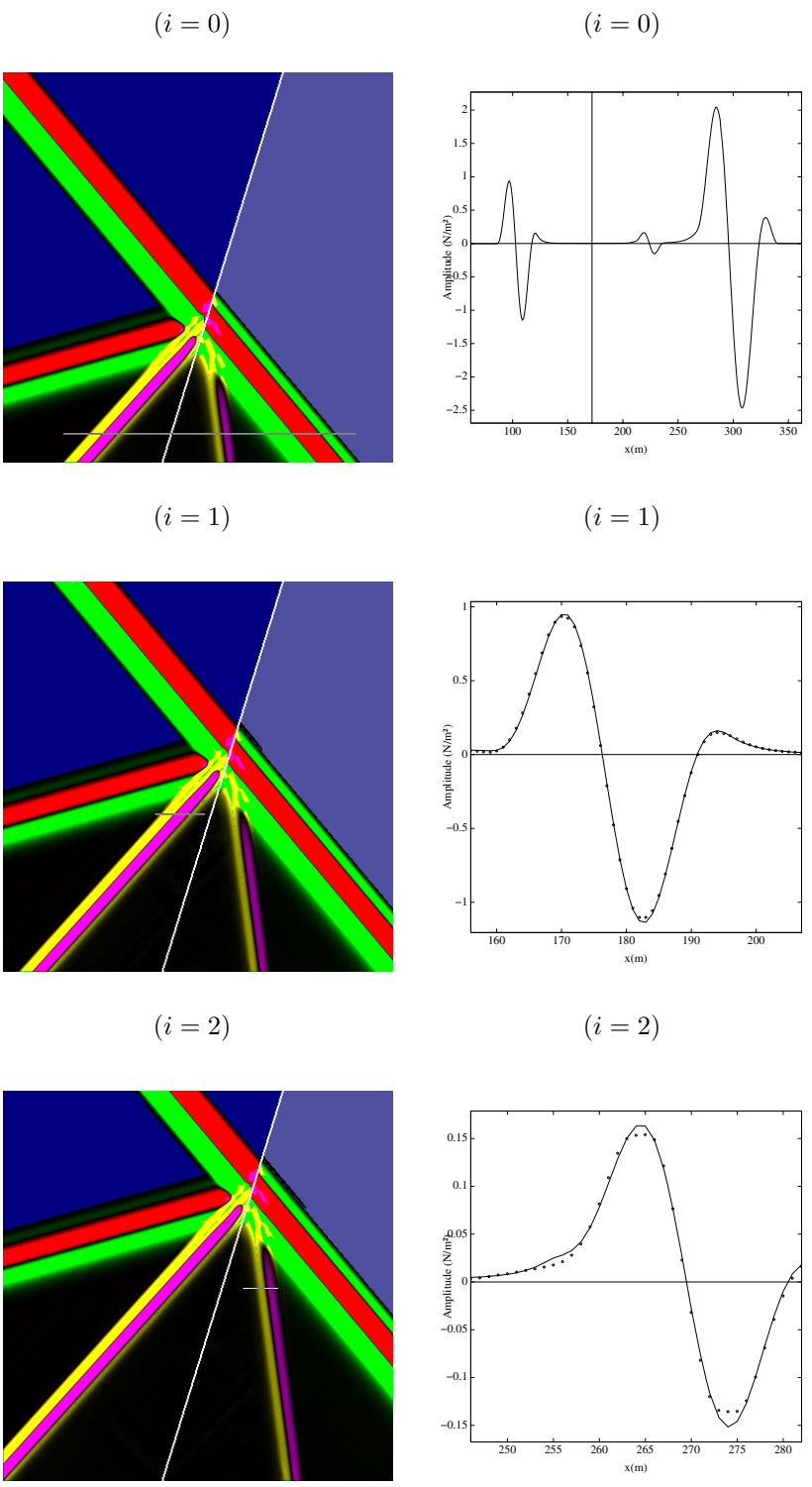


FIG. 5.1. *Test* 1: *plane interface between identical media at* $t = t_0 + 75\, i\, \Delta t$.

TABLE 5.1
Convergence measurements in Test 1.

Scheme	N_x	L_∞ error	L_∞ order	L_1 error	L_1 order
Lax–Wendroff	50	6.265e-1	-	4.676e3	-
	100	1.760e-1	**1.832**	1.336e3	**1.807**
+	200	4.755e-2	**1.888**	3.718e2	**1.845**
	400	1.228e-2	**1.953**	9.437e1	**1.978**
ESIM	800	3.164e-3	**1.956**	2.311e1	**2.030**
	1600	7.916e-4	**1.999**	5.667e0	**2.028**
WPALG	50	4.570e-1	-	6.259e2	-
	100	1.472e-1	**1.634**	1.504e2	**2.058**
+	200	4.260e-2	**1.789**	3.954e1	**1.926**
	400	1.398e-2	**1.607**	1.071e1	**1.884**
ESIM	800	4.464e-3	**1.647**	2.833e0	**1.919**
	1600	1.503e-3	**1.570**	7.369e-1	**1.943**

The second slice ($i = 1$) crosses the reflected SV-wave. The agreement between exact and numerical values is good; one only observes a small amount of numerical diffusion induced by WPALG. This is also so in the case of the last slice ($i = 2$) across the transmitted SV-wave.

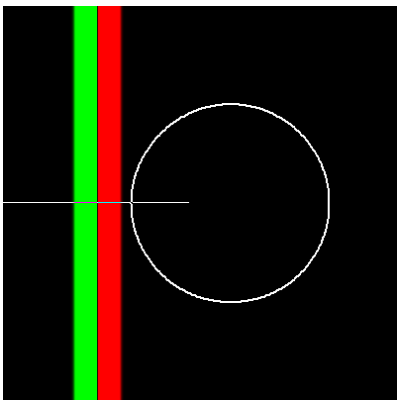
In Table 5.1, we show convergence measurements performed with the Lax–Wendroff scheme and with WPALG, by taking $k = 3$. This table was obtained by taking successively refined meshes with a constant CFL and measuring the differences between the analytical and numerical values. The orders of accuracy (2 for Lax–Wendroff, 1.5 in norm L_∞, and 2 in norm L_1 for WPALG) are the same as in homogeneous medium. Similar results can be obtained by increasing k. However, for $k = 2$, the convergence orders fall below 1.4. (We recall that $k = 2$ sufficies to ensure second-order accuracy for perfect contacts in two-dimensional contexts [12] and for imperfect contacts in one-dimensional contexts [11].)

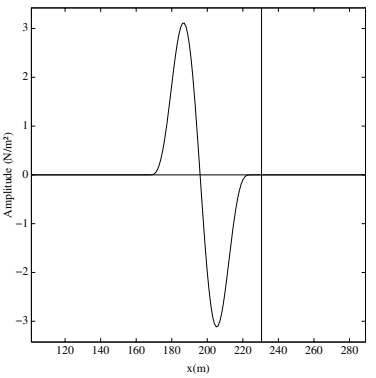
5.3. Test 2: Circular interface between identical media. As a second example, we take the case of a $L_x \times L_y = 600 \times 600$ m^2 domain and a circular interface (radius $a = 100$ m, centered at $x_0 = 330.5$ m, $y_0 = 300$ m). The numerical experiments are performed on $N_x \times N_y = 600 \times 600$ grid points. The physical parameters are the same as in the previous example. The spring-mass conditions are

$$\begin{cases} K_N = K_T = 10^9 \ \text{kg/s}^2, \\ M_N = M_T = 1000 \ \text{kg/m}^2. \end{cases}$$

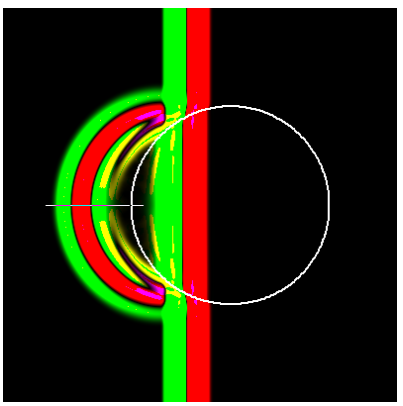
Figures 5.2 and 5.3 show the solution at the initial instant $t_0 = 0.08$ s, and at $t_i = t_0 + 100\, i\, \Delta t$ ($i = 1, \ldots, 5$) on a restricted domain $[100, 500] \times [100, 500]$ m^2 centered on the middle of the computational domain. No special treatment is applied to simulate the wave propagation in infinite medium (such as absorbing boundary conditions or perfectly matched layers), but the integration times are sufficiently short to prevent spurious waves reflected by the edges of the computational region from appearing

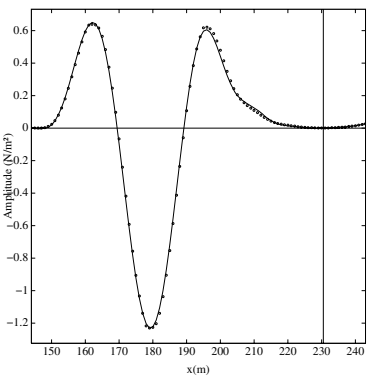
$(i = 0)$ $(i = 0)$



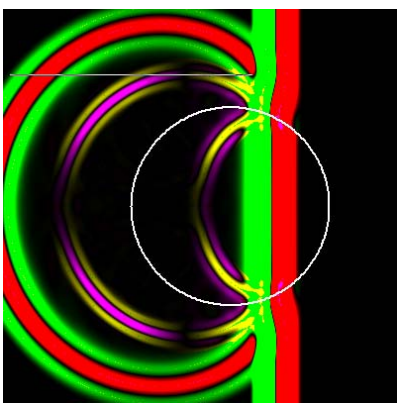


$(i = 1)$ $(i = 1)$





$(i = 2)$ $(i = 2)$



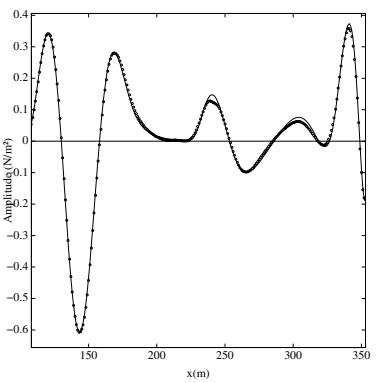
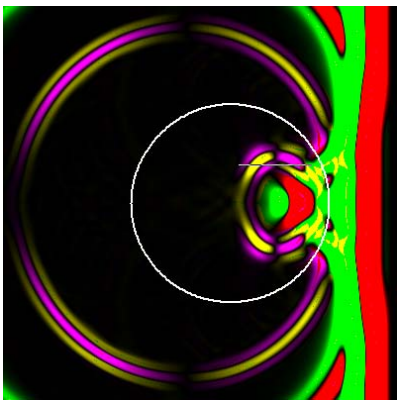
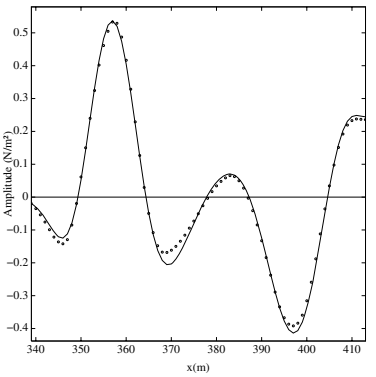


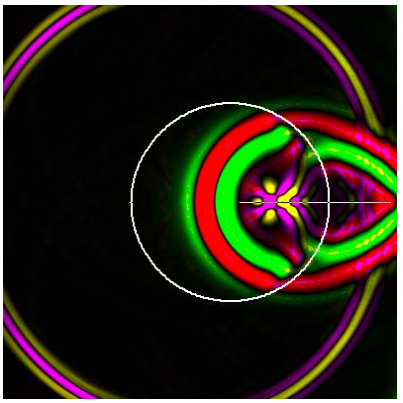
FIG. 5.2. *Test* 2-*a: cylindar between identical media,* $t = t_0 + 100\, i\, \Delta\, t$.

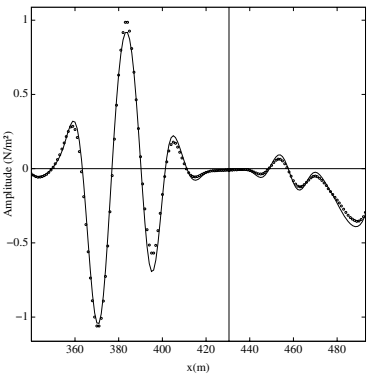
$(i = 3)$ $(i = 3)$



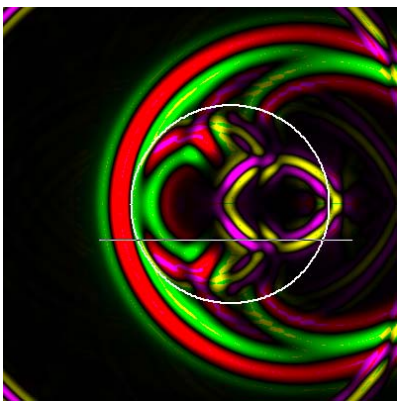


$(i = 4)$ $(i = 4)$





$(i = 5)$ $(i = 5)$



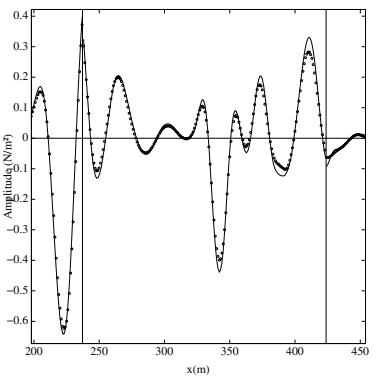


FIG. 5.3. *Test* 2-*b: cylindar between identical media at* $t = t_0 + 100\, i\, \Delta\, t$.

in the restricted domain (the same comment holds for the further figures). The slices are carried out from

- $i = 0$: ($x = 101$ m, $y = 301$ m) to ($x = 289$ m, $y = 301$ m),
- $i = 1$: ($x = 144$ m, $y = 300$ m) to ($x = 243$ m, $y = 300$ m),
- $i = 2$: ($x = 108$ m, $y = 433$ m) to ($x = 353$ m, $y = 433$ m),
- $i = 3$: ($x = 339$ m, $y = 339$ m) to ($x = 413$ m, $y = 339$ m),
- $i = 4$: ($x = 340$ m, $y = 300$ m) to ($x = 493$ m, $y = 300$ m),
- $i = 5$: ($x = 198$ m, $y = 264$ m) to ($x = 454$ m, $y = 264$ m).

The analytical solutions are computed on $N_{Bessel} = 90$ points (see (C.16) in Appendix C), 256 Fourier points, with 1-Hz sampling frequency.

The incident P-wave is converted into transmitted and reflected P- and SV-waves ($i = 1, 2$). A series of refraction-conversion events is then observed ($i = 3, 4, 5$). The agreement between the numerical and analytical values is excellent.

5.4. Test 3: Circular interface between different media. As a third example, we consider the same circular interface as in the previous test but take it to have different physical parameters on both each sides of Γ,

$$(\rho,\, c_p,\, c_s) = \begin{cases} \rho_0 = 2600 \text{ kg/m}^3,\, c_{p0} = 6400 \text{ m/s},\, c_{s0} = 3200 \text{ m/s}, \\ \rho_1 = 1200 \text{ kg/m}^3,\, c_{p1} = 2800 \text{ m/s},\, c_{s1} = 1400 \text{ m/s}. \end{cases}$$

These parameters are those of aluminium (inside Γ) and Plexiglas (outside Γ). Apart from these physical parameters, the values used for the computations are the same as in section 5.3. The initial values are the same as in Figure 5.2, $i = 0$.

Figure 5.4 gives the results of a parametric study in terms of the normal stiffness K_N: perfect contact ($i = 0$) which amounts to $K_N \to +\infty$, $K_N = 10^9$ kg/s^2 ($i = 1$), and $K_N = 10^8$ kg/s^2 ($i = 2$). The inertial effects and the tangential stiffness are not taken into account here,

$$\begin{cases} K_T \to +\infty, \\ M_N = M_T = 0 \text{ kg/m}^2. \end{cases}$$

The computations are shown in Figure 5.4 after 350 time steps. The slices are performed from $x = 100$ m to $x = 500$ m, at $y = 300$ m. Compared with the case of perfect contact ($i = 0$), one can see in ($i = 1, 2$) that the signs of the reflected waves are reversed. The waves that have entirely crossed the interface also differ greatly among the three cases. The case ($i = 2$) is much more difficult to handle than the cases ($i = 0, 1$) from the computational point of view. To obtain the good agreement shown in Figure 5.4 ($i = 2$), it is necessary to use $k = 4$ (for the numerical solution) and 32768 Fourier points with a sampling frequency of 0.018125 Hz (for the analytical solution).

Smaller values of K_N have also been investigated (figures not shown here). Below $K_N = 10^7$ kg/s^2, numerical instabilities have been observed when $k = 2$ (see subsection 4.7), whereas the computations remain stable when $k \geq 3$. Instabilities increase below $K_N = 10^5$ kg/s^2 if $k = 3$. Stable computations and excellent agreement with the analytical solutions are obtained down to $K_N = 10^4$ kg/s^2 if $k = 4$.

5.5. Test 4: Cubic spline between identical media. Up to now, we have dealt only with canonical objects with constant curvature. However, the use of the

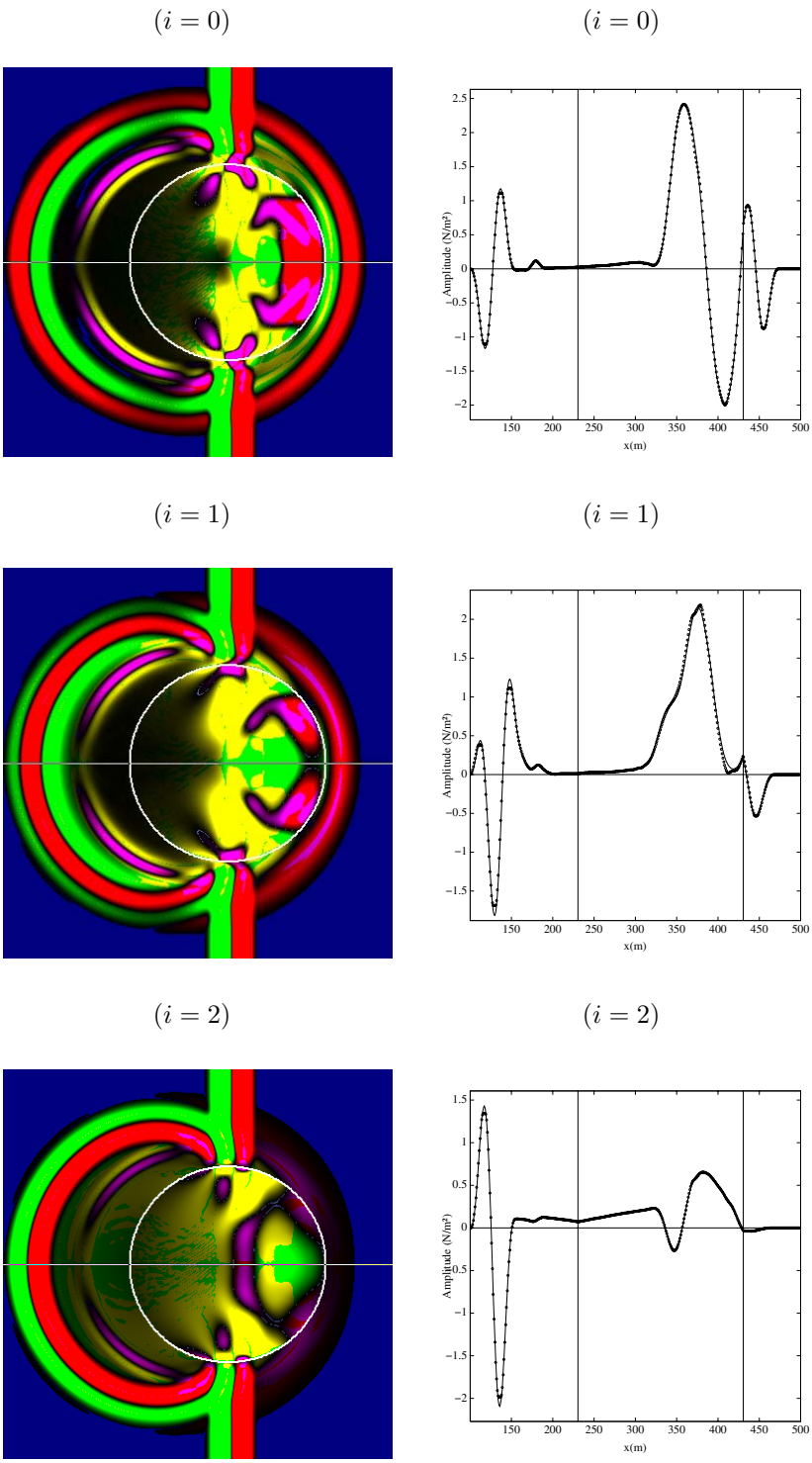


FIG. 5.4. *Test* 3: *cylindar between different media at* $t = t_0 + 360\,\Delta\,t$, *with various values of normal stifness.*

interface method is not restricted to such simple geometries. In this section, we study a more complex configuration corresponding to a cubic spline. The other parameters are the same as in section 5.3, except $t_0 = 0.07$ s.

Figure 5.5 shows snapshots of the solutions at times $t_i = t_0 + 150\, i\, \Delta t$ ($i = 0, 1, 2$). We obviously have no analytical solutions to compare with the numerical values. However, the computations remain stable, and the wave phenomena seem to be realistic.

5.6. Test 5: Plane interface with variable jump conditions. In the last experiment, we study the case of a plane interface with variable spring-mass conditions. The interface Γ is placed horizontally at $y = 130$ m on a $L_x \times L_y = 600 \times 400$ m^2 domain. The physical parameters have the same values as in section 5.2, and they are identical on both sides of Γ. The variable values of normal and tangential stiffness are

$$K_N = K_T = \begin{cases} 10^{12} \ \text{kg/s}^2 \ \text{ on } [x = 0 \text{ m} \ , \ x = 270 \text{ m}], \\ g(\tau) \ \text{ on } [x = 270 \text{ m} \ , \ x = 330 \text{ m}], \\ 10^{7} \ \text{kg/s}^2 \ \text{ on } [x = 330 \text{ m} \ , \ x = 600 \text{ m}], \end{cases} \tag{5.3}$$

where $g(\tau)$ is a cubic polynomial that ensures a C^1 continuity (τ is the abscissa along Γ). The stiffness values (5.3) are those of almost perfectly bonded media on the left part of the interface and almost perfectly disconnected media on the right part of the interface. The inertial effects are not taken into account: $M_N = M_T = 0$ kg/m^2.

The computations are initialized by a transient explosive P-wave centered at ($x_s = 300$ m, $y_s = 200$ m), with an elastic potential [14],

$$\Phi_{IP}(x,\, y,\, t) = \left(\tilde{\Phi}_{IP} * f\right)(x,\, y,\, t),$$

$$\tilde{\Phi}_{IP}(r,\, t) = \frac{1}{2\,\pi} \frac{H\left(t - \frac{r}{c_{p1}}\right)}{\sqrt{t^2 - \left(\frac{r}{c_{p1}}\right)^2}}, \qquad r = \sqrt{(x - x_s)^2 + (y - y_s)^2}, \tag{5.4}$$

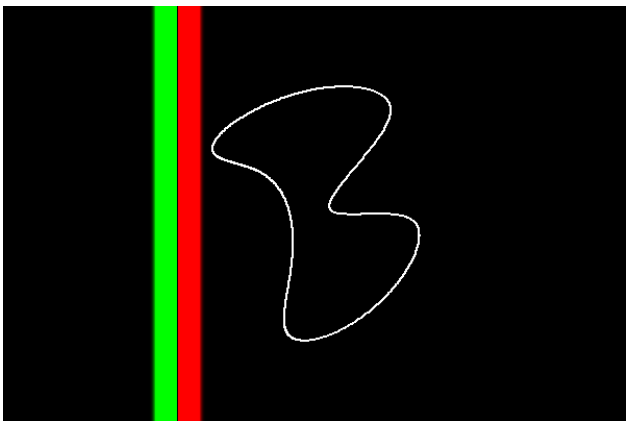
where H is the Heaviside function, $*$ denotes the time convolution, and f is the Ricker wavelet

$$f(t) = \frac{1}{2\,\pi^2\, f_c^2}\, e^{-\left(\pi\, f_c\, (t - t_c)\right)^2}. \tag{5.5}$$

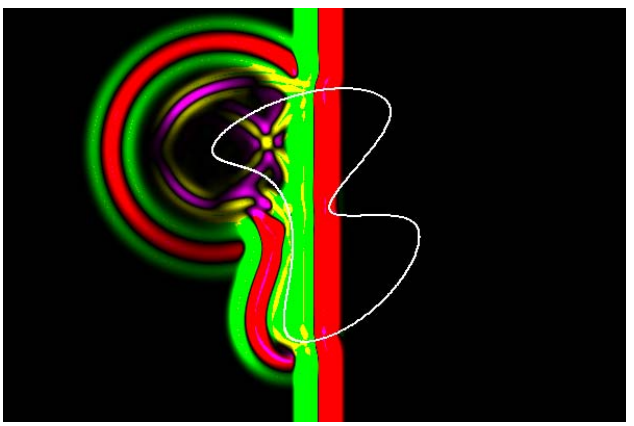
The velocities and stresses of the incident wave can be deduced from (5.4) thanks to classical elastodynamics relations [1]. The numerical experiments are performed on $N_x \times N_y = 600 \times 400$ grid points, with $f_c = 70$ Hz, $t_0 = 0.03$ s, $t_c = 0.01429$ s, CFL $= 0.9$, and $k = 3$.

Figure 5.6 shows snapshots of the numerical solution at $t = t_0 + 150\, i\, \Delta t$. The palettes are changed in each snapshot to maintain a constant range of colours despite the geometrical spreading of the waves. On the left part of the second snapshot ($i = 1$), the incident wave is totally transmitted; on the right part, the wave is almost totally reflected. The rapid change in the stiffness values occuring near $x = 300$ m acts as a point source for cylindrical P- and SV-waves. On the last snapshot, one can also observe a rightward-moving surface wave, denoted by a yellow-magenta zone.

$(i = 0)$



$(i = 1)$



$(i = 2)$

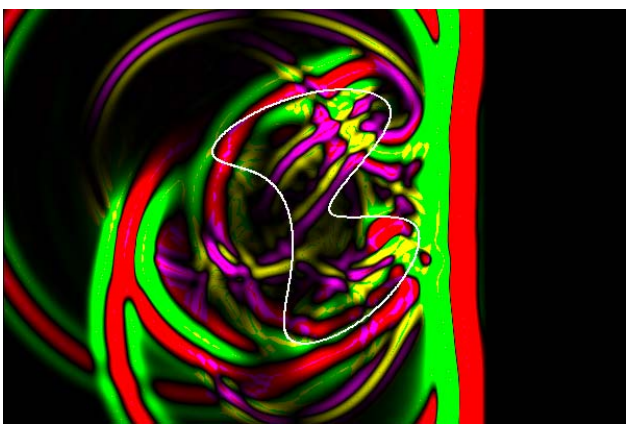
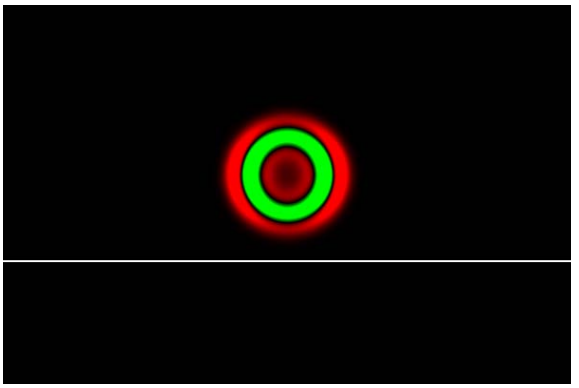
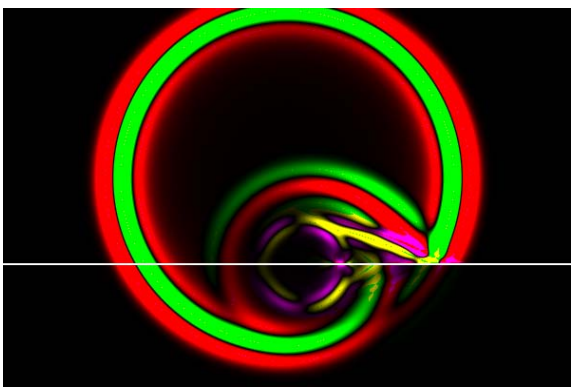


FIG. 5.5. *Test* 4: *cubic spline between identical media, at* $t = t_0 + 150\, i\, \Delta\, t$.

$(i = 0)$



$(i = 1)$



$(i = 2)$

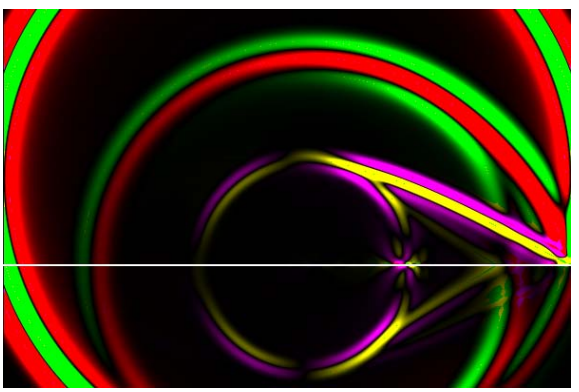


FIG. 5.6. *Test* 5: *plane interface with variable contact at* $t = t_0 + 150\, i\, \Delta t$.

Since both sides are almost stress-free surfaces in that place, this surface wave is a Rayleigh wave [1].

6. Conclusion. This paper deals with the numerical simulation of two-dimensional wave propagation in elastic solids separated by imperfect contacts. These contacts are modeled by interfaces with linear jump conditions: the [illegible] frequently used in geophysics [17] and nondestructive evaluation of materials [22]. The spring-mass conditions are incorporated into finite-difference time-domain schemes on a uniform Cartesian grid, via an [illegible]. This is an extension of previous studies on perfect contacts in one-dimensional [15] and two-dimensional [12] contexts and on imperfect contacts in a one-dimensional context [11]. The interface method also provides a fine description of the subcell geometry of the interfaces, at a negligible extra computational cost. Comparisons with original analytical solutions show the great accuracy of this approach.

Future paths of study might consist of taking into account numerically more realistic models for imperfect contact situations. A first improvement consists of describing the energy dissipation, as done, for example, in linear viscous bonding models [5]. A second improvement consists of extending the present approach for including nonlinear contact laws, which model realistic fractures and which prevent from interpenetration between the materials [23].

Appendix A. On the asymmetry of one-dimensional spring-mass conditions.

The aim of this section is to show that in some limit cases, the asymmetry of the jump conditions (2.10) has a negligible influence.

Consider the simplified case of a compressional plane wave propagating through Ω_0 normally to the plane interface Γ located at $x = \alpha$. Due to the symmetries involved in the problem, we have a one-dimensional configuration, with parameters $K = K_N$, $M = M_N$, $c_0 = c_{p0}$, and $c_1 = c_{p1}$. (A similar study could be performed with a shear plane wave and parameters $K = K_T$, $M = M_T$, $c_0 = c_{s0}$, and $c_1 = c_{s1}$.) The normalized harmonic components of the elastic displacement u and those of the elastic stress σ in medium Ω_0 (with $k_0 = \omega/c_0$) are

$$\hat{u}(x,\omega) = -i\,k_0\,e^{-ik_0x} + i\,k_0\,R^{\pm}\,e^{ik_0x},$$
$$\hat{\sigma}(x,\omega) = -\rho_0\,\omega^2\,e^{-ik_0x} - \rho_0\,\omega^2\,R^{\pm}\,e^{ik_0x},$$

and in medium Ω_1 (with $k_1 = \omega/c_1$), they are

$$\hat{u}(x,\omega) = -i\,k_1\,T^{\pm}\,e^{-ik_1x},$$
$$\hat{\sigma}(x,\omega) = -\rho_1\,\omega^2\,T^{\pm}\,e^{-ik_1x},$$

where ω is the angular frequency and $R^{\pm}$ and $T^{\pm}$ are the reflection and transmission coefficients it is required to determine (the superscripts $\pm$ are explained below). Under jump conditions shifted to the left (as in the present paper),

$$[\hat{u}(\alpha,\,\omega)] = \frac{1}{K}\,\hat{\sigma}(\alpha^-,\,\omega), \qquad [\hat{\sigma}(\alpha,\,\omega)] = -M\,\omega^2\,\hat{u}(\alpha^-,\,\omega),$$

a simple algebraic procedure gives

$$
\begin{aligned}
R^- &= \frac{\rho_1\, c_1 - \rho_0\, c_0 - i\,\omega\left(\dfrac{\rho_0\, c_0\, \rho_1\, c_1}{K} - M\right)}{\rho_0\, c_0 + \rho_1\, c_1 + i\,\omega\left(\dfrac{\rho_0\, c_0\, \rho_1\, c_1}{K} + M\right)}\, e^{-i\,\omega\frac{2\,\alpha}{c_0}}, \\
T^- &= \frac{2\,\rho_0\, c_1\left(1 + \dfrac{M\,\omega^2}{K}\right)}{\rho_0\, c_0 + \rho_1\, c_1 + i\,\omega\left(\dfrac{\rho_0\, c_0\, \rho_1\, c_1}{K} + M\right)}\, e^{i\,\omega\left(\frac{1}{c_1} - \frac{1}{c_0}\right)\alpha}.
\end{aligned}
\tag{A.1}
$$

Under jump conditions shifted to the right,

$$
[\hat{u}(\alpha,\,\omega)] = \frac{1}{K}\,\hat{\sigma}(\alpha^+,\,\omega), \qquad [\hat{\sigma}(\alpha,\,\omega)] = -M\,\omega^2\,\hat{u}(\alpha^+,\,\omega),
$$

we obtain in the same way

$$
\begin{aligned}
R^+ &= \frac{\rho_1\, c_1 - \rho_0\, c_0 - i\,\omega\left(\dfrac{\rho_0\, c_0\, \rho_1\, c_1}{K} - M\right)}{\rho_0\, c_0 + \rho_1\, c_1 + i\,\omega\left(\dfrac{\rho_0\, c_0\, \rho_1\, c_1}{K} + M\right)}\, e^{-i\,\omega\frac{2\,\alpha}{c_0}}, \\
T^+ &= \frac{2\,\rho_0\, c_1}{\rho_0\, c_0 + \rho_1\, c_1 + i\,\omega\left(\dfrac{\rho_0\, c_0\, \rho_1\, c_1}{K} + M\right)}\, e^{i\,\omega\left(\frac{1}{c_1} - \frac{1}{c_0}\right)\alpha}.
\end{aligned}
\tag{A.2}
$$

Comparisons between (A.1) and (A.2) show that [illegible] under both types of jump conditions and that

$$
\left|\frac{T^- - T^+}{T^-}\right| = \frac{M\,\omega^2}{K}.
$$

With realistic contacts [11, 18], the stiffness and mass satisfy $K \sim \rho\, c^2\,/\,h$ and $M \sim \rho\, h$, where ρ and c are the parameters of the intermediate medium (or [illegible]) and h is its thickness. Since $\omega = 2\,\pi\, c/\lambda$ (where λ is the wavelength) and $h \ll \lambda$ (a basic assumption underlying the spring-mass model), we obtain

$$
\left|\frac{T^- - T^+}{T^-}\right| \sim 4\,\pi^2\left(\frac{h}{\lambda}\right)^2 \ll 1.
$$

Hence, [illegible] under both types of jump conditions.

Appendix B. Zeroth order matrices of spring-mass conditions. The matrices $\boldsymbol{C}_l^0$ in (4.3) are ($l = 0, 1$)

$$
\boldsymbol{C}_l^0 = \begin{pmatrix}
-y' & x' & 0 & 0 & 0 \\
x' & y' & 0 & 0 & 0 \\
0 & 0 & y'^2 & -2\,x'\,y' & x'^2 \\
0 & 0 & -x'\,y' & x'^2 - y'^2 & x'\,y'
\end{pmatrix}.
$$

Setting

$$\beta_1 = \frac{\rho_0}{K_N}\,\frac{1}{\sqrt{x'^2+y'^2}}, \quad \beta_2 = \frac{\rho_0}{K_T}\,\frac{c_{s0}^2}{\sqrt{x'^2+y'^2}},$$

$$\beta_3 = \frac{M_N}{\rho_0}\,\sqrt{x'^2+y'^2}, \quad \beta_4 = \frac{M_T}{\rho_0}\,\sqrt{x'^2+y'^2},$$

the nonnull components of $\boldsymbol{E}_0^0$ in (4.3) are

$$\begin{aligned}
&\boldsymbol{E}_0^0[1,1] = \beta_1\left(y'^2 c_{p0}^2 + x'^2\left(c_{p0}^2 - 2\,c_{s0}^2\right)\right), && \boldsymbol{E}_0^0[3,3] = -\beta_3\,y',\\
&\boldsymbol{E}_0^0[1,2] = \beta_1\left(-2\,x'\,y'\,c_{s0}^2\right), && \boldsymbol{E}_0^0[3,4] = \beta_3\,x',\\
&\boldsymbol{E}_0^0[1,6] = \beta_1\left(-2\,x'\,y'\,c_{s0}^2\right), && \boldsymbol{E}_0^0[3,9] = -\beta_3\,y',\\
&\boldsymbol{E}_0^0[1,7] = \beta_1\left(x'^2 c_{p0}^2 + y'^2\left(c_{p0}^2 - 2\,c_{s0}^2\right)\right), && \boldsymbol{E}_0^0[3,10] = \beta_3\,x',\\
&\boldsymbol{E}_0^0[2,1] = \beta_2\left(-2\,x'\,y'\right), && \boldsymbol{E}_0^0[4,3] = \beta_4\,x',\\
&\boldsymbol{E}_0^0[2,2] = \beta_2\left(x'^2 - y'^2\right), && \boldsymbol{E}_0^0[4,4] = \beta_4\,y',\\
&\boldsymbol{E}_0^0[2,6] = \beta_2\left(x'^2 - y'^2\right), && \boldsymbol{E}_0^0[4,9] = \beta_4\,x',\\
&\boldsymbol{E}_0^0[2,7] = \beta_2\left(2\,x'\,y'\right), && \boldsymbol{E}_0^0[4,10] = \beta_4\,y'.
\end{aligned}$$

Appendix C. Exact solution for a plane wave on a circular interface with spring-mass conditions.

This analytical solution is obtained in six steps:

1. Fourier-transforming the incident wave (5.1);
2. writing the elastic potentials on the basis of circular functions;
3. expressing the elastic fields in terms of their potentials;
4. computing the reflection and transmission coefficients from (2.10);
5. returning to Cartesian coordinates;
6. inverting Fourier transform of the elastic fields (not described here).

Step 1. The Fourier transform $\mathcal{F}$ of a function $s(t)$, and its inverse Fourier transform $\mathcal{F}^{-1}$, are denoted by

$$\begin{aligned}
\hat{s}(\omega) &= \mathcal{F}(s(t)) = \frac{1}{2\,\pi}\int_{-\infty}^{+\infty} s(t)\,e^{-i\,\omega\,t}\,d\,t,\\
s(t) &= \mathcal{F}^{-1}(\hat{s}(\omega)) = \int_{-\infty}^{+\infty} \hat{s}(\omega)\,e^{i\,\omega\,t}\,d\,\omega,
\end{aligned} \tag{C.1}$$

where ω is the angular frequency. Applying a Fourier transform (C.1) to the part of the initial wave (5.1)

$$\Phi_{IP}(x,y,t) = f\left(t - \frac{x\cos\theta_1 + y\sin\theta_1}{c_{p1}}\right)$$

gives the harmonic potential of the incident P-wave

$$\hat{\Phi}_{IP}(x,y,\omega) = e^{-i\,\frac{\omega}{c_{p1}}(x\,\cos\theta_1 + y\,\sin\theta_1)}\,\hat{f}(\omega). \tag{C.2}$$

The Fourier transform of the bounded sinusoid (5.2) is

$$\hat{f}(\omega) = A = \frac{\omega_c}{2\,\pi}\left(\frac{1}{\omega^2-\omega_c^2}+\frac{1}{\omega^2-4\,\omega_c^2}\right)\left(e^{-i\,\frac{2\,\pi}{\omega_c}\omega}-1\right). \tag{C.3}$$

Step 2. The circular interface is centered on (x_0, y_0). A point $M(x, y) \in \Omega_1$ has polar coordinates $(x = x_0 + r\,\cos\phi, y = y_0 + r\,\sin\phi)$. We set $(l = 0,\ 1)$

$$k_{pl} = \frac{\omega}{c_{pl}}, \quad k_{sl} = \frac{\omega}{c_{sl}}, \quad S = e^{-i\,k_{p1}(x_0\,\cos\theta_1 + y_0\,\sin\theta_1)}, \quad \theta = \phi - \theta_1.$$

The first-kind Bessel functions J_n satisfy the classical property [14]

$$e^{-i\,r\,\cos\theta} = \sum_{n=0}^{+\infty} \varepsilon_n\, i^n\,(-1)^n\,\cos n\,\theta\, J_n(r), \tag{C.4}$$

with $\varepsilon_n = 1$ if $n = 0$, 2 else. From (C.2), (C.3), and (C.4), we can therefore express the harmonic potential of the incident P-wave as

$$\hat{\Phi}_{IP}(x, y, \omega) = A\,S \sum_{n=0}^{+\infty} \varepsilon_n\, i^n (-1)^n\,\cos n\,\theta\, J_n(k_{p1}\,r). \tag{C.5}$$

To satisfy the Sommerfeld condition, the harmonic elastic potential $\hat{\Phi}_{RP}$ of reflected P-waves and the harmonic pseudopotential $\hat{\boldsymbol{\Psi}}_{RS} = (0, 0, \hat{\Psi}_{RS})$ of reflected SV-waves are written on the basis of second-kind Hankel functions H_n. To prevent singularities from occuring at $r = 0$, the harmonic potential $\hat{\Phi}_{TP}$ of transmitted P-waves and the harmonic pseudopotential $\hat{\boldsymbol{\Psi}}_{TS} = (0, 0, \hat{\Psi}_{TS})$ of transmitted SV-waves are written on the basis of first-kind Bessel functions. Hence, we write

$$\begin{aligned}
&\hat{\Phi}_{RP} = \sum_{n=0}^{+\infty} R_n^p\,\cos n\,\theta\, H_n(k_{p1}\,r), \qquad \hat{\Psi}_{RS} = \sum_{n=0}^{+\infty} R_n^s\,\sin n\,\theta\, H_n(k_{s1}\,r),\\
&\hat{\Phi}_{TP} = \sum_{n=0}^{+\infty} T_n^p\,\cos n\,\theta\, J_n(k_{p0}\,r), \qquad \hat{\Psi}_{TS} = \sum_{n=0}^{+\infty} T_n^s\,\sin n\,\theta\, J_n(k_{s0}\,r),
\end{aligned} \tag{C.6}$$

where R_n^p, R_n^s, T_n^p, and T_n^s are the unknown coefficients of reflection and transmission, which still remain to be determined. Note that numerical routines [16] usually provide Y_n and $Y_n^{'}$ (with $Y_n = J_n$ or H_n). Later in the text, we will need $Y_n^{''}$; to determine this last value, we use the differential equation satisfied by the Bessel and Hankel functions

$$Y_n^{''}(x) + \frac{1}{x}\,Y_n^{'}(x) + \left(1 - \left(\frac{n}{x}\right)^2\right) Y_n(x) = 0.$$

Step 3. The elastic displacement $\boldsymbol{u} = {}^T(u_r, u_\theta)$ is deduced from Φ (for P-waves) or from $\boldsymbol{\Psi} = (0, 0, \Psi)$ (for SV-waves) by

$$\boldsymbol{u} = \mathbf{grad}\,\Phi \ \text{ for P-waves}, \qquad \boldsymbol{u} = \mathbf{curl}\,\boldsymbol{\Psi} \ \text{ for SV-waves}. \tag{C.7}$$

In cylindrical coordinates, the **grad** and **curl** operators are

$$\mathbf{grad}\,\Phi = \left(\frac{\partial\,\Phi}{\partial\,r},\ \frac{1}{r}\frac{\partial\,\Phi}{\partial\,\theta}\right), \qquad \mathbf{curl}\,\boldsymbol{\Psi}(0, 0, \Psi) = \left(\frac{1}{r}\frac{\partial\,\Psi}{\partial\,\theta},\ -\frac{\partial\,\Psi}{\partial\,r}\right). \tag{C.8}$$

In cylindrical coordinates, the three independent components of $\boldsymbol{\sigma}$ are [14]

$$\begin{aligned}
\sigma_{rr} &= (\lambda + 2\,\mu)\ \frac{\partial\, u_r}{\partial\, r} + \lambda \left(\frac{u_r}{r} + \frac{1}{r}\ \frac{\partial\, u_\theta}{\partial\, \theta} \right), \\
\sigma_{r\theta} &= \mu \left(\frac{\partial\, u_\theta}{\partial\, r} - \frac{u_\theta}{r} + \frac{1}{r}\ \frac{\partial\, u_r}{\partial\, \theta} \right), \\
\sigma_{\theta\theta} &= (\lambda + 2\,\mu)\ \left(\frac{1}{r}\ \frac{\partial\, u_\theta}{\partial\, \theta} + \frac{u_r}{r} \right) + \lambda\, \frac{\partial\, u_r}{\partial\, r},
\end{aligned} \tag{C.9}$$

where λ and μ are the Lamé coefficients. From (C.5), (C.6), (C.9), we easily deduce the harmonic fields over the whole domain. The components of the harmonic incident P-wave are

$$\begin{aligned}
\hat{u}_r^{ip} &= A\, S\, k_{p1} \sum_{n=0}^{+\infty} \varepsilon_n\, i^n (-1)^n \cos n\,\theta\, J_n^{'}(k_{p1}\, r), \\
\hat{u}_\theta^{ip} &= -\frac{A\, S}{r} \sum_{n=0}^{+\infty} \varepsilon_n\, i^n\, \frac{n\, \sin n\,\theta}{r}\, J_n(k_{p1}\, r), \\
\hat{\sigma}_{rr}^{ip} &= A\, S \sum_{n=0}^{+\infty} \varepsilon_n\, i^n (-1)^n \cos n\,\theta \Big((\lambda_1 + 2\,\mu_1)\, k_{p1}^2\, J_n^{''}(k_{p1}\, r) \\
&\qquad + \lambda_1 \frac{k_{p1}}{r}\, J_n^{'}(k_{p1}\, r) - \lambda_1 \left(\frac{n}{r}\right)^2\, J_n(k_{p1}\, r) \Big), \\
\hat{\sigma}_{r\theta}^{ip} &= A\, S\, 2\,\mu_1 \sum_{n=0}^{+\infty} \varepsilon_n\, i^n (-1)^n \sin n\,\theta \left(\frac{1}{r^2} J_n(k_{p1}\, r) - \frac{k_{p1}}{r}\, J_n^{'}(k_{p1}\, r) \right), \\
\hat{\sigma}_{\theta\theta}^{ip} &= A\, S \sum_{n=0}^{+\infty} \varepsilon_n\, i^n (-1)^n \cos n\,\theta \Big(\lambda_1\, k_{p1}^2\, J_n^{''}(k_{p1}\, r) \\
&\qquad + (\lambda_1 + 2\,\mu_1)\, \frac{k_{p1}}{r}\, J_n^{'}(k_{p1}\, r) - (\lambda_1 + 2\,\mu_1) \left(\frac{n}{r}\right)^2\, J_n(k_{p1}\, r) \Big).
\end{aligned} \tag{C.10}$$

The components of harmonic reflected P-waves are

(C.11)

$$\begin{aligned}
\hat{u}_r^{rp} &= k_{p1} \sum_{n=0}^{+\infty} R_n^p\, \cos n\,\theta\, H_n^{'}(k_{p1} r), \\
\hat{u}_\theta^{rp} &= -\frac{1}{r} \sum_{n=0}^{+\infty} R_n^p\, n\, \sin n\,\theta\, H_n(k_{p1} r), \\
\hat{\sigma}_{rr}^{rp} &= \sum_{n=0}^{+\infty} R_n^p \cos n\,\theta \left((\lambda_1 + 2\,\mu_1)\, k_{p1}^2\, H_n^{''}(k_{p1} r) + \lambda_1 \frac{k_{p1}}{r} H_n^{'}(k_{p1} r) - \lambda_1 \left(\frac{n}{r}\right)^2 H_n(k_{p1} r) \right), \\
\hat{\sigma}_{r\theta}^{rp} &= 2\,\mu_1 \sum_{n=0}^{+\infty} R_n^p\, n\, \sin n\,\theta \left(\frac{1}{r^2} H_n(k_{p1} r) - \frac{k_{p1}}{r} H_n^{'}(k_{p1} r) \right), \\
\hat{\sigma}_{\theta\theta}^{rp} &= \sum_{n=0}^{+\infty} R_n^p \cos n\,\theta \Big(\lambda_1\, k_{p1}^2\, H_n^{''}(k_{p1} r) + (\lambda_1 + 2\,\mu_1)\ \frac{k_{p1}}{r}\, H_n^{'}(k_{p1} r) \\
&\qquad - (\lambda_1 + 2\,\mu_1) \left(\frac{n}{r}\right)^2\, H_n(k_{p1} r) \Big).
\end{aligned}$$

The components of harmonic reflected SV-waves are

(C.12)
$$
\begin{aligned}
\hat{u}_r^{rs} &= \sum_{n=0}^{+\infty} R_n^s \, \frac{n \, \cos n\,\theta}{r} \, H_n(k_{s1} r), \\
\hat{u}_\theta^{rs} &= -k_{s1} \sum_{n=0}^{+\infty} R_n^s \, \sin n\,\theta \, H_n^{'}(k_{s1} r), \\
\hat{\sigma}_{rr}^{rs} &= -2\,\mu_1 \sum_{n=0}^{+\infty} R_n^s \, \cos n\,\theta \left(\frac{1}{r^2} \, H_n(k_{s1} r) - \frac{k_{s1}}{r} \, H_n^{'}(k_{s1} r) \right), \\
\hat{\sigma}_{r\theta}^{rs} &= -\mu_1 \sum_{n=0}^{+\infty} R_n^s \, \sin n\,\theta \left(k_{s1}^2 \, H_n^{''}(k_{s1} r) - \frac{k_{s1}}{r} \, H_n^{'}(k_{s1} r) + \left(\frac{n}{r} \right)^2 H_n(k_{s1} r) \right), \\
\hat{\sigma}_{\theta\theta}^{rs} &= -2\,\mu_1 \sum_{n=0}^{+\infty} R_n^s \, n \, \cos n\,\theta \left(\frac{k_{s1}}{r} \, H_n^{'}(k_{s1} r) - \frac{1}{r^2} \, H_n(k_{s1} r) \right).
\end{aligned}
$$

The components of harmonic transmitted P-waves are

(C.13)
$$
\begin{aligned}
\hat{u}_r^{tp} &= k_{p0} \sum_{n=0}^{+\infty} T_n^p \, \cos n\,\theta \, J_n^{'}(k_{p0} r), \\
\hat{u}_\theta^{tp} &= -\frac{1}{r} \sum_{n=0}^{+\infty} T_n^p \, n \, \sin n\,\theta \, J_n(k_{p0} r), \\
\hat{\sigma}_{rr}^{tp} &= \sum_{n=0}^{+\infty} T_n^p \cos n\,\theta \Big((\lambda_0 + 2\,\mu_0) \, k_{p0}^2 \, J_n^{''}(k_{p0} r) \\
&\qquad + \lambda_0 \, \frac{k_{p0}}{r} \, J_n^{'}(k_{p0} r) - \lambda_0 \left(\frac{n}{r} \right)^2 J_n(k_{p0} r) \Big), \\
\hat{\sigma}_{r\theta}^{tp} &= 2\,\mu_0 \sum_{n=0}^{+\infty} T_n^p \, n \, \sin n\,\theta \left(\frac{1}{r^2} J_n(k_{p0} r) - \frac{k_{p0}}{r} J_n^{'}(k_{p0} r) \right), \\
\hat{\sigma}_{\theta\theta}^{tp} &= \sum_{n=0}^{+\infty} T_n^p \cos n\,\theta \Big(\lambda_0 \, k_{p0}^2 \, J_n^{''}(k_{p0} r) + (\lambda_0 + 2\,\mu_0) \, \frac{k_{p0}}{r} \, J_n^{'}(k_{p0} r) \\
&\qquad - (\lambda_0 + 2\,\mu_0) \left(\frac{n}{r} \right)^2 J_n(k_{p0} r) \Big).
\end{aligned}
$$

Lastly, the components of harmonic transmitted SV-waves are

(C.14)
$$
\begin{aligned}
\hat{u}_r^{ts} &= \sum_{n=0}^{+\infty} T_n^s \, \frac{n \, \cos n\,\theta}{r} \, J_n(k_{s0} r), \\
\hat{u}_\theta^{ts} &= -k_{s0} \sum_{n=0}^{+\infty} T_n^s \, \sin n\,\theta \, J_n^{'}(k_{s0} r), \\
\hat{\sigma}_{rr}^{ts} &= -2\,\mu_0 \sum_{n=0}^{+\infty} T_n^s \, \cos n\,\theta \left(\frac{1}{r^2} \, J_n(k_{s0} r) - \frac{k_{s0}}{r} \, J_n^{'}(k_{s0} r) \right), \\
\hat{\sigma}_{r\theta}^{ts} &= -\mu_0 \sum_{n=0}^{+\infty} T_n^s \, \sin n\,\theta \left(k_{s0}^2 \, J_n^{''}(k_{s0} r) - \frac{k_{s0}}{r} \, J_n^{'}(k_{s0} r) + \left(\frac{n}{r} \right)^2 J_n(k_{s0} r) \right), \\
\hat{\sigma}_{\theta\theta}^{ts} &= -2\,\mu_0 \sum_{n=0}^{+\infty} T_n^s \, n \, \cos n\,\theta \left(\frac{k_{s0}}{r} \, J_n^{'}(k_{s0} r) - \frac{1}{r^2} \, J_n(k_{s0} r) \right).
\end{aligned}
$$

Step 4. We now compute the coefficients R_n^p, R_n^s, T_n^p, and T_n^s. For this purpose, we deduce from the spring-mass conditions (2.10) that

$$
\begin{aligned}
&\left(\hat{u}_r^{ip} + \hat{u}_r^{rp} + \hat{u}_r^{rs}\right)(a^+,\theta) = \left(\hat{u}_r^{tp} + \hat{u}_r^{ts} + \frac{1}{K_N}\left(\hat{\sigma}_{rr}^{tp} + \hat{\sigma}_{rr}^{ts}\right)\right)(a^-,\theta),\\
&\left(\hat{u}_\theta^{ip} + \hat{u}_\theta^{rp} + \hat{u}_\theta^{rs}\right)(a^+,\theta) = \left(\hat{u}_\theta^{tp} + \hat{u}_\theta^{ts} + \frac{1}{K_T}\left(\hat{\sigma}_{r\theta}^{tp} + \hat{\sigma}_{r\theta}^{ts}\right)\right)(a^-,\theta),\\
&\left(\hat{\sigma}_{rr}^{ip} + \hat{\sigma}_{rr}^{rp} + \hat{\sigma}_{rr}^{rs}\right)(a^+,\theta) = \left(\hat{\sigma}_{rr}^{tp} + \hat{\sigma}_{rr}^{ts} - M_N\,\omega^2\left(\hat{u}_r^{tp} + \hat{u}_r^{ts}\right)\right)(a^-,\theta),\\
&\left(\hat{\sigma}_{r\theta}^{ip} + \hat{\sigma}_{r\theta}^{rp} + \hat{\sigma}_{r\theta}^{rs}\right)(a^+,\theta) = \left(\hat{\sigma}_{r\theta}^{tp} + \hat{\sigma}_{r\theta}^{ts} - M_T\,\omega^2\left(\hat{u}_\theta^{tp} + \hat{u}_\theta^{ts}\right)\right)(a^-,\theta)
\end{aligned}
\tag{C.15}
$$

for all θ. Applying (C.15) to the fields (C.10)–(C.14) yields an infinite number of linear systems. For computational purpose, these systems are computed up to N_{Bessel} terms, giving the systems ($n = 0, 1, \ldots, N_{Bessel}$)

$$
\boldsymbol{Q}_n\,\boldsymbol{X}_n = \boldsymbol{Y}_n \qquad \text{with } \boldsymbol{X}_n = {}^T\left(R_n^p,\, R_n^s,\, T_n^p,\, T_n^s\right)
\tag{C.16}
$$

and

$$
\boldsymbol{Y}_n = -A\,S\,\varepsilon_n\,i^n(-1)^n
\begin{pmatrix}
J_n'(k_{p1}\,a)\\
\dfrac{n}{a}\,J_n(k_{p1}\,a)\\
(\lambda_1 + 2\,\mu_1)\,k_{p1}^2\,J_n''(k_{p1}\,a) + \dfrac{\lambda_1}{a}\,k_{p1}\,J_n'(k_{p1}\,a)\\
\qquad -\lambda_1\left(\dfrac{n}{a}\right)^2 J_n(k_{p1}\,a)\\
2\,\mu_1\,n\left(\dfrac{1}{a^2}\,J_n(k_{p1}\,a) - \dfrac{k_{p1}}{a}\,J_n'(k_{p1}\,a)\right)
\end{pmatrix}.
\tag{C.17}
$$

The coefficients of the matrices $\boldsymbol{Q}_n$ are

$$
\begin{aligned}
&\boldsymbol{Q}_n[1,1] = k_{p1}\,H_n'(k_{p1}\,a), \quad \boldsymbol{Q}_n[1,2] = \frac{n}{a}\,H_n(k_{s1}\,a),\\
&\boldsymbol{Q}_n[1,3] = -\left(k_{p0}\,J_n'(k_{p0}\,a) + \frac{1}{K_N}\left((\lambda_0 + 2\,\mu_0)\,k_{p0}^2\,J_n''(k_{p0}\,a)\right.\right.\\
&\qquad\left.\left. + \frac{\lambda_0}{a}\,k_{p0}\,J_n'(k_{p0}\,a) - \lambda_0\left(\frac{n}{a}\right)^2 J_n(k_{p0}\,a)\right)\right),\\
&\boldsymbol{Q}_n[1,4] = -\left(\frac{n}{a}\,J_n(k_{s0}\,a) - \frac{2\,\mu_0\,n}{K_N}\left(\frac{1}{a^2}\,J_n(k_{s0}\,a) - \frac{k_{s0}}{a}\,J_n'(k_{s0}\,a)\right)\right),\\
&\boldsymbol{Q}_n[2,1] = \frac{n}{a}H_n(k_{p1}\,a), \quad \boldsymbol{Q}_n[2,2] = k_{s1}\,H_n'(k_{s1}\,a),\\
&\boldsymbol{Q}_n[2,3] = -\left(\frac{n}{a}\,J_n(k_{p0}\,a) - \frac{2\,\mu_0\,n}{K_T}\left(\frac{1}{a^2}\,J_n(k_{p0}\,a) - \frac{k_{p0}}{a}\,J_n'(k_{p0}\,a)\right)\right),\\
&\boldsymbol{Q}_n[2,4] = -\left(k_{s0}\,J_n'(k_{s0}\,a) + \frac{\mu_0}{K_T}\left(k_{s0}^2\,J_n''(k_{s0}\,a)\right.\right.\\
&\qquad\left.\left. - \frac{k_{s0}}{a}\,k_{p0}\,J_n'(k_{s0}\,a) + \left(\frac{n}{a}\right)^2 J_n(k_{s0}\,a)\right)\right),
\end{aligned}
\tag{C.18}
$$

$$
\begin{aligned}
\boldsymbol{Q}_n[3,1] &= (\lambda_1 + 2\,\mu_1)\,k_{p1}^2\,H_n^{''}(k_{p1}\,a) + \frac{\lambda_1\,k_{p1}}{a}\,H_n^{'}(k_{p1}\,a) - \lambda_1\left(\frac{n}{a}\right)^2\,H_n(k_{p1}\,a),\\
\boldsymbol{Q}_n[3,2] &= -2\,\mu_1\,n\left(\frac{1}{a^2}\,H_n(k_{s1}\,a) - \frac{k_{s1}}{a}\,H_n^{'}(k_{s1}\,a)\right),\\
\boldsymbol{Q}_n[3,3] &= -\left((\lambda_0 + 2\,\mu_0)\,k_{p0}^2\,J_n^{''}(k_{p0}\,a) + \left(\frac{\lambda_1}{a} - M_N\,\omega^2\right)k_{p0}\,J_n^{'}(k_{p0}\,a)\right.\\
&\qquad \left. -\lambda_0\left(\frac{n}{a}\right)^2\,J_n(k_{p0}\,a)\right),\\
\boldsymbol{Q}_n[3,4] &= \left(\frac{2\,\mu_0\,n}{a^2} + M_N\,\omega^2\,\frac{n}{a}\right)\,J_n(k_{s0}\,a) - \frac{2\,\mu_0\,n\,k_{s0}}{a}\,J_n^{'}(k_{s0}\,a),\\
\boldsymbol{Q}_n[4,1] &= 2\,\mu_1\,n\left(\frac{1}{a^2}\,H_n(k_{p1}\,a) - \frac{k_{p1}}{a}\,H_n^{'}(k_{s1}\,a)\right),\\
\boldsymbol{Q}_n[4,2] &= -\mu_1\left(k_{s1}^2\,H_n^{''}(k_{s1}\,a) - \frac{k_{s1}}{a}\,H_n^{'}(k_{s1}\,a) + \left(\frac{n}{a}\right)^2\,H_n(k_{s1}\,a)\right),\\
\boldsymbol{Q}_n[4,3] &= -\left(\left(\frac{2\,\mu_0\,n}{a^2} + M_T\,\omega^2\,\frac{n}{a}\right)\,J_n(k_{p0}\,a) - \frac{2\,\mu_0\,n\,k_{p0}}{a}\,J_n^{'}(k_{p0}\,a)\right),\\
\boldsymbol{Q}_n[4,4] &= \mu_0\,k_{s0}^2\,J_n^{''}(k_{s0}\,a) - \left(\frac{\mu_0}{a} + M_T\,\omega^2\right)k_{s0}\,J_n^{'}(k_{s0}\,a) + \mu_0\left(\frac{n}{a}\right)^2\,J_n(k_{s0}\,a).
\end{aligned}
$$

Since the values of Hankel functions can be huge (typically 10^{60}), one must be careful when inverting (C.16). Normalization of $\boldsymbol{Q}_n$ and the use of extended arithmetic are required for this purpose.

Step 5. To return to Cartesian coordinates, we use the rotation formulas [6]

(C.19)
$$
\begin{pmatrix} v_1 \\ v_2 \\ \sigma_{11} \\ \sigma_{12} \\ \sigma_{22} \end{pmatrix} = \begin{pmatrix} \cos\phi & -\sin\phi & 0 & 0 & 0 \\ \sin\phi & \cos\phi & 0 & 0 & 0 \\ 0 & 0 & \cos^2\phi & -2\,\sin\phi\,\cos\phi & \sin^2\phi \\ 0 & 0 & \sin\phi\cos\phi & \cos^2\phi - \sin^2\phi & -\sin\phi\,\cos\phi \\ 0 & 0 & \sin^2\phi & 2\,\sin\phi\,\cos\phi & \cos^2\phi \end{pmatrix} \begin{pmatrix} v_r \\ v_\theta \\ \sigma_{rr} \\ \sigma_{r\theta} \\ \sigma_{\theta\theta} \end{pmatrix}.
$$

REFERENCES

[1] J. D. Achenbach, *Wave Propagation in Elastic Solids*, North–Holland, Amsterdam, 1973.

[2] J. M. Baik and R. B. Thompson, *Ultrasonic scattering from imperfect interfaces: A quasi-static model*, J. Nondestructive Eval., 4 (1985), pp. 177–196.

[3] G. B.van Baren, W. A. Mulder, and G. C. Herman, *Finite-difference modeling of scalar-wave propagation in cracked media*, Geophysics, 66 (2001), pp. 267–276.

[4] R. T. Coates and M. Schoenberg, *Finite-difference modeling of faults and fractures*, Geophysics, 69 (1998), pp. 1514–1526.

[5] M. Fehler, *Interaction of seismic waves with a viscous liquid layer*, Bull. Seismol. Soc. Amer., 72 (1982), pp. 55–72.

[6] P. Germain and P. Muller, *Introduction à la Mécanique des Milieux Continus*, Masson, Paris, 1995.

[7] B. Gu, K. Nihei, and L. Myer, *Numerical simulation of elastic wave propagation in fractured rock with the boundary integral equation method*, J. Geophys. Res., 101 (1996), pp. 933–943.

[8] M. Haney and R. Sneider, *Numerical modeling of waves incident on slip discontinuities*, in Annual Report of the Consortium Project on Seismic Inverse Methods for Complex Structures, 2003, pp. 1–10.

[9] R. J. LeVeque, *Numerical Methods for Conservation Laws*, Birkhäuser, Boston, 1990.
[10] R. J. LeVeque, *Wave propagation algorithms for multi-dimensional hyperbolic systems*, J. Comput. Phys., 131 (1997), pp. 327–353.
[11] B. Lombard and J. Piraux, *How to incorporate the spring-mass conditions in finite-difference schemes*, SIAM J. Sci. Comput., 24 (2003), pp. 1379–1407.
[12] B. Lombard and J. Piraux, *Numerical treatment of two-dimensional interfaces for acoustic and elastic waves*, J. Comput. Phys., 195 (2004), pp. 90–116.
[13] A. E. H. Love, *A Treatise on the Mathematical Theory of Elasticity*, Dover Publications, New York, 1944.
[14] P. M. Morse and H. Feshbach, *Methods of Theoretical Physics*, McGraw-Hill, New York, 1953.
[15] J. Piraux and B. Lombard, *A new interface method for hyperbolic problems with discontinuous coefficients. 1D acoustic example*, J. Comput. Phys., 168 (2001), pp. 227–248.
[16] W. H. Press, S. A. Teukolskyn, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in Fortran: The Art of Scientific Computing*, Cambridge University Press, Cambridge, UK, 1992.
[17] L. Pyrak-Nolte, L. Myer, and N. Cook, *Transmission of seismic waves across single natural fractures*, J. Geophys. Res., 95 (1990), pp. 8617–8638.
[18] S. I. Rokhlin and Y. J. Wang, *Analysis of boundary conditions for elastic wave interaction with an interface between two solids*, J. Acoust. Soc. Amer., 89 (1991), pp. 503–515.
[19] E. Saenger, N. Gold, and A. Shapiro, *Modeling the propagation of elastic waves using a modified finite-difference grid*, Wave Motion, 31 (2000), pp. 77–92.
[20] M. Schoenberg, *Elastic wave behavior across linear slip interfaces*, J. Acoust. Soc. Amer., 68 (1980), pp. 1516–1521.
[21] H.G. Tattersall, *The ultrasonic pulse-echo technique as applied to adhesion testing*, J. Appl. Phys., 6 (1973), pp. 819–832.
[22] V. Vlasie and M. Rousseau, *Acoustical validation of the rheological models for a structural bond*, Wave Motion, 37 (2003), pp. 333–349.
[23] J. Zhao and J. G. Cai, *Transmission of elastic P-waves across single fractures with a nonlinear normal deformational behavior*, Rock Mech. Rock Engrg., 34 (2001), pp. 3–22.

SIAM J. SCI. COMPUT.
Vol. 28, No. 1, pp. 206–223
© 2006 Society for Industrial and Applied Mathematics

A LOCAL LEVEL SET METHOD FOR PARAXIAL GEOMETRICAL OPTICS*

JIANLIANG QIAN† AND SHINGYU LEUNG†

Abstract. We propose a local level set method for constructing the geometrical optics term in the paraxial formulation for the high frequency asymptotics of two-dimensional (2-D) acoustic wave equations. The geometrical optics term consists of two multivalued functions: a travel-time function satisfying the eikonal equation locally and an amplitude function solving a transport equation locally. The multivalued travel-times are obtained by solving a level set equation and a travel-time equation with a forcing term. The multivalued amplitudes are computed by a new Eulerian formula based on the gradients of travel-times and takeoff angles. As a byproduct the method is also able to capture the caustic locations. The proposed Eulerian method has complexity of $O(N^2 \mathrm{Log} N)$, rather than $O(N^4)$ as typically seen in the Lagrangian ray tracing method. Several examples including the well-known Marmousi synthetic model illustrate the accuracy and efficiency of the Eulerian method.

Key words. paraxial eikonal equations, level sets, multivalued geometrical optics, caustics

AMS subject classifications. Primary 54C40, 14E20; Secondary 46E25, 20C20

DOI. 10.1137/030601673

1. Introduction.

Geometrical optics is the branch of optics which is characterized by the neglect of the wavelength, i.e., that corresponding to the limiting case (wavelength) going to zero ([illegible]), or, equivalently, for the wave number near infinity ([illegible]), since in this approximation the optical laws may be formulated in the language of geometry [5]. This definition is based on short wavelength assumption of light and can be generalized to deal with other wave phenomena as well. The linear or nonlinear partial differential equations describing these wave propagations involve a parameter, such as the wavelength λ, which is small compared to all other lengths in the problem [48]. The asymptotic method is for the asymptotic solution of PDEs governing these wave propagations.

Consider the linear acoustic wave equation. According to the Debye procedure, inserting the high frequency asymptotic ansatz into the wave equation reduces the resolution of the PDE into the resolution for a sequence of ODEs. Among all the terms in the asymptotic ansatz, the most important term is the zeroth order term, the so-called geometrical optics term [25]. The geometrical optics term consists of two functions, a phase function satisfying the eikonal equation, a first-order nonlinear PDE, and an amplitude function solving a linear transport equation with the phase gradient as coefficients. Thus to construct the geometrical optics term the eikonal equation has to be solved first for the phase function; then the transport equation for amplitude might be integrated afterwards.

Conventionally, the eikonal equation was solved by the method of characteristics, aka. the ray tracing method in the seismology and optics; thus the method inherits the intrinsic shortcoming of the Lagrangian method, i.e., the nonuniform resolution in the desired computational domain [8]. In this setup, the amplitude function could be obtained by solving an ODE system as well, and it has the similar shortcoming

*Received by the editors November 4, 2003; accepted for publication (in revised form) September 22, 2005; published electronically March 3, 2006. This research was supported by ONR grant N00014-02-1-0720.

http://www.siam.org/journals/sisc/28-1/60167.html

†Department of Mathematics, UCLA, Los Angeles, CA 90095-1555 (qian@math.wichita.edu, syleung@math.ucla.edu).

[8]. Certainly it is possible to overcome this drawback by introducing complicated interpolation and bookkeeping data structures [46]. However, in the late 80's Vidale [45], and van Trier and Symes [44] introduced direct discretizations of the eikonal equation based on finite difference schemes, which are Eulerian approaches and yield uniform resolutions of the solution in the computational domain. As pointed out in [44], the obtained solution should be understood as the minimum phase or the first-arrival travel-time in the viscosity solution sense for Hamilton–Jacobi equations [26, 11]. Once the travel-time is obtained, the transport equation for amplitude might be integrated [49, 42, 35]. Since the viscosity solution usually develops kinks, the gradient of the phase function is discontinuous. Therefore, the resulting solution for the transport equation has to be understood in the measure sense [15]. The difference between the ray tracing solution and the finite difference solution of the eikonal equation is that one is multiple-valued and the other single-valued [3, 13]. Although the viscosity concept and related numerical methods provide a natural Eulerian framework for geometrical optics, the drawback is that it provides only first arrivals while some applications may require all arrival times. Moreover, White [47] proved that there is a high probability for so-called transmission caustics to occur in an inhomogeneous medium. Beyond transmission caustics, more than one ray passes over each point in space so that the phase is multivalued. Therefore, it is important to design Eulerian methods for multivalued solutions of the eikonal equation, in general, the Hamilton–Jacobi equation.

In the last decade or so, there have been a lot of efforts in this direction: geometrical domain decomposition type methods [2], slowness matching method [40, 41], dynamic surface extension method [37, 36], segment projection method [14], level set method [29, 33, 24, 10, 34], and phase space method [17]. See [3, 13] for up-to-date reviews of the above methods and moment-based methods [12, 19, 23, 20].

In this work we present a local level set method for paraxial geometrical optics. By paraxial we mean that the wave propagation has a spatial orientation so that one of the spatial directions may be viewed as an evolution direction. This criterion may be quantified by assuming a subhorizontal condition for the eikonal equation [41]. In [34], we presented a global level set method for the paraxial eikonal equation in the two-dimensional (2-D) case, and the computed multivalued travel-time matched the ray tracing solution very well even in the difficult synthetic Marmousi model. Therefore we proceed along the same line and design a fast local level set method for both multivalued travel-times and amplitudes, thus constructing the geometrical optics term. This construction will be useful for modern high resolution seismic imaging [18, 21, 39, 7, 28]. Our formulation for computing multivalued amplitude is based upon a level set formulation and an extension of the Eulerian amplitude formulation used in [35], which is different from the one used in [29]. As a byproduct, our formulation also yields the cuspoid caustic curves (A_3 type) [43], which is related to the catastrophe theory; thus it suggests that the level set theory can be used to study the catastrophe development. See [4] for another Eulerian method for capturing caustics. The overall complexity of our level set method is $O(N^2 \mathrm{Log} N)$ rather than $O(N^4)$ as typically seen in the ray tracing formulation.

The rest of the paper is organized as follows: section 2 presents very briefly the paraxial formulation of the eikonal equation and the level set formulation for wavefront locations, travel-times, amplitudes and caustics; section 3 gives implementation details of the current formulation using the local level set method; section 4 shows some numerical examples to illustrate the efficiency and accuracy of the Eulerian method; and section 5 indicates some possible future works.

2. Governing equations.

2.1. Paraxial formulation for the eikonal equation. Consider a point source condition for the eikonal equation defined in an open, bounded domain

$$\Omega = \{(x,z) : x_{\min} \le x \le x_{\max}, 0 \le z \le z_{\max}\}. \tag{1}$$

To emphasize the point source condition the eikonal equation is written as follows,

$$|\nabla_{\mathbf{x}} \tau(\mathbf{x}, \mathbf{x}_s)| = \frac{1}{c(\mathbf{x})}. \tag{2}$$

$$\lim_{\mathbf{x}\to\mathbf{x}_s} \frac{\tau(\mathbf{x},\mathbf{x}_s)}{\|\mathbf{x}-\mathbf{x}_s\|} = \frac{1}{c(\mathbf{x}_s)}, \qquad \tau \ge 0, \tag{3}$$

where $\mathbf{x}_s$ is the given source point.

In some applications, for example, wave propagation in reflection seismics, the travel-times of interest are carried by the so-called subhorizontal rays [21, 39, 35], where subhorizontal means [illegible]. By the method of characteristics for the eikonal equation (2), the point source condition (3) and the subhorizontal condition, we use depth z as the running parameter so that we have a reduced system [34]

$$\frac{dx}{dz} = \tan\theta \tag{4}$$

$$\frac{d\theta}{dz} = \frac{1}{c}\left(\frac{\partial c}{\partial z}\tan\theta - \frac{\partial c}{\partial x}\right) \tag{5}$$

with

$$x|_{z=0} = x_s \tag{6}$$

$$\theta|_{z=0} = \theta_s, \tag{7}$$

where θ_s varies from $-\theta_{\max} \le \theta \le \theta_{\max} < \frac{\pi}{2}$. In addition, the travel-time is computed by integrating

$$\frac{dt}{dz} = \frac{1}{c\cos\theta} \tag{8}$$

with

$$t|_{z=0} = 0. \tag{9}$$

2.2. Level set equation for wavefronts. If we define $\phi = \phi(z,x,\theta)$ such that the zero level set, $\{(x(z),\theta(z)) : \phi(z,x(z),\theta(z)) = 0\}$, gives the location of the reduced bicharacteristic strip $(x(z),\theta(z))$ at z, then we may differentiate the zero level set equation with respect to z to obtain

$$\phi_z + u\phi_x + v\phi_\theta = 0 \tag{10}$$

with

$$u = \frac{dx}{dz} \text{ and } v = \frac{d\theta}{dz}, \tag{11}$$

which are given by the ray equations (4-5). In essence, we embed the ray tracing equations as the velocity field, $\mathbf{u} = (u, v)$, into the level set equation which governs the motion of the bicharacteristic strips in the phase space.

The initial condition for the level set motion equation (10) is taken to be

$$\phi|_{z=0} = \phi(0, x, \theta) = x - x_s \,, \tag{12}$$

which is obtained from initial conditions (6) and (7). This is a signed distance function, satisfying $|\nabla_{x,\theta}\phi| = 1$, to the initial phase space curve

$$\{(x, \theta) : x = x_s, -\theta_{\max} \le \theta \le \theta_{\max}\} \tag{13}$$

in the reduced phase space

$$\Omega_p = \{(x, \theta) : x_{\min} \le x \le x_{\max}, -\theta_{\max} \le \theta \le \theta_{\max}\}. \tag{14}$$

Using the same boundary condition as in [34], we have

$$\left.\frac{\partial \phi}{\partial \mathbf{n}}\right|_{\partial\Omega_p} = 0 \tag{15}$$

where $\mathbf{n}$ is the outward normal along the boundary of Ω_p.

2.3. Travel-time. To determine the travel-time of the ray from the above level set equation, we now derive a corresponding equation governing the evolution of travel-time. By the subhorizontal condition in the paraxial formulation and the ray equation (8), let $F_{\mathbf{u}}(x, \theta; z)$ be the flow generated by the velocity field $\mathbf{u}$ in the phase space (x, θ) along the z-direction. Thus we can write

$$\frac{dT}{dz}(z, F_{\mathbf{u}}(x, \theta; z)) = \frac{1}{c \cos\theta} \,. \tag{16}$$

Therefore, having $t = T(z, x, \theta)$, we get the following advection equation:

$$\frac{dt}{dz} = \frac{dT}{dz} = T_z + uT_x + vT_\theta = \frac{1}{c \cos\theta} \,. \tag{17}$$

The initial condition for T is specified according to the initial condition (9):

$$T|_{z=0} = T(0, x, \theta) = 0 \,, \tag{18}$$

which is consistent with the initial condition (12).

Following the same argument as in [34], we use the boundary condition

$$\frac{\partial T}{\partial x} = \frac{1}{c \sin\theta} \,, \tag{19}$$

on $x = x_{\min}$ and $x = x_{\max}$, and

$$\frac{\partial T}{\partial \theta} = \left(\sin\theta \frac{\partial c}{\partial z} - \cos\theta \frac{\partial c}{\partial x}\right)^{-1} \,, \tag{20}$$

on $\theta = -\theta_{\max}$ and $\theta = \theta_{\max}$.

2.4. Amplitude. The amplitude of a ray can be computed according to the formula given in [49, 35]:

$$\tilde{A}(x,z;x_s,z_s) = \frac{1}{2\pi}\sqrt{\frac{c}{2}}\sqrt{|\nabla\tilde{T}\times\nabla\tilde{\psi}|}\ , \tag{21}$$

where $\tilde{T}$ and $\tilde{\psi}$ are, respectively, the travel-time and the takeoff angle of a ray reaching (x,z) from (x_s,z_s). Travel-times and takeoff angles are well defined on each solution branch in the physical space (x,z). To compute this quantity in the reduced phase space, we consider T as the extension of $\tilde{T}$ to the phase space; furthermore, we may also extend $\tilde{\psi}$ and $\tilde{A}$ to ψ and A in the (z,x,θ) space, respectively. Since the takeoff angle is constant along a given ray in the phase space, we have

$$\psi_z + u\psi_x + v\psi_\theta = 0\,. \tag{22}$$

Moreover, we have

$$\begin{aligned}
\frac{\partial\tilde{T}}{\partial x} &= T_x + T_\theta\frac{\partial\theta}{\partial x}\,,\\
\frac{\partial\tilde{T}}{\partial z} &= T_z + T_\theta\frac{\partial\theta}{\partial z}\,,\\
\frac{\partial\tilde{\psi}}{\partial x} &= \psi_x + \psi_\theta\frac{\partial\theta}{\partial x}\,,\\
\frac{\partial\tilde{\psi}}{\partial z} &= \psi_x + \psi_\theta\frac{\partial\theta}{\partial z}\,.
\end{aligned} \tag{23}$$

It follows that

$$\left|\nabla\tilde{T}\times\nabla\tilde{\psi}\right| = \left|(T_x\psi_\theta - T_\theta\psi_x)\left(-v + u\frac{\partial\theta}{dx} + \frac{\partial\theta}{\partial z}\right) - \frac{1}{c\cos\theta}\left(\psi_x + \psi_\theta\frac{\partial\theta}{\partial x}\right)\right|\,. \tag{24}$$

Since $\phi(z,x,\theta(x,z)) = 0$ on the zero level set and equation (5), we have

$$\frac{\partial\theta}{\partial x} = -\frac{\partial\phi/\partial x}{\partial\phi/\partial\theta} \ \text{ and } \ \frac{\partial\theta}{\partial z} = v - u\frac{\partial\theta}{\partial x}\,. \tag{25}$$

Finally we have

$$A(z;x,\theta) = \frac{1}{2\pi}\sqrt{\frac{c}{2\cos\theta}}\sqrt{\left|\frac{\psi_x\phi_\theta - \psi_\theta\phi_x}{\phi_\theta}\right|}\,. \tag{26}$$

To compute the amplitude, we need the derivatives of the level set function at different z's. However, as illustrated in [34], reinitialization of the level set function is necessary and, unfortunately, this process would only keep the location of the zero level set unchanged. This means the derivatives of ϕ cannot be computed directly from the level set function by differentiating the function ϕ itself after the regularization procedure. Therefore, to compute the derivatives of the level set function on the zero level set, we need to advect those derivatives as well. We first let $\xi = \phi_x$ and $\eta = \phi_\theta$. Differentiating the advection equation for ϕ with respect to x and θ, respectively, we have

$$\begin{aligned}
&\xi_z + u\xi_x + v\xi_\theta + u_x\xi + v_x\eta = 0\,,\\
&\eta_z + u\eta_x + v\eta_\theta + u_\theta\xi + v_\theta\eta = 0\,.
\end{aligned} \tag{27}$$

We might apply the same idea to the advection equation for takeoff angles as well so that the derivatives of takeoff angles could be determined. Once those ingredients are in place, the amplitude could be obtained. However, we will not use this approach because this is not efficient computationally and it is difficult to reconcile the accuracies of four different derivatives numerically.

Instead, defining

$$\Delta = \psi_x \phi_\theta - \psi_\theta \phi_x \tag{28}$$

and differentiating it with respect to z, we have the advection equation

$$\Delta_z + \nabla_{x,\theta} \cdot (\mathbf{u}\Delta) = 0\,, \tag{29}$$

and the amplitude can then be computed by

$$A(z; x, \theta) = \frac{1}{2\pi} \sqrt{\left| \frac{c}{2\cos\theta} \frac{\Delta}{\phi_\theta} \right|}\,. \tag{30}$$

Therefore, we can simply solve the advection equations for ϕ_x, ϕ_θ and Δ in order to get the amplitude. We need the advection equation for ϕ_x because it is coupled with the one for ϕ_θ.

In general, Δ is a bounded quantity and ϕ_θ may approach zero. When ϕ_θ goes to zero, A goes to ∞ and we are approaching caustics.

Next we have to initialize the quantities that we are going to advect. At $z = 0$, we can set ϕ_x and ϕ_θ equal to 1 and 0, respectively. However, ψ_x is a singular function at the source and it is better to start computing ψ_x and ψ_θ at some $z = dz > 0$ close to zero. Assuming that the velocity c can be approximated by a constant near the source, we have

$$\begin{aligned} \psi_x(z, x, \theta)|_{z=dz} &= \frac{\cos^2\theta}{dz}\,, \\ \psi_\theta(z, x, \theta)|_{z=dz} &= 1\,. \end{aligned} \tag{31}$$

Thus at small $z = dz > 0$,

$$\phi(z, x, \theta)|_{z=dz} = x - dz \tan\theta\,, \tag{32}$$

and $\Delta(dz, x, \theta) \equiv -2$, which is independent of the dz as long as the velocity can be well approximated by a constant near the source. This makes the computation of amplitude stable.

2.5. Caustics. Mathematically, caustic surfaces are envelopes of the family of rays. In the geometrical optics, at a caustic the amplitude of the asymptotic expansion becomes infinite, so that the usual asymptotic expansion is no longer valid at caustics, and some special expansions have been introduced to construct wave fields near the caustics [27, 6].

In the current level set formulation, the caustic curves correspond to

$$\{(z, x) : \phi(z, x, \theta(x, z)) = 0 \text{ and } \phi_\theta(z, x, \theta(x, z)) = 0\}.$$

On the zero level set, when $\phi_\theta \neq 0$, there exists local unique projection onto the x-space; when $\phi_\theta = 0$, the corresponding singularities or catastrophes of the implicit function $\theta = \theta(z, x)$ for fixed z's are identified as caustics [43].

3. Implementation.

3.1. Local level set method. The computational complexity of the algorithm in [34] can be improved dramatically by implementing a narrow banding [1] or PDE-based version [32] of local level set method. In this work we adopt a version similar to the one proposed in [32].

Because we are interested only in the zero level set, all the updates can actually be done in a tube centered at $\phi = 0$. The radius of this tube, γ, is picked to be $5\Delta x$, due to the fact that five grid values are needed for the fifth order weighted essentially nonoscillatory (WENO) scheme when solving the advection equations. Therefore, by only considering the grid points within this tube, the complexity of the algorithm in [34] can be reduced by a factor of N from $O(N^3 \mathrm{Log} N)$ to $O(N^2 \mathrm{Log} N)$.

Reinitialization and orthogonalization are the two core ingredients in the local level set method as proposed in [32]. Reinitialization restores the equally spaced property for the level sets. The usual way is to make ϕ a signed distance function without moving the zero level set of ϕ appreciably. To achieve this, one can solve the following equation to steady state $\tilde{\phi}_\infty$ [38, 50, 9, 16, 32, 30]:

$$\frac{\partial \tilde{\phi}}{\partial \xi} + S(\phi)(|\nabla \tilde{\phi}| - 1) = 0 \tag{33}$$

$$\tilde{\phi}|_{\xi=0} = \phi(z, ., .)\,. \tag{34}$$

Here $S(\phi)$ could be the smoothed signum function [32]

$$S(\phi) = \frac{\phi}{\sqrt{\phi^2 + |\nabla \phi|^2 \Delta x \Delta \theta}}\,, \tag{35}$$

where Δx and $\Delta \theta$ are the mesh sizes along x- and θ-directions, respectively. However, we usually only need to evolve equation (33) for a few steps. How often we should invoke the reinitialization step is a subtle issue; see [32, 30] for some discussions. Because we are only interested in the value of T where $\phi = 0$, we apply the following orthogonalization procedure [9, 16, 32, 30]:

$$\frac{\partial \hat{T}}{\partial \xi} + \mathrm{sgn}(\phi)\left(\frac{\nabla \phi}{|\nabla \phi|} \cdot \nabla \hat{T}\right) = 0\,, \tag{36}$$

$$\hat{T}|_{\xi=0} = T(z, ., .) \tag{37}$$

which theoretically preserves the values of T where $\phi = 0$, but changes them elsewhere such that the new T does not vary too much near the desired region. At the steady state, $\nabla \phi \cdot \nabla \hat{T} = 0$.

3.2. Detecting caustics. Because passing through a caustic implies overturning of the zero level set in the x-θ space, the number of θ's such that $\phi(z, x, \theta(x)) = 0$ will increase or decrease by two when x varies monotonically. Therefore, a simple way to detect caustics is first enumerating the number of roots θ_k's for every x_i, then checking where those numbers have sudden jumps, and finally approximating locations of caustics by taking the midpoint of two adjacent x_i's which have different numbers of θ_k. The resulting approximation of the caustics locations is of first-order accuracy.

Let x_c^* be the exact location of a caustic at some fixed z^*. Assume that x_c is the computed caustic location obtained from the level set formulation, i.e., solving

$$\phi(z^*, x, \theta(x, z^*)) = 0, \tag{38}$$

$$\phi_\theta(z^*, x, \theta(x, z^*)) = 0. \tag{39}$$

Let x_c' be the approximate caustic location computed using our midpoint approximation. We have at least $x_c = x_c^* + O(\Delta x^2)$ which is delivered by the second-order scheme used here. Because the distance between x_c and x_c' is less than $\Delta x/2$, we have $x_c' = x_c + O(\Delta x)$ and, therefore, $x_c' = x^* + O(\Delta x)$.

In terms of multivalued travel-times, passing through a caustic implies that the number of travel-times increases or decreases by two as we will see in numerical examples.

3.3. Algorithm. Hence we have the following local level set algorithm for computing the geometrical optics terms.

I. Initialization.
   1. Given N_x and N_θ, determine
$$\Delta x = \frac{x_{\max} - x_{\min}}{N_z - 1} \text{ and } \Delta\theta = \frac{2\theta_{\max}}{N_\theta - 1}.$$
   2. Initialize ϕ, T, ϕ_x and ϕ_θ at $z = 0$.
   3. For each (x_i, θ_j), where $i = 1, \dots, N_x$ and $j = 1, \dots, N_\theta$, check if any of $|\phi(x_i, \theta_j)|$, $|\phi(x_{i-1}, \theta_j)|$, $|\phi(x_{i+1}, \theta_j)|$, $|\phi(x_i, \theta_{j-1})|$ or $|\phi(x_i, \theta_{j+1})|$ is less than γ. Collect all these points and call the set Γ.

II. March in z until $z = z_{\max}$.
   1. Determine Δz from the CFL condition.
   2. March one Δz step by solving the level set equation (10) in Γ.
   3. Reinitialize the level set function in Γ by solving (33).
   4. March one Δz step by solving the travel-time equation (17) in Γ.
   5. If $z \neq \Delta z$, update Δ by solving (29).
   6. Orthogonalize T, ϕ_x and ϕ_θ to ϕ respectively in Γ by solving (36).
   7. If $z \neq \Delta z$, orthogonalize Δ to ϕ in Γ. Otherwise, initialize Δ.
   8. Update the tube Γ.
   9. Detect caustics by checking if there is a change in the number of θ_k's which gives $\phi(z; x_i, \theta_k) = 0$ for two adjacent x_i's.

III. Output. For each x_i with $i = 1, \dots, N_x$,
   1. Determine all root θ_k such that $\phi(z_{\max}; x_i, \theta_k) = 0$ $(k = 1, \dots)$.
   2. Determine $T(z_{\max}; x_i, \theta_k)$ $(k = 1, \dots)$ by interpolation.
   3. Determine $\phi_x(z_{\max}, x_i, \theta_k)$, $\phi_\theta(z_{\max}; x_i, \theta_k)$ and $\Delta(z_{\max}; x_i, \theta_k)$ $(k = 1, \dots)$ by interpolation, and then compute $A(z_{\max}; x_i, \theta_k)$.

To determine Δz in step II.1, we only need to scan through the grid points in the computational tube Γ in order to determine the maxima of velocity fields u and v, and this takes $O(N\text{Log}N)$ steps.

In step II, the level set equation and the travel-time equation are decoupled and can be solved separately. The spacial derivatives are approximated by a fifth-order WENO-Godunov scheme [22] while a third-order total variation diminishing-Runge–Kutta (TVD-RK) method [31] is used for the time marching.

Unlike the global level set method, the reinitialization step in II.3 is used not only to regularize the level set function but also to ensure the location of the tube in step II.8 more accurate. This step is necessary here and the number of iterations is to be determined so that the information of the location of the zero level set is propagated by a distance larger than γ, the radius of the tube. Numerically, one or two steps per iteration in z would be enough to get a reasonably good solution. However, formally, let β be the CFL number used in the reinitialization and $m_{\min}$ be the minimum number of iterations of reinitialization; then we have

$$m_{\min} = \frac{\gamma}{\beta \min(\Delta x, \Delta\theta)} . \tag{40}$$

Because $\gamma = O(\Delta x)$, $m_{\min} = O(1)$. Overall the complexity of the reinitialization step for each Δz advancement would be equal to the number of grid points within the tube and is given by $O(N\mathrm{Log}N)$.

For the root-finding and the interpolation in step III, we can simply use any nonoscillatory interpolation scheme, for example, linear interpolation or ENO reconstruction.

The next issue is how to update the computational tube Γ. A simple way is to scan all grid points in the domain and to apply the same procedure as that in I.3. The complexity of the resulting method will be $O(N^2)$. However, because the motion of the zero level set is purely advective, zeros will not be generated outside the tube. We can, therefore, update the tube by only scanning through the boundary of Γ, and this requires only $O(N\mathrm{Log}N)$ operations.

As a result, for each iteration in z-direction, the complexity is $O(N\mathrm{Log}N)$. Because of the CFL condition, the number of iterations in z is of $O(N)$. Overall the complexity of this algorithm is only $O(N^2\mathrm{Log}N)$. Comparing to $O(N^4)$ as typically seen in the Lagrangian ray tracing method, this Eulerian method is highly efficient and attractive.

4. Numerical examples. For the first three examples, we put a point source at the origin and velocity functions $c(x, z)$ are all C^∞. The fourth example, the synthetic Marmousi model, is a very challenging one where the velocity function is given only as a gridded function. Unless specified, the computational domain we use in the following examples is chosen to be

$$\Omega_p = \{(x, \theta) : -1 \le x \le 1, \theta_{\max} \le \theta \le \theta_{\max}\} , \tag{41}$$

where $\theta_{\max} = 9\pi/20$. Accordingly, the Marmousi velocity will be rescaled to the above computational domain.

4.1. Constant model. The simplest case one can imagine is the constant velocity model with $c(x, z) = 1$. Figure 1 shows the exact and computed travel-times and amplitudes of the constant model. The solid lines are the exact solutions. The numerical solutions match well with the true values.

Tables 1 and 2 show the convergence results for this test case. Convergence history for travel-time solutions is shown in Table 1, while Table 2 contains that of the amplitudes. The results show second-order convergence for both travel-time solution and amplitude at $x = 1$.

In this simple case, we can also compute the solution where we assume the boundary $x = x_{\max}$ is reflective. When the zero level set hits the boundary $x = x_{\max}$ in the $x - \theta$ space, it will not leave the domain; instead, a new ray will be generated in the

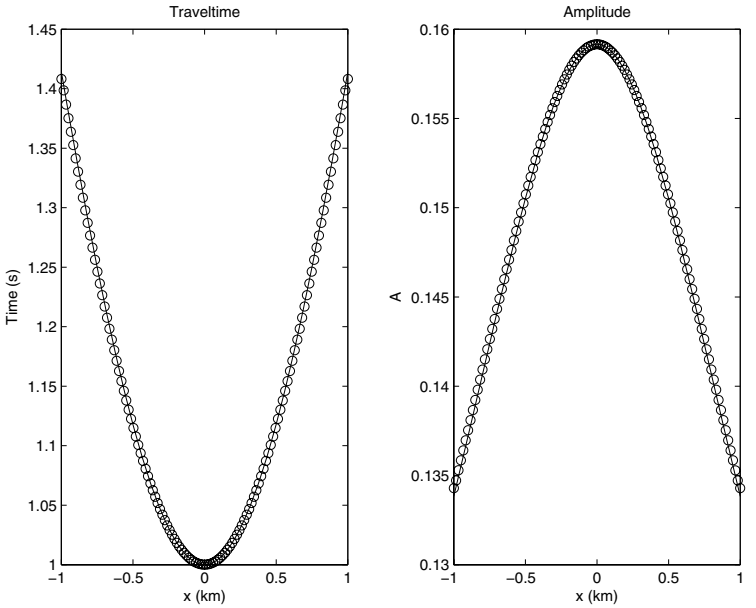


FIG. 1. *(Constant Model) Travel-times and amplitudes at $z = 1$ (km) using a* 120*-by-*120 *grid. Solid line: the exact solution. Circles: the solution using the local level set method.*

TABLE 1
(Constant Model) Accuracy and convergence order of travel-times at $z = 1$ km.

N_x	l_1 error	l_1 order	l_2 error	l_2 order	l_∞ error	l_∞ order
11	0.01044350		0.01077493		0.01504211	
21	0.00580110	0.8482	0.00540664	0.9948	0.00678677	1.1482
41	0.00193626	1.5830	0.00251217	1.5640	0.00251217	1.4337
81	0.00050419	1.9412	0.00047756	1.9369	0.00069198	1.8601
161	0.00011634	2.1156	0.00011220	2.0895	0.00017029	2.0227
321	0.00002791	2.0592	0.00002721	2.0435	0.00004292	1.9882

TABLE 2
(Constant Model) Accuracy and convergence order of amplitudes at $z = 1$ km.

N_x	l_1 error	l_1 order	l_2 error	l_2 order	l_∞ error	l_∞ order
11	0.00093010		0.00091941		0.00123784	
21	0.00052955	0.8126	0.00049836	0.8835	0.00061722	1.0039
41	0.00016404	1.6907	0.00015484	1.6863	0.00020976	1.5570
81	0.00004076	2.0086	0.00003870	2.0000	0.00005544	1.9195
161	0.00000970	2.0709	0.00000930	2.0568	0.00001391	1.9944
321	0.00000261	1.8924	0.00000253	1.8735	0.00000382	1.8623

computational space at the same $x = x_{\max}$ and with θ having opposite sign to the original ray.

For the travel-time, we have to determine boundary conditions for the incident and the reflected rays separately. We can use the same boundary condition as in the nonreflective case for the incident ray. The reflective ray has the same travel-time as the incident ray at the boundary $x = x_{\max}$.

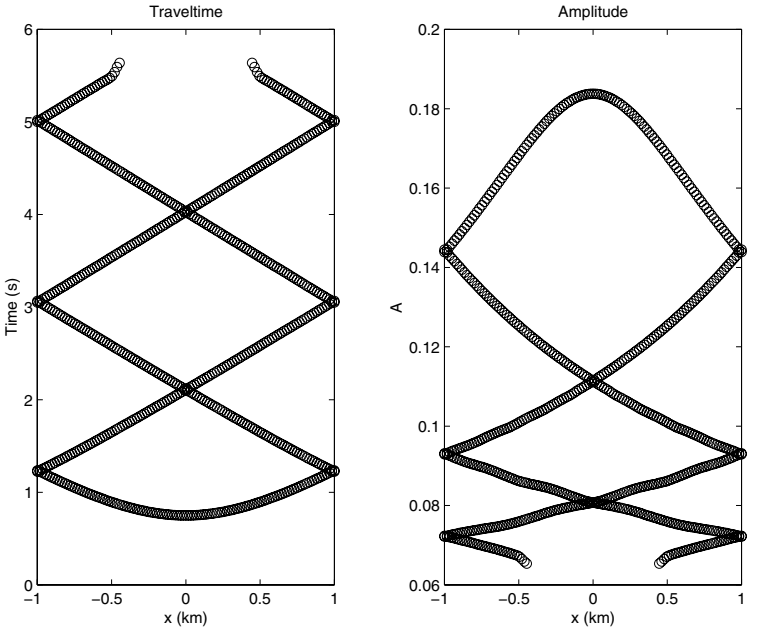


FIG. 2. *(Constant Model) Travel-time at $z = 0.75km$ with reflective boundary conditions.*

Therefore, we have the following boundary condition on $x = x_{\max}$:

$$
\begin{aligned}
\phi(x_{\max}, \theta) &= \phi(x_{\max}, -\theta) \\
\phi_x(x_{\max}, \theta) &= \phi_x(x_{\max}, -\theta) \\
\phi_\theta(x_{\max}, \theta) &= \phi_\theta(x_{\max}, -\theta) \\
\Delta(x_{\max}, \theta) &= \Delta(x_{\max}, -\theta) \\
T(x_{\max}, \theta) &= \begin{cases} \text{from (19)} & \text{if} \quad u(x_{\max}, \theta) > 0 \\ T(x_{\max}, -\theta) & \text{if} \quad u(x_{\max}, \theta) < 0 \end{cases}
\end{aligned} \tag{42}
$$

Similar boundary conditions can be used on the other side $x = x_{\min}$ and will not be fully discussed here. For the other two boundaries $\theta = \theta_{\max}$ and $\theta = -\theta_{\max}$, we can use the same boundary conditions as in the nonreflective case detailed in section 2.

Figure 2 shows the solutions obtained at $z = 1$km using the proposed method with reflective boundary conditions imposed on both boundaries $x = \pm 1$; multiple reflections are captured clearly as shown in travel-times and amplitudes using 120 grid points in both x- and θ-directions.

4.2. Waveguide model. The velocity function is given by

$$c(x, z) = 1.1 - \exp(-0.5x^2)\,. \tag{43}$$

The function is symmetric with respect to $x = 0$, and we also expect the same type of symmetry in both the travel-time and the amplitude.

Figure 3 shows travel-times, amplitudes and some intermediate quantities at $z =$1.6km using 240-by-240 grid points, respectively. The solid lines in the travel-time and amplitude plots are obtained using a ray tracing method. The solutions are symmetric as expected. Δ in this velocity model should be equal to -2 everywhere.

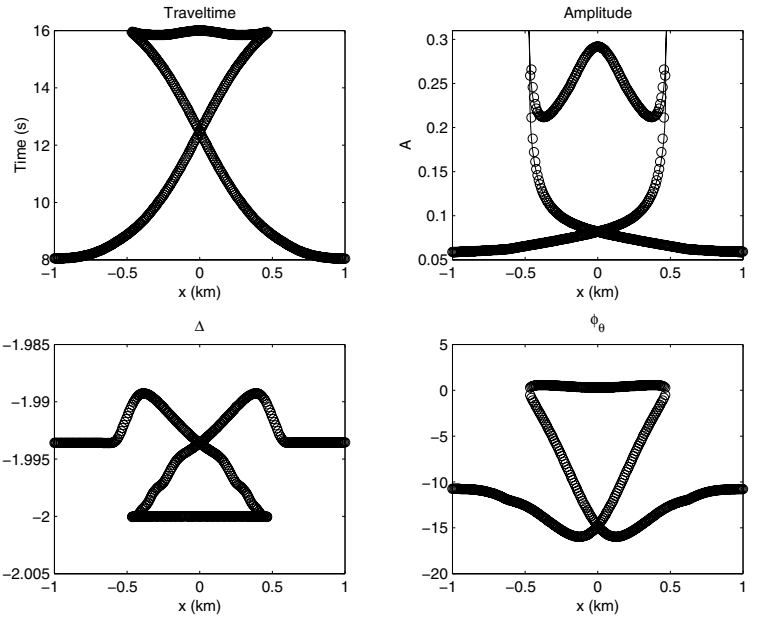


FIG. 3. *(Waveguide Model) Travel-time, amplitude,* Δ *and* ϕ_θ *at* $z = 1.6km$ *using a* 240*-by-*240 *grid.*

As we mentioned earlier, the velocity model is approximated by a constant near the source and this gives the initial condition $\Delta = -2$ at $z = dz$. For this waveguide model, because $u_x = v_\theta = 0$, the equation for Δ is purely advective; therefore the exact solution is $\Delta(z; x, \theta) = -2$, which is independent of z. The variations in the subplot of Δ in Figure 3 are due to numerical errors.

The singularities in the amplitudes, shown in the upper right subfigure in Figure 3, come from the vanishing of ϕ_θ on the zero level set of ϕ at around $x = \pm 0.45$km, shown in the lower right subfigure in Figure 3.

The caustic curves detected by the local level set method are shown in Figure 4: circles are computed locations of the caustics in the waveguide model, and the solid lines are the rays emanating from the source by a ray tracing method. The caustic locations are exactly those places where rays form an envelope as seen in the figure.

To demonstrate the improvement in the computational efficiency of the current formulation, we computed the solutions of this test case on a Pentium IV desktop with different sets of grid points and recorded the computational time. Table 3 shows the dramatic improvement delivered by the $O(N^2 \mathrm{Log} N)$ local level set method proposed here and the typical behavior of $O(N^3 \mathrm{Log} N)$ complexity offered by the global level set method.

4.3. Sinusoidal model. This example is adapted from the sinusoidal waveguide model used in [40, 41], and the velocity function is given by

$$c(x, z) = 1 + 0.2 \sin(0.5\pi z) \sin[3\pi(x + 0.55)] \,. \tag{44}$$

Figure 5 shows travel-times, amplitudes and some intermediate quantities at z =2.0km using 240-by-240 grid points. The subplot for travel-times in Figure 5 shows that the triplications in the travel-time developed at $z = 2.0$ are clearly captured by the level set Eulerian method. Singularities in the amplitude come from the

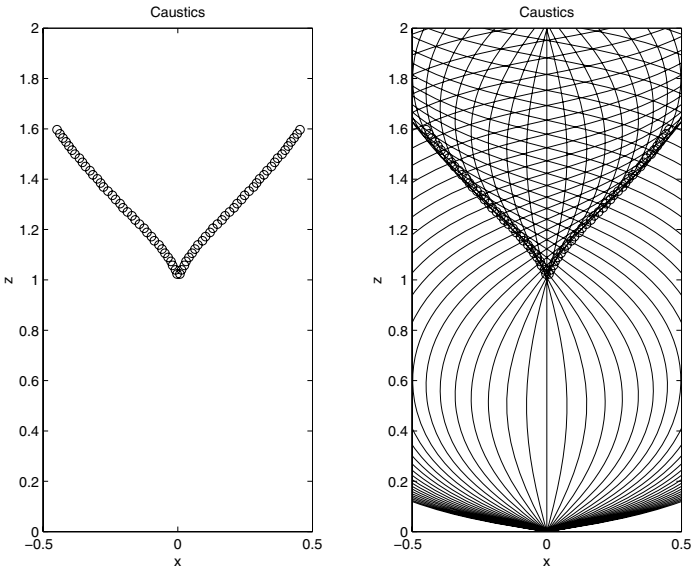


FIG. 4. *(Waveguide Model) Location of caustics and some rays from a ray tracing method. Caustics are determined by the local level set method with a* 360*-by-*360 *grid in the* x*-*θ *space.*

TABLE 3
(Waveguide Model) Time comparison for global and local level set methods: evolving up to $z = 1.6$.

Grid	Global: minutes	Local: minutes
120×120	72	8
240×240	641	19
480×480	5439	58

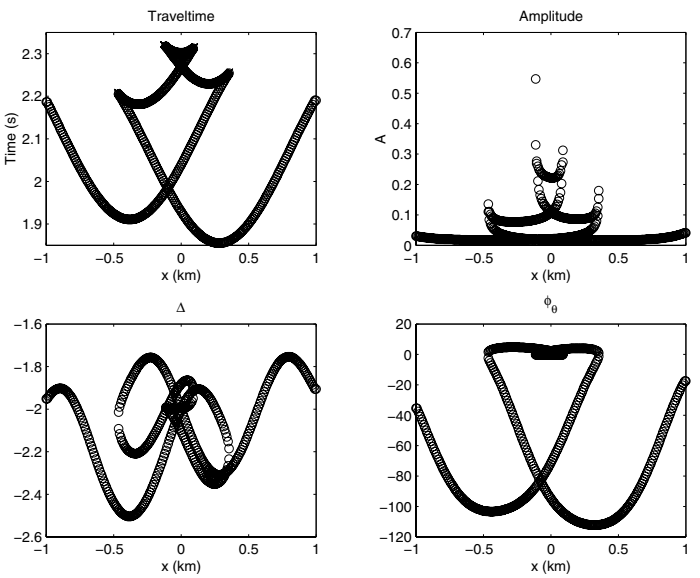


FIG. 5. *(Sinusoidal Model) Travel-time, amplitude,* Δ *and* ϕ_θ *at* $z = 2.0km$ *using a* 240*-by-*240 *grid.*

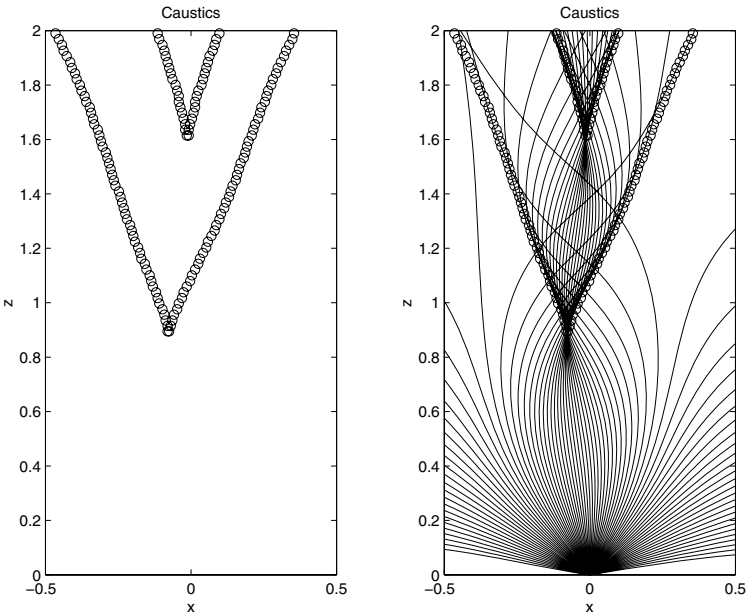


FIG. 6. *(Sinusoidal Model) Location of caustics and some rays from a ray tracing method. Caustics are determined by the local level set method with a* 360*-by-*360 *grid in the* x*-*θ *space.*

overturning of the zero level set in the phase space, i.e., $\phi_\theta = 0$, which is shown in the lower right subfigure in Figure 5.

Figure 6 shows locations of the caustics detected by the proposed method, and the results match those from the ray tracing method.

4.4. Synthetic Marmousi model. This example is the Marmousi model from the 1996 INRIA Workshop on Multi-arrival Travel-times. The calibration data used here were computed by Dr. Klimes and can be found at http://www.caam.rice.edu/~benamou/traveltimes.html. This is a synthetic model which will challenge the level set method used here.

The original Marmousi model is sampled on a 24m-by-24m grid, consisting of 384 samples in the x-direction and 122 samples in the z-direction; therefore the model dimension is 9.192km long in the x-direction and 2.904km deep in the z-direction. In the computational results presented here, we use a portion of Marmousi model, i.e., a window from 2.64km to 9.36km in the x-direction and from 0km to 2.904km in the z-direction. The source is located at x=6.0km and z=2.8km. The purpose is to compute (possibly multivalued) travel-times for those sampling points, i.e., the receivers from 160 to 340 on the surface z=0.0km. It should be emphasized that since we are using a local level set method, we can choose such a large window of the velocity model. In the example presented in [34], we were able to deal with a much smaller portion of the velocity model because we used a global level set method.

As illustrated in [34], to resolve a complicated wave front like the one generated by the Marmousi model, we have to use very fine computational meshes which need huge memory storage in the global level set setup. In the previous work [34], we applied up to 400 grid points in the x-direction to resolve the solutions between receiver 200 to 300. With the local level set method developed here, we are able to tackle a much

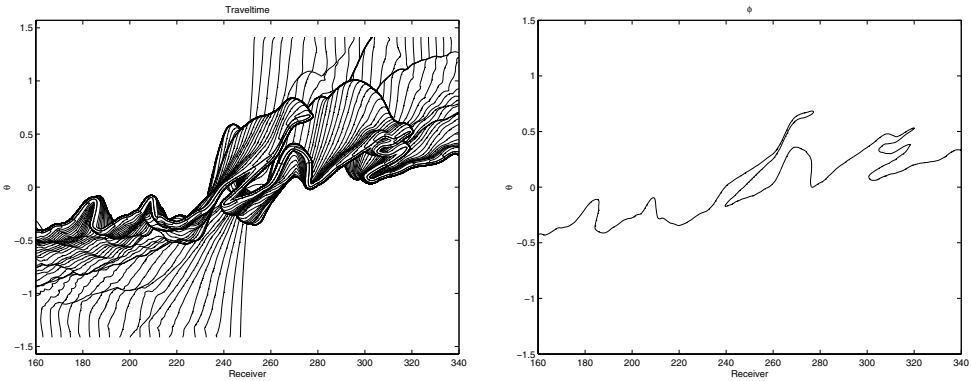


FIG. 7. *(Marmousi Model) Contours of the travel-time and the zero level set at* $z = 0.0km$.

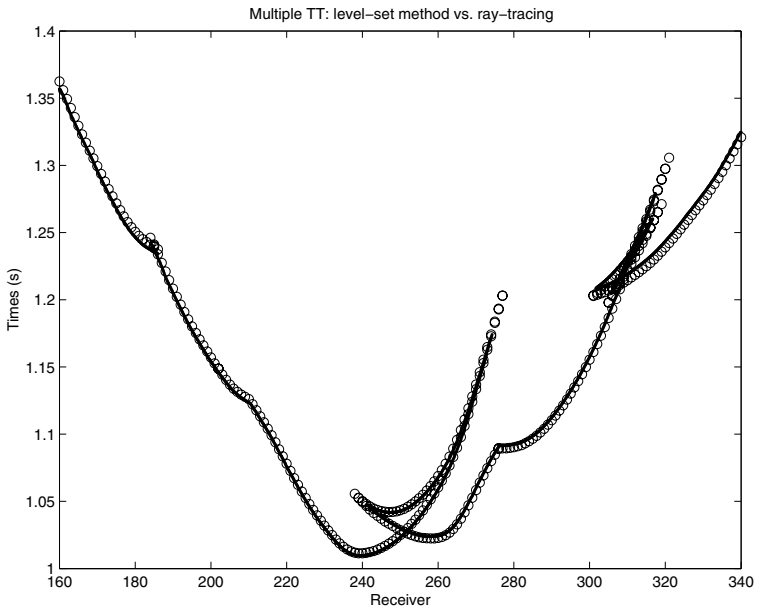


FIG. 8. *(Marmousi Model) Comparison between travel-times using a ray tracing method and the local level set method at* $z = 0.0km$.

larger portion of the original velocity and produce higher resolution travel-times with reasonable memory and computational cost. Figure 7 shows the zero level set and its overlay on the travel-time field by using a refinement of the original model with a refinement ratio = 8, i.e., we are using 1440 grid points in the x-direction to resolve the arrivals between receiver 160 to 340. As we can see from the plot, the zero level set has lots of overturnings and tiny tips. Therefore, there are lots of caustics developed in the wave propagation. The resulting travel-times at the surface along with the ray tracing solutions are shown in Figure 8; the two solutions match with each other reasonably well. Near receiver 340 we have captured five branches of travel-times as shown in Figure 9.

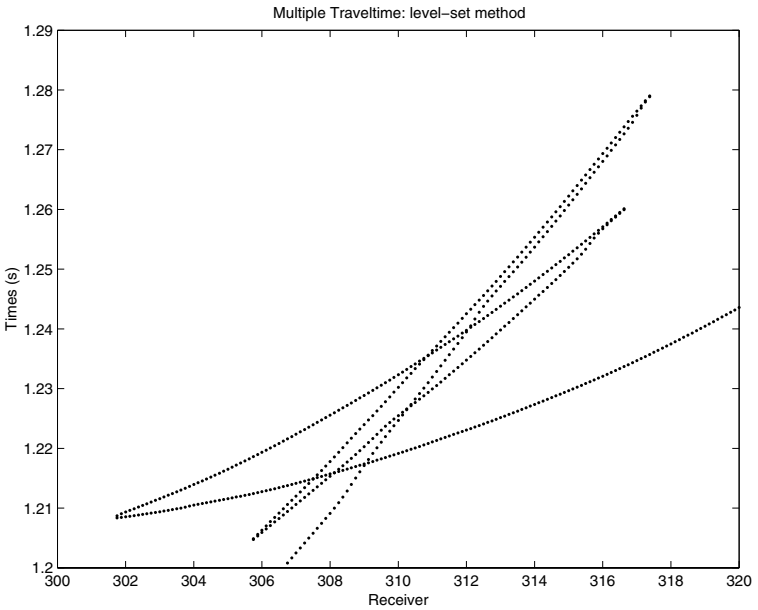


FIG. 9. *(Marmousi Model) Zoom-in at receiver* 310*: the local level set method at* $z = 0.0km$.

5. Conclusion. We have proposed a local level set method for paraxial geometrical optics, so that the multivalued travel-time and amplitude can be constructed with ease. The computational complexity of the algorithm is $O(N^2 \mathrm{Log} N)$, therefore it is highly efficient. Although the subhorizontal condition is required for the approach to work, it is suitable for many applications, such as reflection seismics, underwater acoustics and optics. The future work includes grafting this approach to high resolution seismic inversion [39] and computing the high frequency wave field using a time domain approach proposed in [6] rather than the usual frequency domain approach in [27].

Acknowledgment. J. Qian thanks Professors R. Burridge, S. Osher, F. Reitich and W. W. Symes for helpful discussions. This work was supported by ONR grant N00014-02-1-0720. J. Qian was partially supported by NSF DMS-0542174.

REFERENCES

[1] D. ADALSTEINSSON AND J. A. SETHIAN, *A fast level set method for propagating interfaces*, J. Comput. Phys., 118 (1995), pp. 269–277.
[2] J.-D. BENAMOU, *Direct solution of multi-valued phase-space solutions for Hamilton–Jacobi equations*, Comm. Pure Appl. Math., 52 (1999), pp. 1443–1475.
[3] J. D. BENAMOU, *An introduction to Eulerian geometrical optics* (1992–2002), J. Sci. Comput., 19 (2003), pp. 63–93.
[4] J.-D. BENAMOU AND I. SOLLIEC, *A Eulerian method for capturing caustics*, J. Comput. Phys., 162 (2000), pp. 132–163.
[5] M. BORN AND E. WOLF, *Principles of Optics*, Macmillan, New York, 1964.
[6] R. BURRIDGE, *Asymptotic evaluation of integrals related to time-dependent fields near caustics*, SIAM J. Appl. Math., 55 (1995), pp. 390–409.
[7] R. BURRIDGE, M. V. DE HOOP, D. MILLER, AND C. SPENCER, *Multiparameter inversion in anisotropic media*, Geophys. J. Internat., 134 (1998), pp. 757–777.

[8] V. Cerveny, I. A. Molotkov, and I. Psencik, *Ray Method in Seismology*, Univerzita Karlova Press, Praha, Czech Republic, 1977.
[9] S. Chen, B. Merriman, S. Osher, and P. Smereka, *A simple level set method for solving Stefan problems*, J. Comput. Phys., 135 (1997), pp. 8–29.
[10] L.-T. Cheng, H. Liu, and S. J. Osher, *High Frequency Wave Propagation in Schrodinger Equations Using the Level Set Method*, preprint 2003; available online at www.levelset.com.
[11] M. G. Crandall and P. L. Lions, *Viscosity solutions of Hamilton–Jacobi equations*, Trans. Amer. Math. Soc., 277 (1983), pp. 1–42.
[12] B. Engquist and O. Runborg, *Multi-phase computations in geometrical optics*, J. Comput. Appl. Math., 74 (1996), pp. 175–192.
[13] B. Engquist and O. Runborg, *Computational high frequency wave propagation*, Acta Numer., 12 (2003), pp. 181–266.
[14] B. Engquist, O. Runborg, and A-K Tornberg, *High frequency wave propagation by the segment projection method*, J. Comput. Phys., 178 (2002), pp. 373–390.
[15] L. C. Evans, *Towards a quantum analogue of weak KAM theory*, Comm. Math. Phys., 244 (2004), pp. 311–334.
[16] R. Fedkiw, T. Aslam, B. Merriman, and S. Osher, *A non-oscillatory Eulerian approach to interfaces in multimaterial flows (the ghost fluid mathod)*, J. Comput. Phys., 152 (1999), pp. 457–492.
[17] S. Fomel and J. Sethian, *Fast phase space computation of multiple traveltimes*, Proc. Natl. Acad. Sci., 99 (2002), pp. 7329–7334.
[18] S. Geoltrain and J. Brac, *Can we image complex structures with first-arrival traveltime*, Geophys., 58 (1993), pp. 564–575.
[19] L. Gosse, *Using K-branch entropy solutions for multivalued geometric optics computations*, J. Comput. Phys., 180 (2002), pp. 155–182.
[20] L. Gosse and P. A. Markowich, *Multiphase semiclassical approximations of an electron in a one-dimensional crystalline lattice*, J. Comput. Phys., 197 (2004), pp. 387–417.
[21] S. Gray and W. May, *Kirchhoff migration using eikonal equation traveltimes*, Geophys., 59 (1994), pp. 810–817.
[22] G. S. Jiang and D. Peng, *Weighted ENO schemes for Hamilton–Jacobi equations*, SIAM J. Sci. Comput., 21 (2000), pp. 2126–2143.
[23] S. Jin and X. Li, *Multi-phase computations of the semi-classical limit of the Schrodinger equation and related problems*, Phys. D, 182 (2003), pp. 46–85.
[24] S. Jin and S. Osher, *A Level Set Method for the Computation of Multivalued Solutions to Quasi-linear Hyperbolic PDEs and Hamilton-Jacobi Equations*, preprint 2003; available online at www.levelset.com.
[25] J. B. Keller and R. M. Lewis, *Asymptotic methods for partial differential equations: The reduced wave equation and Maxwell's equations*, Surv. Appl. Math., 1 (1995), pp. 1–82.
[26] P. L. Lions, *Generalized Solutions of Hamilton–Jacobi Equations*, Pitman, Boston, 1982.
[27] D. Ludwig, *Uniform asymptotic expansions at a caustic*, Comm. Pure Appl. Math., XIX (1966), pp. 215–250.
[28] S. Operto, S. Xu, and G. Lambare, *Can we image quantitatively complex models with rays*, Geophys., 65 (2000), pp. 1223–1238.
[29] S. Osher, L.-T. Cheng, M. Kang, H. Shim, and Y-H Tsai, *Geometric optics in a phase space based level set and Eulerian framework*, J. Comput. Phys., 179 (2002), pp. 622–648.
[30] S. Osher and R. P. Fedkiw, *Level Set Methods and Dynamic Implicit Surfaces*, Springer-Verlag, New York, 2003.
[31] S. J. Osher and C. W. Shu, *High-order essentially nonoscillatory schemes for Hamilton-Jacobi equations*, SIAM J. Numer. Anal., 28 (1991), pp. 907–922.
[32] D. Peng, B. Merriman, S. Osher, H. K. Zhao, and M. Kang, *A pde-based fast local level set method*, J. Comput. Phys., 155 (1999), pp. 410–438.
[33] J. Qian, L.-T. Cheng, and S. J. Osher, *A level set based Eulerian approach for anisotropic wave propagations*, Wave Motion, 37 (2003), pp. 365–379.
[34] J. Qian and S. Leung, *A level set method for paraxial multivalued traveltimes*, J. Comput. Phys., 197 (2004), pp. 711–736.
[35] J. Qian and W. W. Symes, *Adaptive finite difference method for traveltime and amplitude*, Geophys., 67 (2002), pp. 167–176.
[36] S. J. Ruuth, B. Merriman, and S. J. Osher, *A fixed grid method for capturing the motion of self-intersecting interfaces and related PDEs*, J. Comput. Phys., 151 (1999), pp. 836–861.
[37] J. Steinhoff, M. Fan, and L. Wang, *A new Eulerian method for the computation of propagating short acoustic and electromagnetic pulses*, J. Comput. Phys., 157 (2000), pp. 683–706.
[38] M. Sussman, P. Smereka, and S. J. Osher, *A level set approach for computing solutions to incompressible two-phase flows*, J. Comput. Phys., 114 (1994), pp. 146–159.

[39] W. W. Symes, *Mathematics of Reflection Seismology*, Annual Report, The Rice Inversion Project, Rice University, Houston, TX, 1995; also available online from http://www.trip.caam.rice.edu/.
[40] W. W. Symes, *A slowness matching finite difference method for traveltimes beyond transmission caustics*, Expanded Abstracts, in the 68th Annual International Meeting, Society of Exploration Geophysicists, 1998, pp. 1945–1948.
[41] W. W. Symes and J. Qian, *A slowness matching Eulerian method for multivalued solutions of eikonal equations*, J. Sci. Comput., 19 (2003), pp. 501–526.
[42] W. W. Symes, R. Versteeg, A. Sei, and Q. H. Tran, *Kirchhoff Simulation, Migration and Inversion Using Finite Difference Traveltimes and Amplitudes*, Annual Report, The Rice Inversion Project, Rice University, Houston, TX, 1994; also available online from http://www.trip.caam.rice.edu/.
[43] R. Thom, *Structural Stability and Morphogenesis*, Benjamin, New York, 1975.
[44] J. van Trier and W. W. Symes, *Upwind finite-difference calculation of traveltimes*, Geophys., 56 (1991), pp. 812–821.
[45] J. Vidale, *Finite-difference calculation of travel times*, Bull. Seismol. Soc. Amer., 78 (1988), pp. 2062–2076.
[46] V. Vinje, E. Iversen, and H. Gjystdal, *Traveltime and amplitude estimation using wavefront construction*, Geophys., 58 (1993), pp. 1157–1166.
[47] B. S. White, *The stochastic caustic*, SIAM J. Appl. Math., 44 (1984), pp. 127–149.
[48] B. S. White, *Wave action on currents with vorticity*, J. Fluid Mech., 386 (1999), pp. 329–344.
[49] L. Zhang, *Imaging by the wavefront propagation method*, Ph.D. thesis, Stanford University, Stanford, CA 94305, 1993.
[50] H.-K. Zhao, T. Chan, B. Merriman, and S. J. Osher, *A variational level set approach for multiphase motion*, J. Comput. Phys., 127 (1996), pp. 179–195.

SIAM J. SCI. COMPUT.
Vol. 28, No. 1, pp. 224–240
© 2006 Society for Industrial and Applied Mathematics

TOPOLOGICAL OPTIMIZATION OF THE EVALUATION OF FINITE ELEMENT MATRICES*

ROBERT C. KIRBY†, ANDERS LOGG‡, L. RIDGWAY SCOTT§, AND ANDY R. TERREL†

Abstract. We present a topological framework for finding low-flop algorithms for evaluating element stiffness matrices associated with multilinear forms for finite element methods posed over straight-sided affine domains. This framework relies on phrasing the computation on each element as the contraction of each collection of reference element tensors with an element-specific geometric tensor. We then present a new concept of *complexity-reducing relations* that serve as distance relations between these reference element tensors. This notion sets up a graph-theoretic context in which we may find an optimized algorithm by computing a minimum spanning tree. We present experimental results for some common multilinear forms showing significant reductions in operation count and also discuss some efficient algorithms for building the graph we use for the optimization.

Key words. finite element, variational form, optimized algorithm, minimum spanning tree

AMS subject classifications. 65N30, 05C90, 68R10

DOI. 10.1137/050635547

1. Introduction.

Several ongoing projects have led to the development of tools for automating important aspects of the finite element method, with the potential for increasing code reliability and decreasing development time. By developing libraries for existing languages or new domain-specific languages, these software tools allow programmers to define variational forms and other parts of a finite element method with succinct, mathematical syntax. Existing C++ libraries for finite elements include DOLFIN [6, 7], Sundance [16, 17], and deal.II [2]. These rely on some combination of operator overloading and object orientation to present a high-level syntax to the user. Other projects have defined new languages with their own grammar and syntax, such as FreeFEM [20], GetDP [5], and Analysa [1]. Somewhere between these two approaches is the FEniCS Form Compiler, FFC [13, 15], developed primarily by the second author. This Python library relies on operator overloading to define variational forms, but rather than using the Python interpreter to evaluate the forms, FFC generates low-level code that can be compiled into other platforms such as DOLFIN.

While these tools are effective at exploiting modern software engineering to produce workable systems, we believe that additional mathematical insight will lead to even more powerful codes with more general approximating spaces and more powerful algorithms. To give one example, work by the first author [9, 10] shows how the Ciarlét definition of the finite element leads to a code for arbitrary order elements of general type. This code, FIAT, is used by FFC and is currently being integrated with Sundance. For further examples, we refer the reader to work by the first three authors and Knepley (see [11, 12]), which study how to efficiently (in the sense of

*Received by the editors July 8, 2005; accepted for publication (in revised form) October 28, 2005; published electronically March 3, 2006.

http://www.siam.org/journals/sisc/28-1/63554.html

†Department of Computer Science, University of Chicago, Chicago, IL 60637-1581 (kirby@cs.uchicago.edu, aterrel@uchicago.edu). The first author's work was supported by the U.S. Department of Energy Early Career Principal Investigator Program under award DE-FG02-04ER25650.

‡Toyota Technological Institute at Chicago, University Press Building, 1427 East 60th St., Chicago, IL 60637-1581 (logg@tti-c.org).

§The Computation Institute and Departments of Computer Science and Mathematics, University of Chicago, Chicago, IL 60637-1581 (ridg@cs.uchicago.edu).

operation count) evaluate local stiffness matrices for finite element methods. For a given variational form, for each element e, there exists a "geometric tensor" G_e. Each entry of the $n \times n$ element stiffness matrix for e may be written as the contraction of a specific reference element tensor (depending only on the form, the polynomial basis, and the reference element) with G_e. We saw that there are relations between many of the reference element tensors (equality, colinearity, small Hamming distance) that may be exploited to find an algorithm with significantly fewer floating point operations than standard techniques.

In [11], we devised a crude algorithm that searched for and exploited these dependencies, generating simple Python code for evaluating the local stiffness matrix for the Laplacian on triangles given the geometric tensor. In this paper, we set the optimization process in a more abstract graph/topological context. In section 2, we express the calculation for some finite element operators as tensor contractions. These ideas are more fully developed in [13], where the first two authors show how this tensor structure may be used to implement a compiler capable of generating the reference tensors, and hence code, for building the stiffness matrices for general multilinear forms on affine elements. Further, a generalization to nonaffine mappings, including curvilinear elements, is suggested. In section 3, we introduce the idea of [illegible], which are a kind of distance relations among the tensors that serve to model the cost of complexity in computing the contractions. Using these ideas, we show how to derive an algorithm for performing the computation that is optimal in a certain sense. We demonstrate the reduction in operation count for several finite element operations in section 4. After this, we show how these efficient algorithms may be derived more efficiently through the use of certain search procedures in section 5. Finally, we state some conclusions and remarks about ongoing work in section 6.

2. Finite element formulation. The finite element method is a general methodology for the discretization of differential equations. A linear (or linearized) differential equation for the unknown function u is expressed in the form of a canonical variational problem, and the discrete approximation U of u is sought as the solution of a discrete version of the following variational problem [4, 3]: Find $U \in V$ such that

$$a(v, U) = L(v) \quad \forall v \in \hat{V}, \tag{1}$$

where $(\hat{V}, V)$ is a pair of suitable discrete (typically piecewise polynomial) function spaces, $a : \hat{V} \times V \to \mathbb{R}$ a bilinear form, and $L : \hat{V} \to \mathbb{R}$ a linear form.

The variational problem (1) corresponds to a linear system $A\xi = b$ for the expansion coefficients $\xi \in \mathbb{R}^M$ of the discrete function U in a basis $\{\varphi_i\}_{i=1}^M$ for V. If $\{\hat{\varphi}_i\}_{i=1}^M$ is the corresponding basis for $\hat{V}$, the entries of A and b are given by $A_{ij} = a(\hat{\varphi}_i, \varphi_j)$ and $b_i = L(\hat{\varphi}_i)$, respectively. When we consider general multilinear forms below, the multilinear form a is represented by a [illegible] A.

2.1. Multilinear forms. Let now $\{V_i\}_{i=1}^r$ be a given set of discrete function spaces defined on a triangulation $\mathcal{T} = \{e\}$ of $\Omega \subset \mathbb{R}^d$. We consider a general multilinear form a defined on the product space $V_1 \times V_2 \times \cdots \times V_r$:

$$a : V_1 \times V_2 \times \cdots \times V_r \to \mathbb{R}. \tag{2}$$

Typically, $r = 1$ (linear form) or $r = 2$ (bilinear form), but forms of higher arity appear frequently in applications and include variable coefficient diffusion and advection of momentum in the incompressible Navier–Stokes equations.

Let $\{\varphi_i^1\}_{i=1}^{M_1}, \{\varphi_i^2\}_{i=1}^{M_2}, \ldots, \{\varphi_i^r\}_{i=1}^{M_r}$ be bases of $V_1, V_2, \ldots, V_r$, and let $i = (i_1, i_2, \ldots, i_r)$ be a multi-index. The multilinear form a then defines a rank r tensor given by

$$A_i = a(\varphi_{i_1}^1, \varphi_{i_2}^2, \ldots, \varphi_{i_r}^r). \tag{3}$$

In the case of a bilinear form, the tensor A is a matrix (the stiffness matrix), and in the case of a linear form, the tensor A is a vector (the load vector).

To compute the tensor A by assembly, we need to compute the [illegible] A^e on each element e of the triangulation $\mathcal{T}$ of Ω [13]. Let $\{\varphi_i^{e,1}\}_{i=1}^{n_1}$ be the restriction to e of the subset of $\{\varphi_i^1\}_{i=1}^{M_1}$ supported on e, and define the local bases on e for $V_2, \ldots, V_r$ similarly. The rank r element tensor A^e is then given by

$$A_i^e = a_e(\varphi_{i_1}^{e,1}, \varphi_{i_2}^{e,2}, \ldots, \varphi_{i_r}^{e,r}). \tag{4}$$

2.2. Evaluation by tensor representation. The element tensor A^e can be efficiently computed by representing A^e as a special tensor product. If the multilinear form a is given by an integral over the domain Ω, then each entry A_i^e of A^e is given by an integral over the element e. By a change of variables and a series of linear transformations, we may rewrite this integral as an integral over a reference element E. In particular, when the map from the reference element E is affine, the linear transformations of derivatives can be moved outside of the integral to obtain a representation of the element tensor A^e as a tensor product of a constant tensor A^0 and a tensor G_e that varies over the set of elements,

$$A_i^e = A_{i\alpha}^0 G_e^\alpha, \tag{5}$$

or more generally a sum $A_i^e = A_{i\alpha}^{0,k} G_{e,k}^\alpha$ of such tensor products, where i and α are multi-indices and we use the convention that repetition of an index means summation over that index. We refer to A^0 as the [illegible] and to G_e as the [illegible]. The rank of the reference tensor is the sum of the rank $r = |i|$ of the element tensor and the rank $|\alpha|$ of the geometric tensor G_e. As we shall see, the rank of the geometric tensor depends on the specific form.

In [13], we present an algorithm that computes the tensor representation (5) for fairly general multilinear forms. This algorithm forms the foundation for FFC [15].

As an example, we consider here the tensor representation of the element tensor A^e for Poisson's equation $-\Delta u(x) = f(x)$ with homogeneous Dirichlet boundary conditions on a domain Ω. The bilinear form a is given here by $a(v,u) = \int_\Omega \nabla v(x) \cdot \nabla u(x)\,\mathrm{d}x$, and the linear form L is given by $L(v) = \int_\Omega v(x) f(x)\,\mathrm{d}x$. By a change of variables using an affine map $F_e : E \to e$, we obtain the following representation of the element tensor A^e:

$$\begin{aligned} A_i^e &= \int_e \nabla\varphi_{i_1}^{e,1}(x) \cdot \nabla\varphi_{i_2}^{e,2}(x)\,\mathrm{d}x \\ &= \det F_e' \frac{\partial X_{\alpha_1}}{\partial x_\beta} \frac{\partial X_{\alpha_2}}{\partial x_\beta} \int_E \frac{\partial \Phi_{i_1}^1(X)}{\partial X_{\alpha_1}} \frac{\partial \Phi_{i_2}^2(X)}{\partial X_{\alpha_2}}\,\mathrm{d}X = A_{i\alpha}^0 G_e^\alpha, \end{aligned} \tag{6}$$

where $A_{i\alpha}^0 = \int_E \frac{\partial \Phi_{i_1}^1(X)}{\partial X_{\alpha_1}} \frac{\partial \Phi_{i_2}^2(X)}{\partial X_{\alpha_2}}\,\mathrm{d}X$, $G_e^\alpha = \det F_e' \frac{\partial X_{\alpha_1}}{\partial x_\beta} \frac{\partial X_{\alpha_2}}{\partial x_\beta}$, and $\Phi_{i_j}^j = \varphi_{i_j}^{e,j} \circ F_e$ for $j = 1, 2$. We see that the reference tensor A^0 is here a rank four tensor and the geometric tensor G_e is a rank two tensor. In [11], we saw that dependencies frequently occur between A_i^0 and A_j^0 for many multi-indices i, j that can reduce the overall cost of computation.

3. Optimizing stiffness matrix evaluation.

3.1. An abstract optimization problem. Tensors of arbitrary rank may be rewritten as vectors, with contraction becoming the Euclidean inner product. So, we will deal only with vectors and inner products to formulate our abstract optimization problem. We let $Y = \{y^i\}_{i=1}^n$ be a collection of n vectors in $\mathbb{R}^m$. In the most general case, this is a collection rather than a set, as some of the items in Y may be identical. Corresponding to Y, we must find a process for computing for arbitrary $g \in \mathbb{R}^m$ the collection of items $\{(y^i)^t g\}_{i=1}^n$. Throughout this paper, we will measure the cost of this as the total number of multiply-add pairs (MAPs) required to complete all the dot products. This cost is always bounded by nm, but we hope to improve on that. This could be alternatively formalized as building an abstract control-flow graph for performing the dot products that is equivalent to the naive process but contains a minimal number of nodes. Our techniques, however, rely on structure that is not apparent to traditional optimizing compilers, so we prefer the present formulation.

We seek out ways of optimizing the local matrix evaluation that rely on notions of distance between a few of the underlying vectors. The Euclidean metric is not helpful here; we focus on other, discrete measures of distance such that if y and z are close together, then $y^t g$ is easy to compute once $z^t g$ is known (and vice versa). Many of the dependencies we considered in [11] were between pairs of vectors—equality, colinearity, and Hamming distance. Here we develop a theory for optimizing the evaluation of finite element matrices under binary relations between the members of the collection. This starts by introducing some notions of distance on the collection of vectors and finds an optimized computation with respect to those notions by means of a minimum spanning tree.

3.2. Complexity-reducing relations.

DEFINITION 1. [illegible] $\rho : Y \times Y \to [0, m]$ [illegible] ρ [illegible] complexity-reducing [illegible] $y, z \in Y$ [illegible] $\rho(y, z) \le k < m$ [illegible] $y^t g$ [illegible] $z^t g$ [illegible] k [illegible]

EXAMPLE 1. [illegible] $e^+(y, z) = d(1 - \delta_{y,z})$ [illegible] $\delta_{y,z}$ [illegible] e^+ [illegible] $e^+(y, z) = 0$ [illegible] $y^t g = z^t g$ [illegible] $g \in \mathbb{R}^m$ [illegible] $e^-(u, v) = e^+(u, -v)$ [illegible] $u = -v$ [illegible] $u^t g$ [illegible] $v^t g$ [illegible]

EXAMPLE 2. [illegible]

$$c(y, z) = \begin{cases} 0, & y = z, \\ 1, & y = \alpha z \text{ [illegible] } \alpha \in \mathbb{R}, \quad \alpha \neq 0, 1, \\ m & \text{otherwise.} \end{cases} \tag{7}$$

[illegible] c [illegible] $y^t g = (\alpha z)^t g = \alpha(z^t g)$ [illegible] $y^t g$ [illegible] $z^t g$ [illegible]

EXAMPLE 3. [illegible] $H^+(y, z)$ [illegible] y [illegible] z [illegible] H^+ [illegible] $H^+(y, z) = k$ [illegible] y [illegible] z [illegible] k [illegible] $y - z$ [illegible] k [illegible] $(y - z)^t g$ [illegible] k [illegible] $y^t g = (y - z)^t g + z^t g$ [illegible] $H^-(y, z) = H^+(y, -z)$

THEOREM 1. [illegible] ρ_1 [illegible] ρ_2 [illegible]

$$\rho(y, z) = \min(\rho_1(y, z), \rho_2(y, z)). \tag{8}$$

[illegible] ρ [illegible]

Pick $y, z \in Y$, let $1 \leq i \leq 2$ be such that $\rho(y, z) = \rho_i(y, z)$, and let $\rho_i(y, z) \equiv k$. But ρ_i is a complexity-reducing relation, so for any $g \in \mathbb{R}^m$, $y^t g$ may be computed in no more than $k = \rho(y, z)$ MAPs. Hence ρ is complexity-reducing. ∎

This simple result means that we may consider any finite collection of complexity-reducing relations (e.g., colinearity together with Hamming distance) as if they were a single relation for the purpose of finding an optimized computation in the later discussion.

DEFINITION 2. [illegible] composite [illegible] simple

To see that not all complexity-reducing relations are metrics, it is easy to find a composite complexity-reducing that violates the triangle inequality. Let $\rho(y, z) = \min(H^+(y, z), c(y, z))$. It is not hard to show that H^+ and c are both metrics. If we take $y = (1, 2, 2)^t$ and $z = (0, 4, 4)^t$, then $\rho(y, z) = 3$ since the vectors are neither colinear nor do they share any common entries. However, if we let $x = (0, 2, 2)^t$, then $\rho(y, x) = 1$ since the vectors share two entries and $\rho(x, z) = 1$ by colinearity. Hence, $\rho(y, z) > \rho(y, x) + \rho(x, z)$, and the triangle inequality fails.

REMARK 1. [illegible] Steiner points [illegible] $\{(1, 2, 2)^t, (0, 4, 4)^t\}$ [illegible] $(0, 2, 2)^t$ [illegible] g [illegible]

3.3. Finding an optimized computation. Having defined complexity-reducing relations and given several examples, we now show how they may be used to determine an optimized evaluation of the stiffness matrix. We shall work in the context of a single complexity-reducing relation ρ, which may be composite.

In order to compute $\{(y^i)^t g\}_{i=1}^n$, we would like to pick some $y^i \in Y$ and compute $(y^i)^t g$. Then we want to pick some y^j that is very close to y^i under ρ and then compute $(y^j)^t g$. Then we pick some y^k close to either y^i or y^j and compute that dot product. So, this process continues until all the dot products have been computed. Moreover, since the vectors Y depend only on the variational form and finite element space and not the mesh or parameters, it is possible to do this search once offline and generate low-level code that will exploit these relations. We first formalize the notion of finding the optimal computation and then how the code may be generated.

We introduce a weighted, undirected graph $G = (Y, E)$, where Y is our collection of vectors defined above. Our graph is completely connected; that is, every pair of vectors y^i, y^j are connected by an edge. The weight of this edge is defined to be $\rho(y^i, y^j)$. We may think of walking along the edge from y^i to y^j as using the dot product $(y^i)^t g$ to compute $(y^j)^t g$. If ρ is composite, it will be helpful to know later which underlying relation gave the minimum value used for ρ. So, suppose that $\rho(y, z) = \min\{\rho_i(y, z)\}_{i=1}^R$. For any fixed y, z, let $\varrho(y, z)$ be a in integer in $[1, R]$ such that $\rho_{\varrho(y,z)}(y, z) = \rho(y, z)$. In addition to weights, we thus associate with each edge $\{y^i, y^j\}$ the entity $\varrho(y^i, y^j)$.

A standard graph-theoretic object called a [illegible] is exactly what we need [14]. A spanning tree, which we shall denote (Y, E'), is a subgraph that satisfies certain properties. First, it contains all of the n nodes of the original graph. Second, (Y, E') is connected. Finally, E' has $n - 1$ edges, so that there are no cycles

(thus it is a tree). Now there are possibly many spanning trees for a given graph. Every spanning tree has a [illegible] associated with it that is the sum of the weights of its edges. A [illegible] spanning tree is a spanning tree such that the sum of the edge weights is as small as possible. Minimum spanning tree algorithms start with a particular node of the graph, called the [illegible]. Regardless of which root is chosen, minimum spanning tree algorithms will generate trees with exactly the same sum of edge weights.

While technically a minimum spanning tree is undirected, we can think of it as being a directed graph with all edges going away from the root. Such a notion tells us how to compute all of the dot products with minimal operations with respect to ρ. We start with the root node, which we assume is y^0, and compute $(y^0)^t g$. Then, for each of the children of y^0 in the tree, we compute the dot products with g using the result of $(y^0)^t g$. Then we use the dot products of the children to compute the dot products of each of the children's children, and so on. This is just a breadth-first traversal of the tree. A depth-first traversal of the tree would also generate a correct algorithm, but it would likely require more motion of the computed results in and out of registers at runtime.

The total number of MAPs in this process is m for computing $(y^0)^t g$ plus the sum of the edge weights of (Y, E'). As the sum of edge weights is as small as it can be, we have a minimum-cost algorithm for computing $\{(y^i)^t g\}_{i=1}^n$ with respect to ρ for any $g \in \mathbb{R}^m$. On the other hand, it is not a true optimal cost, as one could find a better ρ or else use relations between more than two vectors (say, three coplanar vectors). One other variation is in the choice of root vector. If, for example, some y^i has several elements that are zero, then it can be dotted with g with fewer than m MAPs. Hence, we pick some $\bar{y} \in Y$ such that the number of nonzero entries is minimal to be the root. We summarize these results in a theorem.

THEOREM 2. [illegible] $G = (Y, E)$ [illegible] $g \in \mathbb{R}^m$ [illegible] $\{(y^i)^t g\}_{i=1}^n$ [illegible] $m' + w$ [illegible] m' [illegible] Y [illegible] w [illegible] G

The overhead of walking through the tree at runtime would likely outweigh the benefits of reducing the floating point cost. We can instead traverse the tree and generate low-level code for computing all of the dot products—this function takes as an argument the vector g and computes all of the dot products of Y with g. An example of such code was presented in [11].

3.4. Comparison to spectral elements. Our approach is remarkably different from the spectral element method. In spectral element methods, one typically works with tensor products of Lagrange polynomials over logically rectangular domains. Efficient algorithms for evaluating the stiffness matrix or its action follow naturally by working dimension-by-dimension. While such decompositions are possible for unstructured shapes [8], these are restricted to specialized polynomial bases. On the other hand, our approach is blind to both the element shape and the kind of approximating spaces used. While spectral element techniques may ultimately prove more effective when available, our approach will enable some level of optimization in more general cases, such as Raviart–Thomas–Nedelec [21, 22, 18, 19] elements on tetrahedra.

4. Experimental results. Here we show that this optimization technique is successful at generating low-flop algorithms for computing the element stiffness matrices associated with some standard variational forms. First, we consider the bilinear

TABLE 1
Element matrix indices and associated tensors (the slice A_i^0 for each fixed index i) displayed as vectors for the Laplacian on triangles with quadratic basis functions. All vectors are scaled by six, so they appear as integers.

Index	Vector			
(0, 0)	3	3	3	3
(0, 1)	1	0	1	0
(0, 2)	0	1	0	1
(0, 3)	0	0	0	0
(0, 4)	0	-4	0	-4
(0, 5)	-4	0	-4	0
(1, 0)	1	1	0	0
(1, 1)	3	0	0	0
(1, 2)	0	-1	0	0
(1, 3)	0	4	0	0
(1, 4)	0	0	0	0
(1, 5)	-4	-4	0	0

Index	Vector			
(2, 0)	0	0	1	1
(2, 1)	0	0	-1	0
(2, 2)	0	0	0	3
(2, 3)	0	0	4	0
(2, 4)	0	0	-4	-4
(2, 5)	0	0	0	0
(3, 0)	0	0	0	0
(3, 1)	0	0	4	0
(3, 2)	0	4	0	0
(3, 3)	8	4	4	8
(3, 4)	-8	-4	-4	0
(3, 5)	0	-4	-4	-8

Index	Vector			
(4, 0)	0	0	-4	-4
(4, 1)	0	0	0	0
(4, 2)	0	-4	0	-4
(4, 3)	-8	-4	-4	0
(4, 4)	8	4	4	8
(4, 5)	0	4	4	0
(5, 0)	-4	-4	0	0
(5, 1)	-4	0	-4	0
(5, 2)	0	0	0	0
(5, 3)	0	-4	-4	-8
(5, 4)	0	4	4	0
(5, 5)	8	4	4	8

forms for the Laplacian and advection in one coordinate direction for tetrahedra. Second, we study the trilinear form for the weighted Laplacian. In all cases, we generated the element tensors using FFC [13, 15], which in turn relies on FIAT [9, 10] to generate the finite element basis functions and integration rules. Throughout this section, we let $d = 2, 3$ refer to the spatial dimension of Ω.

4.1. Laplacian. We consider first the standard Laplacian operator

$$a(v, u) = \int_\Omega \nabla v(x) \cdot \nabla u(x) \, \mathrm{d}x. \tag{9}$$

We gave a tensor product representation of the local stiffness matrix in (6). The indices of the local stiffness matrix and the associated tensors are shown in Table 1.

Because the element stiffness matrix is symmetric, we need to build only the triangular part. Even without any optimization techniques, this naturally leads from computing $|P|^2$ contractions at a cost of d^2 MAPs each to $\binom{|P|+1}{2}$ contractions, where $|P|$ is the dimension of the polynomial space P. Beyond this, symmetry opens up a further opportunity for optimization. For every element e, G_e is symmetric. The equality of its off-diagonal entries means that the contraction can be performed in $\binom{d+1}{2}$ rather than d^2 entries. This is readily illustrated in the two-dimensional case. We contract a symmetric 2×2 tensor G with an arbitrary 2×2 tensor K:

$$\begin{aligned} G : K &= \begin{pmatrix} G_{11} & G_{12} \\ G_{12} & G_{22} \end{pmatrix} : \begin{pmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{pmatrix} \\ &= G_{11}K_{11} + G_{12}(K_{12} + K_{21}) + G_{22}K_{22} \\ &= \tilde{G}^t \hat{K}, \end{aligned} \tag{10}$$

where $\tilde{G}^t = (G_{11}, G_{12}, G_{22})$ and $\hat{K}^t = (K_{11}, K_{12} + K_{21}, K_{22})$.

This simple calculation implies a linear transformation $\hat{\cdot}$ from $\mathbb{R}^{d \times d}$ into $\mathbb{R}^{\binom{d+1}{2}}$ obtained by taking the diagonal entries of the matrix together with the sum of the off-diagonal entries, that may be applied to each reference tensor, together with an associated mapping $\tilde{\cdot}$ on symmetric tensors that takes only the symmetric part and casts it as a vector. Hence, the overall cost of computing an element stiffness matrix before optimizations goes from $|P|^2 d^2$ to $\binom{|P|+1}{2}\binom{d+1}{2}$.

TABLE 2
Element matrix indices and associated tensors (the slice $A^0_{i,:}$ for each fixed index i) for the Laplacian on triangles with quadratic basis functions after transformation to make use of symmetry.

Index	Vector		
(0, 0)	3	6	3
(0, 1)	1	1	0
(0, 2)	0	1	1
(0, 3)	0	0	0
(0, 4)	0	-4	-4
(0, 5)	-4	-4	0
(1, 1)	3	0	0
(1, 2)	0	-1	0
(1, 3)	0	4	0
(1, 4)	0	0	0
(1, 5)	-4	-4	0

Index	Vector		
(2, 2)	0	0	3
(2, 3)	0	4	0
(2, 4)	0	-4	-4
(2, 5)	0	0	0
(3, 3)	8	8	8
(3, 4)	-8	-8	0
(3, 5)	0	-8	-8
(4, 4)	8	8	8
(4, 5)	0	8	0
(5, 5)	8	8	8

TABLE 3
Number of MAPs in the optimized algorithm for computing the Laplacian element stiffness matrix on triangles and tetrahedra for Lagrange polynomials of degree one through three without using symmetry.

Triangles				
Degree	n	m	nm	MAPs
1	9	4	36	13
2	36	4	144	25
3	100	4	400	74

Tetrahedra				
Degree	n	m	nm	MAPs
1	16	9	144	43
2	100	9	900	205
3	400	9	3600	864

An interesting property of this transformation of the reference tensor is that it is [illegible] for the complexity-reducing relations we consider. The Hamming distance between two items under $\hat{\cdot}$ is bounded by the Hamming distance between the items. More precisely, $\rho(\hat{y}, \hat{z}) \leq \rho(y, z)$. Furthermore, if items are colinear before transformation, their images will be as well. So, for the optimizations we consider, we will not destroy any dependencies. Moreover, the transformation may introduce additional dependencies. For example, before applying the transformation, entries (0,1) and (1,5) are not closely related by Hamming distance or colinearity, as seen in Table 1. However, after the transformation, we see that the same items in Table 2 are colinear. Other examples can be readily found.

We optimized the evaluation of the Laplacian for Lagrange finite elements of degrees one through three on triangles and tetrahedra using a composite complexity-reducing relation with H^+, H^-, and c defined in Examples 2 and 3. We performed the optimization both with and without symmetry. The results are shown in Tables 3 and 4. Figure 1 shows a diagram of the minimum spanning tree computed by our code for the Laplacian on quadratic elements using symmetry. Each node of the graph is labeled with a pair (i, j) indicating the matrix entry (the vectors themselves are displayed in Table 2), and the edges are labeled with the associated weights. Simple inspection reveals that the sum of the edge weights is 14, which when added to 3 to compute the dot product for the root node agrees with the entry for quadratics in Table 4.

These techniques are successful in reducing the flop count, down to less than one operation per entry on triangles and less than two on tetrahedra for quadratic and cubic elements. We showed in [11] for low degree elements on triangles that going from standard numerical quadrature to tensor contractions led to a significant reduction

TABLE 4
Number of MAPs in the optimized algorithm for computing the Laplacian element stiffness MAPs and tetrahedra for Lagrange polynomials of degree one through three using symmetry.

Triangles				
Degree	n	m	nm	MAPs
1	6	3	18	9
2	21	3	63	17
3	55	3	165	46

Tetrahedra				
Degree	n	m	nm	MAPs
1	10	6	60	27
2	55	6	330	101
3	210	6	1260	370

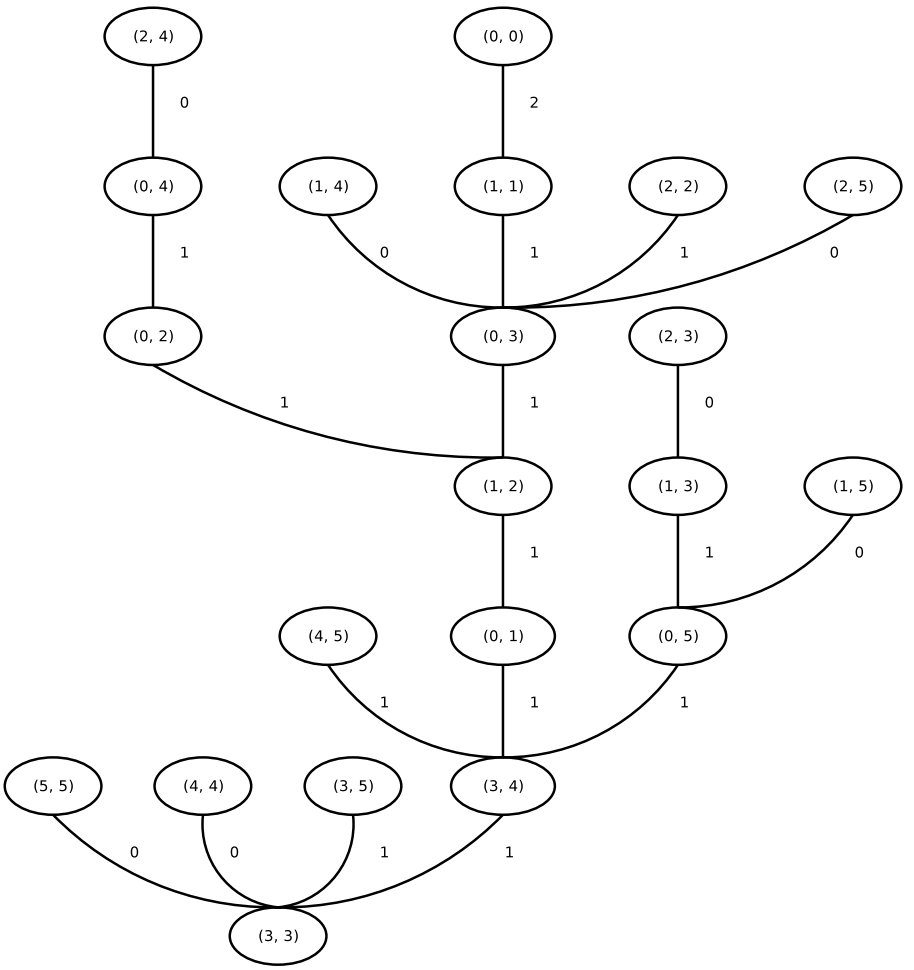


FIG. 1. *Minimum spanning tree for optimized computation of the Laplacian using quadratic elements on triangles. The node labeled* (3,3) *is the root, and the flow is from bottom to top.*

in actual runtime for matrix assembly. From the tensor contractions, we got another good speedup by simply omitting multiplication by zeros. From this, we gained only a modest additional speedup by using our additional optimizations. However, this is most likely due to the relative costs of memory access and floating point operations. We have to load the geometric tensor from memory, and we have to write every entry

TABLE 5
Number of MAPs in the optimized algorithm for computing the coordinate-advection element stiffness matrix on triangles and tetrahedra for Lagrange polynomials of degree one through three.

Triangles				
Degree	n	m	nm	MAPs
1	9	2	18	4
2	36	2	72	22
3	100	2	200	59

Tetrahedra				
Degree	n	m	nm	MAPs
1	16	3	48	9
2	100	3	300	35
3	400	3	1200	189

of the matrix to memory. So then $n+m$ in the tables gives a lower bound on memory access for computing the stiffness matrix. Our optimizations lead to algorithms for which there are a comparable number of arithmetic and memory operations. Hence, our optimization has succeeded in reducing the cost of computing the local stiffness matrix to a small increment to the cost of writing it to memory.

4.2. Advection in one coordinate direction. Now we consider constant coefficient advection aligned with a coordinate direction

$$a(v,u) = \int_\Omega v(x) \frac{\partial u(x)}{\partial x_1} \,\mathrm{d}x. \tag{11}$$

This is part of the operator associated with constant coefficient advection in some arbitrary direction—optimizing the other coordinate directions would give similar results. These results are shown in Table 5. Again, our optimization generates algorithms for which the predominant cost of computing the element matrix is writing it down, as there is significantly less than one floating point cycle per matrix entry in every case.

4.3. Weighted Laplacian. Our final operator is the variable coefficient Laplacian

$$a_w(v,u) = \int_\Omega w(x) \nabla v(x) \cdot \nabla u(x) \,\mathrm{d}x. \tag{12}$$

This form may be viewed as a trilinear form $a(w,v,u)$ in which w is the projection of the coefficient into the finite element space. For many problems, this can be performed without a loss in order of convergence. For nonlinear problems, we will have to reassemble this form at each nonlinear iteration, so it is an important step to optimize. The reference tensor is

$$A^0_{i\alpha} = \int_E \Phi_{\alpha_1}(X) \frac{\partial \Phi_{i_1}(X)}{\partial X_{\alpha_2}} \frac{\partial \Phi_{i_2}(X)}{\partial X_{\alpha_3}} \,\mathrm{d}X \tag{13}$$

and the geometric tensor is

$$G^\alpha_e = w_{\alpha_1} \det F'_e \frac{\partial X_{\alpha_2}}{\partial x_\beta} \frac{\partial X_{\alpha_3}}{\partial x_\beta}. \tag{14}$$

Note that the geometric tensor is the outer product of the geometric tensor for the constant coefficient Laplacian, which we shall denote $(G^L)_e$, with the coefficients of the weight function. Unlike the constant coefficient case, the amount of arithmetic per entry in the element stiffness matrix grows with the polynomial degree.

TABLE 6

Number of MAPs in the optimized algorithm for computing the weighted Laplacian element stiffness matrix on triangles and tetrahedra for Lagrange polynomials of degree one through three using symmetry.

Triangles

Degree	n	m	nm	MAPs
1	6	9	54	27
2	21	18	378	218
3	55	30	1650	1110

Tetrahedra

Degree	n	m	nm	MAPs
1	10	24	240	108
2	55	60	3300	1650
3	210	120	25200	14334

We could simply proceed with the optimization as we did for other forms—the element tensor is only a collection of vectors that will have to be dotted into G_e for each element in the mesh, but now the dot products are more expensive since the vectors are longer. In this case, G_e must be explicitly formed for each element (this costs $|P|d^2$ once $(G^L)_e$ is formed). On the other hand, we could use the decomposition of G_e into $(G^L)_e$ and the coefficient vector w_k and do the contractions in stages. For example, we could organize the contraction as

$$A_i^e = \left(A^0_{i,(\alpha_1,\alpha_2,\alpha_3)} \left(G^L\right)_e^{(\alpha_2,\alpha_3)}\right) w_{\alpha_1}; \tag{15}$$

that is, for each of the $|P|^2$ entries of the stiffness matrix, we compute $|P|$ contractions of $d\times d$ tensors with $(G^L)_e$. This is a similar optimization problem as the Laplacian but with $|P|$ times more vectors to optimize over. After we do this set of computations, we must compute $|P|^2$ dot products with the coefficient vector w_k. Note that the contractions with $(G^L)_e$ may be optimized, but the resulting vectors to dot with w_k will not be known until runtime and must be computed at full cost. Hence, a lower bound for this approach is $|P|$ MAPs per entry (assuming that the contractions with G^L were absolutely free). On the other hand, one could contract with the coefficient first to give an array of $|P|^2$ tensors of size $d \times d$ and then contract each of these with $(G^L)_e$. As before, the first step can be optimized, but the second step cannot.

In any of these cases, we may use the same transformations to exploit symmetry as we did in the constant coefficient case. Since $G_e^{\alpha_1,\alpha_2,\alpha_3} = G_e^{\alpha_1,\alpha_3,\alpha_2}$, we may view each slice $A^0_{i\cdot}$ of the reference tensor as an array of $|P|$ tensors of size $d \times d$ and apply the transformation to each of these. If we fully form G_e, this reduces the cost from $|P|d^2$ to $|P|\binom{d+1}{2}$. In all of our experiments, we made use of this approach.

In Table 6, we see the cost of computing the weighted Laplacian by the first approach (optimizing directly the tensor product $A_i^e = A^0_{i\alpha} G_e^\alpha$). While the optimizations are not as successful as for the constant coefficient operators, we still get reductions of 30%–50% in the operation counts.

When we perform the contraction in stages, we find more dependencies (for example, the slices of two of the tensors could be colinear although the entire tensors are not). We show the cost of performing the optimized stage for contracting with $(G^L)_e$ first in Table 7 and for contracting with w_k first in Table 8.

In order to get a fair comparison between these approaches, we must factor in the additional costs of building G_e or performing the second stage of contraction. Once $(G^L)_e$ is built and symmetrized, it costs an additional $|P|\binom{d+1}{2}$ MAPs to construct G_e. If we optimize the computation of contracting with $(G^L)_e$ first, we do not have to build G_e, but we must perform a dot product with w_k for each entry of the matrix.

TABLE 7

Number of MAPs in the optimized algorithm for performing all of the contractions with $(G^L)_e$ in the weighted Laplacian on triangles and tetrahedra first, resulting in $\binom{|P|+1}{2}$ arrays of length $|P|$ to contract with w_k.

Triangles

Degree	n	m	nm	MAPs
1	18	3	54	9
2	126	3	378	115
3	550	3	1650	683

Tetrahedra

Degree	n	m	nm	MAPs
1	40	6	240	27
2	550	6	3300	693
3	4200	6	25200	7021

TABLE 8

Number of MAPs in the optimized algorithm for performing all of the contractions with the coefficient in the weighted Laplacian on triangles and tetrahedra first, resulting in $\binom{|P|+1}{2}$ arrays of length $\binom{d+1}{2} = 6$ to contract with $(G^L)_e$.

Triangles

Degree	n	m	nm	MAPs
1	18	3	54	7
2	63	6	378	138
3	165	10	1650	899

Tetrahedra

Degree	n	m	nm	MAPs
1	60	4	240	9
2	330	10	3300	465
3	1260	20	25200	7728

TABLE 9

Comparing the total number of MAPs for fully forming G_e, contracting with $(G^L)_e$ first and contracting with w_k first on both triangles and tetrahedra.

Triangles

G_e			
Degree	MST	Additional	Total
1	27	3*3	38
2	218	3*6	236
3	1110	3*10	1140

$(G^L)_e$ first			
Degree	MST	Additional	Total
1	9	6*3	27
2	115	21*6	241
3	683	55*10	1233

w_k first			
Degree	MST	Additional	Total
1	7	6*3	25
2	138	21*3	201
3	899	55*3	1064

Tetrahedra

G_e			
Degree	MST	Additional	Total
1	108	6*4	132
2	1650	6*10	1710
3	14334	6*20	14454

$(G^L)_e$ first			
Degree	MST	Additional	Total
1	27	10*4	67
2	693	55*10	1234
3	7021	210*20	11221

w_k first			
Degree	MST	Additional	Total
1	9	10*6	69
2	465	55*6	795
3	7728	210*6	8988

This costs $|P|$ per contraction with $\binom{|P|+1}{2}$ entries in the matrix. If we optimize the contraction with each w_k first, then we have an additional $\binom{|P|+1}{2}$ contractions with $(G^L)_e$ at a cost of $\binom{d+1}{2}$ each. We expect that which of these will be most effective must be determined (automatically) on a case-by-case basis. Table 9 shows the comparisons for the first approach (labeled G_e), the second approach (labeled $(G^L)_e$), and the third approach (labeled w_k) by indicating the cost of the optimized computation plus

the additional stages of computation. In most of these cases, contracting with the coefficient first leads to the lowest total cost.

5. Optimizing the optimization process. Since our graph (Y, E) is completely connected, we have $|E| = O(|Y|^2)$ and our optimization process requires complexity that is at least quadratic in the number of entries in the element stiffness matrix. In this section, we show how certain useful complexity-reducing relations may be evaluated over all of Y in better than quadratic time and then discuss how we may build a sparse graph (Y, E') with $|E| = O(|Y|)$ that will admit a much more efficient optimization process. Even though this process must be run only once per form and element (say, the Laplacian with quadratics on triangles), the quadratic algorithm can become very time consuming and challenge a single computer's resources for forms of high arity using high degree polynomials on tetrahedra.

5.1. Search algorithms. Before discussing how we may evaluate some of these complexity-reducing relations over the collection Y in better than quadratic time, we first describe some basic notation we will use for hash tables throughout this section and the next.

Hash tables are standard data structures [14] that associate each member of a set of $\cdot, \prime$ to some value, possibly drawn from another set of objects. The important point about hash tables is that the basic operations of setting and getting values are expected to be independent of the number of entries in the table (expected constant time access). Many higher-level programming languages have library support or built-in features supporting hash tables (many implementations of the standard template library in C++ include hash tables, and scripting languages such as Python and Perl have them built in as primitive types).

We begin by establishing some notation for the basic operations we use. If a is a key of table T, then we find the value associated with a by $T[a]$. If there is no value associated with a (that is, if a is not a key of T), we may update T by adding a key a associated with value b by the notation $T[a] \leftarrow b$. We use the same notation to indicate setting a new value to an existing key.

As before, we label the vectors y^i for $1 \le i \le n$. We want to partition the labels into a set of subsets $\mathcal{E}$ such that for each $E \in \mathcal{E}$, the vectors associated with each label in E are equal. Moreover, if two vectors are equal, then their labels must belong to the same E. This is easily accomplished by setting up a hash table whose keys are vectors and whose values are subsets of the integers $1 \le i \le n$. This process is described in Algorithm 1.

Algorithm 1 Determining equality among vectors

```
E an empty table mapping vectors to subsets of {i}_{i=1}^n
for all 1 ≤ i ≤ n do
  if y^i is a key of E then
    E[y^i] ← E[v^i] ∪ {i}
  else if −y^i is a key of E then
    E[−y^i] ← E[−y^i] ∪ {i}
  else
    E[y^i] ← {i}
  end if
end for
```

Floating point arithmetic presents a slight challenge to hashing. Numbers which are close together (within some tolerance) that should be treated as equal must be rounded to so that they are indeed equal. Hashing relies on a function that maps items into a set of integers (the "hash code"). These functions are discontinuous and sensitive to small perturbations. For most numerical algorithms in floating point arithmetic, we may define equality to be "near equality," but hash tables require us to round or use rational arithmetic before any comparisons are made. We have successfully implemented our algorithms in both cases.

As an alternative to hashing, one could form a binary search tree or sort the vectors by a lexicographic ordering. These would rely on a more standard "close to equal" comparison operation but run only in $O(mn \log(mn))$ time. So, for large enough data sets, hashing will be more efficient.

We may similarly partition the labels into a set of subsets $\mathcal{C}$ such that for each $C \in \mathcal{C}$, the vectors associated with the labels in C are colinear. Similarly, if two vectors are colinear, then their labels must belong to the same C. This process may be performed by constructing the collection of unit vectors $\hat{A}$, with $\hat{y}_i = \frac{y_i}{\|y_i\|}$ for each $1 \leq i \leq n$ for some norm $\|\cdot\|$ and using Algorithm 1 on $\hat{A}$.

Finding vectors that are close together in Hamming distance is more subtle. At worst, the cost is $O(mn^2)$, as we have to compare every entry of every pair of vectors. However, we may do this in expected linear time with some assumptions about Y. We first describe the algorithm and then state the conditions under which the algorithm performs in worse than linear time.

Our vectors each have m components. We start by forming m empty hash tables. Each H_i will map numbers that appear in the ith position of any vector to the labels of vectors that have that entry in that position. This is presented in Algorithm 2, in which y^i_j denotes the jth entry of y^i.

Algorithm 2 MAPPING UNIQUE ENTRIES AT EACH POSITION TO VECTORS CONTAINING THEM

```
for all 1 ≤ i ≤ d do
  H_i an empty table mapping numbers to sets of vector labels from {i}_{i=1}^n
end for
for all 1 ≤ i ≤ n do
  for all 1 ≤ j ≤ m do
    if y_j^i is a key of H_j then
      H_j[y_j^i] ← H_j[y_j^i] ∪ {i}
    else
      H_j[y_j^i] := {i}
    end if
  end for
end for
```

This process runs in expected $O(nm)$ time. From these tables, we can construct a table that gives the Hamming distance between any two vectors, as seen in Algorithm 3. This algorithm counts down from d each time it discovers an entry that two vectors share. Our algorithm reflects the symmetry of the Hamming distance.

On output, for $1 \leq i < j \leq n$, if $D[i][j]$ has no entry, the distance between v^i and v_j is m. Otherwise, the table contains the Hamming distance between the two vectors.

Algorithm 3 Computing Hamming distances efficiently

```
D an empty table
for all 1 ≤ i ≤ n do
  D[i] an empty table
end for
for all 1 ≤ i ≤ m do
  for all a in the keys of H_i do
    for all unique combinations k, ℓ of labels in H_i[a] do
      α := min(k, ℓ), β := max(k, ℓ)
      if D[α] has a key β then
        D[α][β] ← D[α][β] − 1
      else
        D[α][β] := m − 1
      end if
    end for
  end for
end for
```

Regarding complexity, there is a double loop over the entries of each H_i. Hence, the algorithm is quadratic in the maximum number of vectors that share the same entry at the same location. Presumably, this is considerably less than n on most data sets.

5.2. Using a sparse graph. If we create a graph $(Y, \tilde{E})$ with significantly fewer edges than (Y, E), we may be able to get most of the reduction in operation count while having a more efficient optimization process. For example, we might choose to put a cutoff on ρ, using only edges that have a large enough complexity reduction. So, we can define the set of edges being

$$E_k = \{\{y^i, y^j\} : \rho(y^i, y^j) \leq k\}. \tag{16}$$

For example, if we use Algorithms 2 and 3 to evaluate H^+ over all pairs from Y, then we are using $\rho = H^+$ and $k = m - 1$. Also, note that our structure D encodes a sparse graph. That is, the vectors of Y are the nodes of the graph, the neighbors of each y^i are simply the keys of $D[y^i]$, and the edge weight between some y^i and neighbor y^j is $D[y^i][j^j]$. It is not hard to move from this table into whatever data structure is used for graphs.

Then we could add colinearity or other complexity-reducing relations to this graph. If we use Algorithm 1 on the unit vectors to determine sets of colinear vectors, we can update the graph either by adding edges or updating the edge weights for each pair of colinear vectors.

If $|E_k| = O(|Y|)$, then computing a minimum spanning tree will require only $O(n \log n)$ time rather than $O(n^2 \log n)$. However, there is no guarantee that (Y, E_k) will be a connected graph. Some vectors might not have close neighbors, or else some subgraphs do not connect with each other. An optimized algorithm can still be obtained by finding the connected components of the (Y, E_k) and finding a minimum spanning tree for each component. Then the total cost of the computation is m times the number of connected components plus the sum of the weights of each of the minimum spanning trees.

6. Conclusion and ongoing work. We have developed a general optimization strategy for the evaluation of local stiffness matrices for finite elements. This is based on first formulating the computation as a sequence of tensor contractions and then introducing a new concept of complexity-reducing relations that allows us to set the optimization in a graph context. The optimization itself proceeds by computing a minimum spanning tree. These techniques worked very well at reducing the cost of evaluating finite element matrices for several forms using Lagrange elements of degrees one through three on triangles and tetrahedra. Finally, we discussed some efficient algorithms for detecting equality and colinearity and for evaluating the pairwise Hamming distance over the entire set of tensors.

We hope to extend our optimizations in several ways. First, we remarked that it may be possible to enrich the collection of tensors in such as way as to deliver a minimum spanning tree with lower total weight. We hope to explore such a process in the future. Also, we saw in [11] that frequently some of the tensors are linear combinations of two or more other tensors. However, both locating and making use of such relations in a more formal context have been difficult. We are working on geometric search algorithms to locate linear dependencies efficiently. However, once they are located, our optimization process must occur over a hypergraph rather than a graph. Finding a true minimum is also much more difficult, and we are working on heuristics that will allow us to combine these two approaches.

Finally, we plan to integrate our optimization strategy with FFC. While FFC currently generates very efficient code for evaluating variational forms, we will improve upon this generated code by piping the tensors through our optimization process before generating code to perform the contractions. This will lead to a domain-specific optimizing compiler for finite elements; by exploiting latent mathematical structure, we will automatically generate more efficient algorithms for finite elements than people write by hand.

REFERENCES

[1] B. Bagheri and R. Scott, *Analysa*, http://people.cs.uchicago.edu/~ridg/al/aa.html.

[2] W. Bangerth, R. Hartmann, and G. Kanschat, `deal.II` *Differential Equations Analysis Library*, http://www.dealii.org.

[3] S. C. Brenner and L. R. Scott, *The Mathematical Theory of Finite Element Methods*, Springer-Verlag, New York, 1994.

[4] P. G. Ciarlet, *Numerical Analysis of the Finite Element Method*, Les Presses de l'Universite de Montreal, Montreal, QC, Canada, 1976.

[5] P. Dular and C. Geuzaine, *GetDP: A General Environment for the Treatment of Discrete Problems*, http://www.geuz.org/getdp/.

[6] J. Hoffman, J. Jansson, and A. Logg, *DOLFIN*, http://www.fenics.org/dolfin/.

[7] J. Hoffman and A. Logg, *DOLFIN: Dynamic Object Oriented Library for FINite Element Computation*, Tech. report 2002–06, Chalmers Finite Element Center Preprint Series, Chalmers Finite Element Center, Göteborg, Sweden, 2002.

[8] G. E. Karniadakis and S. J. Sherwin, *Spectral/hp Element Methods for CFD*, Numerical Mathematics and Scientific Computation, Oxford University Press, New York, 1999.

[9] R. C. Kirby, *FIAT: A new paradigm for computing finite element basis functions*, ACM Trans. Math. Software, 30 (2004), pp. 502–516.

[10] R. C. Kirby, *Optimizing FIAT with the level* 3 *BLAS*, ACM Trans. Math. Software, to appear.

[11] R. C. Kirby, M. Knepley, A. Logg, and L. R. Scott, *Optimizing the evaluation of finite element matrices*, SIAM J. Sci. Comput., 27 (2005), pp. 741–758.

[12] R. C. Kirby, M. Knepley, and L. R. Scott, *Evaluation of the Action of Finite Element Operators*, Tech. report 2004–07, University of Chicago, Chicago, IL, 2004.

[13] R. C. Kirby and A. Logg, *A compiler for variational forms*, ACM Trans. Math. Software, to appear.

[14] A. Levitin, *Introduction to the Design and Analysis of Algorithms*, Addison–Wesley, Boston, 2003.
[15] A. Logg, *The FEniCS Form Compiler FFC*, http://www.fenics.org/ffc/.
[16] K. Long, *Sundance*, http://csmr.ca.sandia.gov/~krlong/sundance.html.
[17] K. Long, *Sundance, a rapid prototyping tool for parallel PDE-constrained optimization*, in Large-Scale PDE-Constrained Optimization, Lect. Notes Comput. Sci. Engrg., Springer-Verlag, Berlin, 2003.
[18] J.-C. Nédélec, *Mixed finite elements in* $\mathbf{R}^3$, Numer. Math., 35 (1980), pp. 315–341.
[19] J.-C. Nédélec, *A new family of mixed finite elements in* $\mathbf{R}^3$, Numer. Math., 50 (1986), pp. 57–81.
[20] O. Pironneau, F. Hecht, and A. Le Hyaric, *FreeFEM*, http://www.freefem.org/.
[21] P.-A. Raviart and J. M. Thomas, *A mixed finite element method for* 2*nd order elliptic problems*, in Mathematical Aspects of Finite Element Methods (Proc. Conf., Consiglio Naz. delle Ricerche (C.N.R.), Rome, 1975), Lecture Notes in Math. 606, Springer, Berlin, 1977, pp. 292–315.
[22] P.-A. Raviart and J. M. Thomas, *Primal hybrid finite element methods for* 2*nd order elliptic equations*, Math. Comp., 31 (1977), pp. 391–413.

SIAM J. SCI. COMPUT.
Vol. 28, No. 1, pp. 241–259
© 2006 Society for Industrial and Applied Mathematics

SPHERICAL SPLINES FOR DATA INTERPOLATION AND FITTING*

V. BARAMIDZE†, M. J. LAI†, AND C. K. SHUM‡

Abstract. We study minimal energy interpolation and discrete and penalized least squares approximation problems on the unit sphere using nonhomogeneous spherical splines. Several numerical experiments are conducted to compare approximating properties of homogeneous and nonhomogeneous splines. Our numerical experiments show that nonhomogeneous splines have certain advantages over homogeneous splines.

Key words. spherical splines, data fitting

AMS subject classifications. 65D05, 65D07, 65D17

DOI. 10.1137/040620722

1. Introduction. Contemporary research in atmospheric sciences, geodesy, and geophysics requires the use of global data heterogeneously distributed in space around Earth. For spherical data interpolation/approximation, tensor products of univariate splines are not a good choice, since data locations are not usually spaced over a regular grid. Radial basis functions are not good candidates either, since the data values may have no rotational symmetry.

Spherical Bernstein–Bézier splines (introduced in [1] and studied in [2] and [7]) are well suited for scattered data interpolation/approximation problems. The spherical spline functions have many properties in common with classical polynomial splines over planar triangulations. Moreover, many spline interpolation and approximation methods for planar scattered data problems have analogues in spherical setting [2].

One of the disadvantages of using homogeneous spherical splines is that spline spaces of even and odd degrees have only zero function in common due to homogeneity of the basis. More explicitly, the spline space $S_d^r(\Delta)$ of degree d and smoothness r (defined in [2]) with d odd does not contain constant functions, and the spline space $S_d^r(\Delta)$ with d even does not contain linear functions. Therefore in the homogeneous spline spaces reproduction of polynomials of degree $m \leq d$ is not possible unless $m = d \bmod(2)$.

To overcome this disadvantage we propose a simultaneous employment of two spaces $S_d^r(\Delta)$ and $S_{d-1}^r(\Delta)$, with one degree even and the other odd. A direct sum of two spline spaces forms the larger nonhomogeneous spherical polynomial spline space proposed by [10]. A simple structure of the resulting spline space allows us to use methods developed for homogeneous splines in nonhomogeneous spline spaces. The spline solution obtained via minimization over the larger space has in general higher accuracy. In addition, spherical nonhomogeneous splines reproduce spherical nonhomogeneous polynomials.

*Received by the editors December 13, 2004; accepted for publication (in revised form) July 27, 2005; published electronically March 10, 2006. Results in this paper are based on research supported by the National Science Foundation under grant 0327577.

http://www.siam.org/journals/sisc/28-1/62072.html

†Department of Mathematics, The University of Georgia, Athens, GA 30602 (vbaramid@math.uga.edu, mjlai@math.uga.edu).

‡Laboratory for Space Geodesy and Remote Sensing, The Ohio State University, Columbus, OH 43210 (ckshum@osu.edu).

In this paper we establish existence and uniqueness of homogeneous and nonhomogeneous spherical spline approximations. Moreover, we show how the use of a powerful iterative algorithm [4] solving special linear systems leads to efficient computation of spherical splines of arbitrary degree and smoothness.

The paper is organized as follows. In section 2 we discuss homogeneous spherical polynomials and then introduce nonhomogeneous spherical polynomials. We review properties necessary for the implementation of the computational algorithms which are presented in sections 3, 4, 5. In section 3 we describe a minimal energy (ME) interpolation method using both homogeneous and nonhomogeneous splines. We discuss discrete and penalized least squares spline approximation algorithms in sections 4 and 5. In each section we start with a description of the method, follow with a computational algorithm, and end with numerical examples. Some examples are designed to demonstrate advantages of nonhomogeneous splines over homogeneous. In addition, we solve real world data fitting problems and offer visual evaluation of the solutions. One potential application of the development of spherical splines is in the investigation of inverse problems arising in geophysics. Using heterogeneously distributed satellite measurements from new satellite gravity missions such as GRACE (Gravity Recovery and Climate Experiment) [14] we aim to obtain an alternative representation of Earth's gravity field and a more realistic description of the geopotential.

2. Preliminary. In this section we review several basic concepts, outline some properties of spherical splines, explain computational algorithms for scattered data interpolation and fitting, and present a numerical iterative technique for implementation of the algorithms. All the material in this preparatory section can be found either in [1] and [2] or in [4].

Let us define spherical Bernstein–Bézier polynomials first. We begin with a concept of homogeneity. A trivariate function $f(v)$ is homogeneous of degree d if

$$f(\alpha v) = \alpha^d f(v). \tag{2.1}$$

Let $\mathcal{H}_d$ denote the space of trivariate homogeneous polynomials of degree d. That is, $\mathcal{H}_d := \operatorname{span}\{x^i y^j z^k,\ i+j+k=d\}$.

Let $\tau = \langle v_1, v_2, v_3\rangle$ be a nondegenerate spherical triangle; i.e., assume that the area on the unit sphere bounded by the great circular arcs connecting v_1 and v_2, v_1 and v_3, and v_2 and v_3 is not zero. Let $b_1(v)$, $b_2(v)$, and $b_3(v)$ be the spherical barycentric coordinates of a point $v \in \mathbb{S}^2$, i.e.,

$$v = b_1(v)v_1 + b_2(v)v_2 + b_3(v)v_3.$$

Linear independence of vectors $v_1,\ v_2$, and $v_3 \in \mathbb{R}^3$ imply that the b_i's are uniquely determined. Clearly, the b_i's are linear functions of v. It was shown in [1] that the set

$$B^d_{ijk}(v) = \frac{d!}{i!j!k!} b_1(v)^i b_2(v)^j b_3(v)^k, \qquad i+j+k=d, \tag{2.2}$$

of Bernstein–Bézier basis polynomials of degree d forms a basis for $\mathcal{H}_d$. It is easy to see that $B^d_{ijk}(v)$ is a homogeneous polynomial of degree d. Thus, when restricted to $\mathbb{S}^2$,

$$p(v) = \sum_{i+j+k=d} c_{ijk} B^d_{ijk}(v) \tag{2.3}$$

is called a homogeneous spherical Bernstein–Bézier (SBB)- polynomial of degree d.

Due to this special representation, many properties of SBB polynomials are analogous to those of classical planar Bernstein–Bézier polynomials [1]. However, evaluating integrals of spherical polynomials is considerably more difficult than in the planar setting. All integrals needed for our algorithms have to be evaluated numerically.

Let us now define nonhomogeneous spherical polynomials. It was shown in [10] that $\mathcal{H}_d \oplus \mathcal{H}_{d-1}$ restricted to the unit sphere is identical to the space $\mathcal{P}_d$ of all trivariate polynomials of degree d restricted to the unit sphere. Therefore the set $\{B^d_{ijk},\, i+j+k=d\} \cup \{B^{d-1}_{ijk},\, i+j+k=d-1\}$ on $\mathbb{S}^2$ forms a basis for $\mathcal{P}_d|_{\mathbb{S}^2}$. We can express a nonhomogeneous spherical polynomial p in terms of SBB-basis functions as

$$p = \sum_{i+j+k=d} a_{ijk} B^d_{ijk}(v) + \sum_{i+j+k=d-1} c_{ijk} B^{d-1}_{ijk}(v).$$

With this definition it is easy to see that evaluating (de Casteljau's algorithm), taking derivatives, and computing integrals of homogeneous polynomials can be easily adapted for nonhomogeneous polynomials.

We are now ready to define spherical splines. Given a set $\mathcal{V}$ of points on the unit sphere $\mathbb{S}^2$ we can form a triangulation Δ, which may be a triangulation of a spherical domain on $\mathbb{S}^2$ or of the entire sphere. We will assume that Δ is regular in the sense that any two triangles either do not intersect each other or share their common vertex or their common edge. Let

$$S^r_d(\Delta) := \{s \in C^r(\mathbb{S}^2), s|_\tau \in \mathcal{H}_d, \tau \in \Delta\}$$

be the spherical spline space of degree d and smoothness $r \geq -1$. Here

$$s|_\tau = \sum_{i+j+k=d} c^\tau_{ijk} B^{d,\tau}_{ijk}(v).$$

The space of nonhomogeneous spherical splines is defined as

$$N^r_d(\Delta) := \{s \in C^r(\mathbb{S}^2), s|_\tau \in \mathcal{P}_d, \tau \in \Delta\}$$

with

$$s|_\tau = \sum_{i+j+k=d} a^\tau_{ijk} B^{d,\tau}_{ijk}(v) + \sum_{i+j+k=d-1} c^\tau_{ijk} B^{d-1,\tau}_{ijk}(v).$$

In general, we know neither the dimension of $S^r_d(\Delta)$ nor the construction of locally supported basis functions such as finite elements for arbitrary d and r. However, the lack of understanding of the dimension and construction of locally supported basis functions of general spline spaces does not prevent us from using them for scattered data interpolation and fitting. It is one of the purposes of this paper to show how to use spherical splines of arbitrary d and r with $d > r$ for scattered data interpolation and fitting. The crucial step in the process is to solve a singular linear system arising from the application of the Lagrange multipliers method [2] to the minimization. The linear system has a form which is explained later in detail:

$$\begin{bmatrix} A & L^T \\ L & 0 \end{bmatrix} \begin{bmatrix} \mathbf{c} \\ \lambda \end{bmatrix} = \begin{bmatrix} F \\ G \end{bmatrix}.$$

It is necessary to derive a computational algorithm to efficiently obtain the vector $\mathbf{c}$ containing coefficients of the spline solution. We use the following iterative method [4]:

$$\begin{bmatrix} A & L^T \\ L & -\epsilon I \end{bmatrix} \begin{bmatrix} \mathbf{c}^{(\ell+1)} \\ \lambda^{(\ell+1)} \end{bmatrix} = \begin{bmatrix} F \\ G - \epsilon\lambda^{(\ell)} \end{bmatrix},$$

for $\ell = 0, 1, \ldots$, where $\epsilon > 0$ is a fixed number, e.g., $\epsilon = 10^{-4}$, $\lambda^{(\ell)}$ is an iterative solution of a Lagrange multiplier coefficient vector with $\lambda^0 = 0$, and I is the identity matrix. The above matrix iterative steps can in fact be rewritten as follows:

$$\left(A + \frac{1}{\epsilon} L^T L\right) \mathbf{c}^{(\ell+1)} = AF\mathbf{c}^{(\ell)} + \frac{1}{\epsilon} L^T G$$

with $\mathbf{c}^{(0)} = 0$. Note that the size of the above linear system is much smaller than the original. The iterations converge very quickly as shown in the following theorem. In our numerical experiments, a few iterations (less than 10) often suffice. A general convergence theorem is proved in [4]. To state the convergence result, we need the following definition.

DEFINITION 2.1. [illegible] A [illegible] $n \times n$ [illegible] L [illegible] $m \times n$ [illegible] A [illegible] L [illegible] $\mathbf{c}^T A\mathbf{c} \geq 0$ [illegible] $A\mathbf{c} = 0$ $L\mathbf{c} = 0$ [illegible] $\mathbf{c} = 0$

THEOREM 2.2. [illegible] A [illegible] L [illegible] $A + \frac{1}{\epsilon} L^T L$ [illegible] $\epsilon > 0$ [illegible] C [illegible]

$$\|\mathbf{c}^{(\ell+1)} - \mathbf{c}\| \leq C\epsilon \|\mathbf{c}^{(\ell)} - \mathbf{c}\| \quad \text{[illegible]} \quad \ell \geq 0.$$

3. Minimal energy interpolation.

3.1. Energy functionals. Recall that $\mathcal{V} := \{v \in \mathbb{S}^2\}$ is a set of points on the unit sphere and $\{f(v), v \in \mathcal{V}\}$ is a set of spherical function values given at the locations $\mathcal{V}$. Let Δ be a regular spherical triangulation of $\mathbb{S}^2$ with vertices $\mathcal{V}$. For two integers d and r let $S_d^r(\Delta)$ be the spline space of spherical homogeneous polynomials of degree d and smoothness r.

We need to find a spline function $s \in S_d^r(\Delta)$ interpolating f at $\mathcal{V}$. In general such a spline is not unique. A typical way to use extra degrees of freedom is to minimize a functional $\mathcal{E}(s)$ measuring smoothness of s. In [2] the energy functional

$$\mathcal{E}(s) = \int_{\mathbb{S}^2} (\Delta^* s)^2 d\mu \tag{3.1}$$

was used, where Δ^* is the Laplace–Beltrami operator, μ is the Lebesgue measure on $\mathbb{S}^2$, and the integral in (3.1) is taken over the unit sphere. Note that the Laplace–Beltrami operator annihilates constants only. We propose an alternative functional motivated by Sobolev-type seminorms defined in [12]. Let

$$\mathcal{E}_\delta(s) = \int_{\mathbb{S}^2} \sum_{|\alpha|=2} (D^\alpha s_\delta)^2 d\mu. \tag{3.2}$$

In (3.2) s_δ is the unique homogeneous extension of s of degree δ to $\mathbb{R}^3 \setminus \{0\}$ defined by $s_\delta(v) = |v|^\delta s(\frac{v}{|v|})$. If the degree d of the homogeneous spline space is even, we take $\delta = 0$; if odd, we take $\delta = 1$. After evaluating second order partial derivatives, $D^\alpha s_\delta$ are restricted to $\mathbb{S}^2$ and are then integrated.

To establish existence and uniqueness of a spherical spline in $S_d^r(\Delta)$ interpolating f and minimizing (3.2), we need the following.

LEMMA 3.1. [illegible] Δ [illegible] $f \neq 0$ [illegible]

(1) $\mathcal{E}_0(f) = 0$ [illegible] f [illegible]

(2) $\mathcal{E}_1(f) = 0$ [illegible] f [illegible] $\mathbb{S}^2$

If $\mathcal{E}_\delta(f) = 0$, then by the definition, $D^\alpha f_\delta = 0$ on every triangle $\tau \in \Delta$. Consider $\delta = 1$. Since f_1 is linear homogeneous, $D^\alpha f_1$ is homogeneous of degree -1, and therefore, by the uniqueness of homogeneous extensions, $(D^\alpha f_1|_\tau)_{-1} = D^\alpha f_1$. On the other hand, by the definition, $(D^\alpha f_1|_\tau)_{-1}(v) = |v|^{-1}(D^\alpha f_1|_\tau)(\frac{v}{|v|})$. As we noted above, $D^\alpha f_1|_\tau = 0$, and therefore $D^\alpha f_1 = 0$ as well. Hence f_1 is a polynomial of degree at most 1. Since it is a homogeneous linear function, f_1 must be a homogeneous linear polynomial on $\mathbb{R}^3$. Therefore by the uniqueness of homogeneous extensions, f is a linear homogeneous polynomial on τ. A similar proof works for $\delta = 0$. The other direction follows trivially from definition (3.2). □

Let

$$\Gamma(f) := \{s \in S_d^r(\Delta) : s(v) = f(v) \text{ for all } v \in \mathcal{V}\}$$

be the set of all splines in $S_d^r(\Delta)$ interpolating f at the vertices of triangulation Δ. Let $s_f \in \Gamma(f)$ denote a spherical spline minimizing (3.2) over $\Gamma(f)$.

LEMMA 3.2. $\Gamma(f)$ $s_f \in S_d^r(\Delta)$ $f = 0$ (3.2) $\delta = d \bmod(2)$

Since $\mathcal{E}_\delta(s) \geq 0$ for all $s \in S_d^r(\Delta)$, $\mathcal{E}_\delta(s_f) = 0$ is the absolute minimum of $\mathcal{E}_\delta$ achieved at $s_f = 0$. To show the uniqueness, assume there is another $s \in \Gamma(0)$ with $\mathcal{E}_\delta(s) = 0$. We need to prove that $s = s_f$. By our assumption, $\mathcal{E}_\delta(s) = 0$ on every triangle $\tau \in \Delta$. By Lemma 3.1 s is either a linear homogeneous function (if d is odd) or s is a constant (if d is even) on every triangle $\tau \in \Delta$. Since s interpolates 0 at the vertices of each triangle, $s = 0$ on each triangle. Therefore $s = s_f$. □

THEOREM 3.3. Δ $\mathbb{S}^2$ $\mathcal{V}$ $\{f(v),\ v \in \mathcal{V}\}$ f d, r $d \geq 3r+2$ $s_f \in S_d^r(\Delta)$ f $\mathcal{E}_\delta$

Since $d \geq 3r + 2$, $\Gamma(f)$ is not empty [3]. There exists an ME spline s_f interpolating f since $\Gamma(f)$ is a nonempty closed convex set. To prove uniqueness suppose there exists $q_f \in \Gamma(f)$ minimizing $\mathcal{E}_\delta$, i.e., $\mathcal{E}_\delta(s_f) = \mathcal{E}_\delta(q_f)$. Since $\mathcal{E}_\delta(s_f + \nu s)$ achieves its minimal value at $\nu = 0$ over $s \in \Gamma(0)$, we have

$$\frac{d}{d\nu}\mathcal{E}_\delta(s_f + \nu s)|_{\nu=0} = 0,$$

which leads to

$$\sum_{|\alpha|=2} \int_{\mathbb{S}^2} (D^\alpha s_{f,\delta})(D^\alpha s_\delta) d\mu = 0$$

for $s \in \Gamma(0)$. In particular for $s = s_f - q_f$ we get

$$\sum_{|\alpha|=2} \int_{\mathbb{S}^2} (D^\alpha s_{f,\delta})^2 d\mu = \sum_{|\alpha|=2} \int_{\mathbb{S}^2} (D^\alpha s_{f,\delta})(D^\alpha q_{f,\delta}) d\mu.$$

Therefore,

$$\mathcal{E}_\delta(s_f - q_f) = 0,$$

and by Lemma 3.2 $s_f - q_f \equiv 0$. This completes the proof. □

To develop ME interpolation using nonhomogeneous splines, we fix integers $d \geq 1$ and $r \geq 0$ and recall that $N_d^r(\Delta) = S_d^r(\Delta) \oplus S_{d-1}^r(\Delta)$. To simplify our notation let us

assume that d is odd. We can present a spline function $s \in N_d^r(\Delta)$ as a sum $s_1 + s_0$, where the subscript 1 indicates that $s_1 \in S_d^r(\Delta)$ and that we use its linear extension to compute derivatives. Similarly, the subscript 0 indicates that $s_0 \in S_{d-1}^r(\Delta)$ and that we work with its constant extension.

We define an energy functional which annihilates nonhomogeneous linear polynomials as well as constants and homogeneous linear polynomials

$$\mathcal{E}(s) = \lambda \int_{\mathbb{S}^2} \sum_{|\alpha|=2} (D^\alpha s_1)^2 d\mu + (1-\lambda) \int_{\mathbb{S}^2} \sum_{|\alpha|=2} (D^\alpha s_0)^2 d\mu, \tag{3.3}$$

with $0 < \lambda < 1$.

LEMMA 3.4. [illegible] d [illegible] r [illegible] $N_d^r(\Delta)$ [illegible] f [illegible]

$$\widetilde{\Gamma}(f) := \{s \in N_d^r(\Delta) : s(v) = f(v) \text{ [illegible] } v \in \mathcal{V}\}$$

[illegible] $N_d^r(\Delta)$ [illegible] f [illegible] Δ [illegible] $s_f = 0 \in N_d^r(\Delta)$ [illegible] $f = 0$ [illegible] (3.3)

[illegible]. By the definition, $0 = s_f = s_1 + s_0$, where $s_1|_\tau$ is a homogeneous polynomial of degree d and $s_0|_\tau$ is a homogeneous polynomial of degree $d-1$. Since $S_d^r(\Delta) \cap S_{d-1}^r(\Delta) = 0$, $s_1 = 0$ and $s_0 = 0$. By definition (3.3), $\mathcal{E}(s_f) = 0$. Since $\mathcal{E}(s) \geq 0$ for all $s \in N_d^r(\Delta)$, $\mathcal{E}(0) = 0$ is the absolute minimum of $\mathcal{E}$. To show the uniqueness, assume there is $q_f \in \Gamma(0)$ with $\mathcal{E}(q_f) = 0$. We need to show that $q_f = s_f = 0$. As above, we know $q_f = q_1 + q_0$ for some $q_1 \in S_d^r(\Delta)$ and $q_0 \in S_{d-1}^r(\Delta)$. Then $\mathcal{E}_1(q_1) = 0$ and $\mathcal{E}_0(q_0) = 0$ on every triangle $\tau \in \Delta$. By Lemma 3.1 q_1 is a linear homogeneous polynomial and q_0 is a constant on $\mathbb{S}^2$. Therefore $q_1 + q_0$ is a trivariate linear polynomial satisfying zero interpolation conditions over points $v \in \mathcal{V}$, none of which is the origin. By the linear independence of x, y, z, and 1, $q_f = 0$ on every triangle. Therefore $q_f = s_f$. □

THEOREM 3.5. [illegible] Δ [illegible] $\mathbb{S}^2$ [illegible] $\mathcal{V}$ [illegible] $\{f(v), v \in \mathcal{V}\}$ [illegible] $d \geq 1$ $r \geq 0$ [illegible] $d \geq 3r + 2$ [illegible] $s_f \in N_d^r(\Delta)$ [illegible] f [illegible] $\mathcal{E}$

[illegible]. The proof follows from Lemma 3.4 as in the proof of Theorem 3.3. □

3.2. Computational algorithms. In this section we explain how to compute ME spherical interpolating splines. We use a coefficient vector $\mathbf{c} := (c_{ijk}^\tau, i+j+k = d, \tau \in \Delta)$ to represent each spline function in $S_d^r(\Delta)$. To simplify the data management we linearize the triple indices of SBB-coefficients c_{ijk} and correspondingly the indices of SBB-basis functions B_{ijk}^d. From the properties of SBB-polynomials, we have

$$c_{d00} = f(v_1), \quad c_{0d0} = f(v_2), \quad c_{00d} = f(v_3)$$

for the vertices of each triangle $\tau \in \Delta$. We assemble interpolation conditions into a matrix $\mathbf{K}$, according to the order of the coefficient vector $\mathbf{c}$. Then $\mathbf{Kc} = \mathbf{F}$ is the linear system of equations such that the coefficient vector $\mathbf{c}$ corresponds to a spline s interpolating f at the data sites $\mathcal{V}$.

To ensure the C^r continuity across each edge of Δ, we impose smoothness conditions which can be found in [1]. Let $\mathbf{M}$ denote the smoothness matrix such that $\mathbf{Mc} = 0$ if and only if $s \in S_d^r(\Delta)$.

Next fix $\delta = d \bmod(2)$. Define the energy matrix $\mathbf{E}$ by $\mathbf{E} = \operatorname{diag}(\mathbf{E}_\tau,\ \tau \in \Delta)$. Each block $\mathbf{E}_\tau$ is associated with a triangle τ and contains the entries

$$\mathbf{E}_{ij} := \int_\tau \sum_{|\alpha|=2} D^\alpha(B_i)_\delta D^\alpha(B_j)_\delta d\mu, \tag{3.4}$$

where B_i denotes a SBB-polynomial basis function (2.2) of degree d corresponding to the order of the linearized triple indices (i, j, k), $i + j + k = d$.

The problem of minimizing (3.2) over $S_d^r(\Delta)$ can be formulated as follows:

$$\text{Minimize } \mathbf{c}^T\mathbf{E}\mathbf{c} \text{ subject to } \mathbf{M}\mathbf{c} = 0 \text{ and } \mathbf{K}\mathbf{c} = \mathbf{F}.$$

Using the method of Lagrange multipliers, we must solve the linear system

$$\begin{bmatrix} \mathbf{E} & \mathbf{K}^T & \mathbf{M}^T \\ \mathbf{K} & 0 & 0 \\ \mathbf{M} & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{c} \\ \eta \\ \gamma \end{bmatrix} = \begin{bmatrix} 0 \\ \mathbf{F} \\ 0 \end{bmatrix}.$$

Here γ and η are vectors of Lagrange multiplier coefficients. We obtain the least square solution to the singular linear system above using the iterative method discussed in section 2. Lemma 3.2 implies that $\mathbf{E}$ is symmetric and positive definite with respect to

$$\begin{bmatrix} \mathbf{K} \\ \mathbf{M} \end{bmatrix}.$$

By Theorem 2.2, the iterative method converges to the vector $\mathbf{c}$, which is the coefficient vector of the unique interpolating spline minimizing (3.2). This furnishes a computational algorithm.

To solve the interpolation problem over $N_d^r(\Delta)$, we proceed similarly. Consider $s = s_1 + s_0$ with splines s_1 and s_0 of degrees d and $d-1$, respectively. Order the coefficients over each triangle τ as above and denote them by $\mathbf{c}_1^\tau$ and $\mathbf{c}_0^\tau$. Let $\tilde{\mathbf{c}} = (\mathbf{c}_1, \mathbf{c}_0)$, with $\mathbf{c}_1 := (\mathbf{c}_1^\tau,\ \tau \in \Delta)$ and $\mathbf{c}_0 := (\mathbf{c}_0^\tau,\ \tau \in \Delta)$. We denote interpolation, smoothness and energy matrices by $\mathbf{K}_1$, $\mathbf{K}_0$, $\mathbf{M}_1$, $\mathbf{M}_0$, $\mathbf{E}_1$, $\mathbf{E}_0$ correspondingly. Therefore interpolation conditions for s are

$$\tilde{\mathbf{K}}\tilde{\mathbf{c}} := \begin{bmatrix} \mathbf{K}_1 & \mathbf{K}_0 \end{bmatrix} \begin{bmatrix} \mathbf{c}_1 \\ \mathbf{c}_0 \end{bmatrix} = \mathbf{F}.$$

For s to be smooth we require both s_1 and s_0 to be smooth. The C^r smoothness conditions for s are

$$\tilde{\mathbf{M}}\tilde{\mathbf{c}} := \begin{bmatrix} \mathbf{M}_1 & 0 \\ 0 & \mathbf{M}_0 \end{bmatrix} \begin{bmatrix} \mathbf{c}_1 \\ \mathbf{c}_0 \end{bmatrix} = 0.$$

With definition (3.3) it is clear that the energy matrix in this case is defined by

$$\tilde{\mathbf{E}} = \begin{bmatrix} \lambda\mathbf{E}_1 & 0 \\ 0 & (1-\lambda)\mathbf{E}_0 \end{bmatrix}.$$

Therefore $s \in N_d^r(\Delta)$ minimizes (3.3), interpolates f at the vertices of Δ, and is C^r continuous if and only if the vector $\tilde{\mathbf{c}}$ of its coefficients satisfies the system of linear equations

$$\begin{bmatrix} \tilde{\mathbf{E}} & \tilde{\mathbf{K}}^T & \tilde{\mathbf{M}}^T \\ \tilde{\mathbf{K}} & 0 & 0 \\ \tilde{\mathbf{M}} & 0 & 0 \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{c}} \\ \gamma \\ \eta \end{bmatrix} = \begin{bmatrix} 0 \\ \mathbf{F} \\ 0 \end{bmatrix}.$$

TABLE 3.1
Linear and constant polynomial reproduction on eight triangles.

$S_d^r(\Delta_1) \setminus f$	1	$x+z$	$z+1$
$S_3^1(\Delta_1)$	$4.2265e-01$	$1.1016e-15$	$2.1144e-01$
$S_4^1(\Delta_1)$	$4.6629e-15$	$2.5398e-01$	$9.1140e-02$
$N_4^1(\Delta_1)$	$6.4389e-15$	$1.4950e-15$	$1.5551e-15$

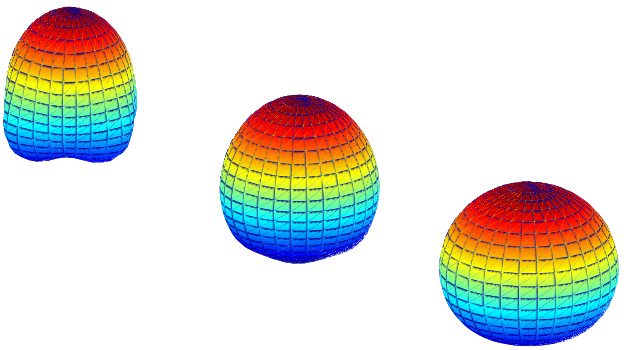


FIG. 3.1. *Interpolation of $z+1$ in S_3^1, S_4^1, and N_4^1 from left to right.*

As in the case of homogeneous splines, the linear system is singular. However, by Lemma 3.4 we have the following.

COROLLARY 3.6. $\tilde{\mathbf{E}}$ [illegible]

$$\begin{bmatrix} \tilde{\mathbf{K}} \\ \tilde{\mathbf{M}} \end{bmatrix}.$$

[illegible] Since $\mathcal{E}(s) = \tilde{\mathbf{c}}^T \tilde{\mathbf{E}} \tilde{\mathbf{c}} \geq 0$, $\tilde{\mathbf{c}}^T \tilde{\mathbf{E}} \tilde{\mathbf{c}} = 0$ implies that s is a linear polynomial. Zero side conditions force $s = 0$. By the linear independence of the basis functions, $\tilde{\mathbf{c}} = 0$. □

Thus by Theorem 2.2, the application of the iterative scheme allows us to successfully obtain its numerical solution.

3.3. Numerical experiments with ME splines. In this section we present several examples on scattered data interpolation using the ME method.

[illegible] 3.1. Let Δ_1 be the triangulation of the entire sphere based on six vertices $(1,0,0)$, $(0,1,0)$, $(0,0,1)$, $(-1,0,0)$, $(0,-1,0)$, and $(0,0,-1)$ and consisting of eight triangles. We sample 1, $x+z$ and $z+1$ at the six vertices, and compute ME spline interpolants in spherical spline spaces $S_3^1(\Delta)$, $S_4^1(\Delta)$, and $N_4^1(\Delta)$.

The maximal relative errors

$$\frac{\|s(w) - f(w)\|_\infty}{\|f(w)\|_\infty}$$

are computed based on the errors at 5120 points w almost evenly spaced over $\mathbb{S}^2$ and reported in Table 3.1. Note that not only linear and constant homogeneous polynomials are reproduced in $N_4^1(\Delta)$, but a nonhomogeneous polynomial $z+1$ is reproduced as well. In Figure 3.1 we present a visualization of the results of the last column of Table 3.1. It was shown in [1] that spherical linear functions are spheres through the origin. As expected from the table, the first two surfaces are not spheres.

TABLE 3.2
Dependence of ME splines on weights in $\mathcal{E}$.

$\lambda \setminus e(\Delta)$	$e(\Delta_1)$	$e(\Delta_2)$	$e(\Delta_3)$	$e(\Delta_4)$
0.1	$9.4421e-02$	$2.3349e-02$	$2.2270e-03$	$2.1254e-04$
0.2	$9.4363e-02$	$2.2691e-02$	$1.7986e-03$	$2.0737e-04$
0.3	$9.4292e-02$	$2.2085e-02$	$1.7570e-03$	$2.1420e-04$
0.4	$9.4222e-02$	$2.1608e-02$	$1.9870e-03$	$2.2644e-04$
0.5	$9.4141e-02$	$2.1168e-02$	$2.1526e-03$	$2.4197e-04$
0.6	$9.4053e-02$	$2.0780e-02$	$2.4118e-03$	$2.5990e-04$
0.7	$9.3948e-02$	$2.0461e-02$	$2.7717e-03$	$2.7978e-04$
0.8	$9.3833e-02$	$2.0265e-02$	$3.1331e-03$	$3.1210e-04$
0.9	$9.3701e-02$	$2.0109e-02$	$3.5004e-03$	$3.6150e-04$

TABLE 3.3
Convergence of various splines interpolating f.

$S_d^r(\Delta) \setminus e(\Delta)$	$e(\Delta_1)$	$e(\Delta_2)$	$e(\Delta_3)$	$e(\Delta_4)$
$S_3^1(\Delta)$	$3.7879e-01$	$6.5860e-02$	$3.7846e-03$	$2.9833e-04$
$S_4^1(\Delta)$	$8.2341e-02$	$1.9801e-02$	$3.8708e-03$	$4.1190e-04$
$N_4^1(\Delta)$	$9.3702e-02$	$2.0109e-02$	$1.7570e-03$	$2.0737e-04$

EXAMPLE 3.2. Next we investigate how the choice of λ affects the error in minimization over $N_d^r(\Delta)$. We interpolate $f(x,y,z) = 1 + 0.3x^8 + e^{0.2y^3}$ in $N_4^1(\Delta_i)$, $i = 1,\ldots,4$. The initial triangulation is Δ_1. The triangulation Δ_2 is obtained by bisecting the edges of Δ_1 and splitting each triangle into four subtriangles. Similarly we obtain uniform refinements Δ_3 and Δ_4. Each time, we evaluate the spline interpolant at 5120 almost evenly spaced points w and list maximal relative errors

$$e(\Delta_i) := \frac{\|s(w) - f(w)\|_\infty}{\|f(w)\|_\infty}$$

in Table 3.2. The results suggest that the errors on each Δ_i are almost the same for all the values of λ.

EXAMPLE 3.3. We compare the interpolation results for the function $f(x,y,z) = 1 + 0.3x^8 + e^{0.2y^3}$ in nonhomogeneous and homogeneous spaces. We use Table 3.2 as a guide for the choice of λ. The results in Table 3.3 demonstrate that on finer triangulations nonhomogeneous splines approximate the original function f better than homogeneous splines.

EXAMPLE 3.4. This is an example of scattered data spline interpolation. We are given 868 points in $\mathbb{R}^3$ (courtesy of Thomas Grandine at Boeing). We translate the point cloud so that its center coincides with the origin and project translated points onto the unit sphere. The projections give us locations on the sphere surface and the distances between the origin and translated points give us corresponding experimental function values. The locations on the sphere are triangulated, and the ME homogeneous cubic and quartic splines are computed as well as nonhomogeneous quartic splines. In Figure 3.2 we present one of the solutions together with the translated point cloud.

EXAMPLE 3.5. We present an example of scattered data interpolation over the Earth. We are given a satellite data set of simulated geopotential values (in m^2/s^2) observed by the gravity mission satellite, CHAMP [13], along its orbit for two days. The data collected amount to 5760 values. The CHAllenging Mini-satellite Payload (CHAMP) is a German geoscience satellite, launched on July 15, 2000, with a cir-

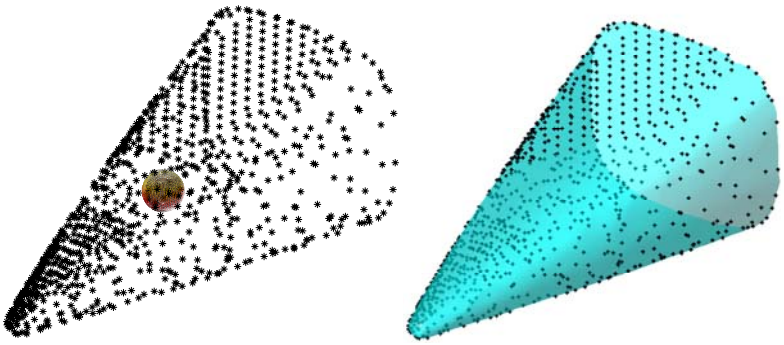


FIG. 3.2. *ME spherical spline interpolant of the point cloud.*

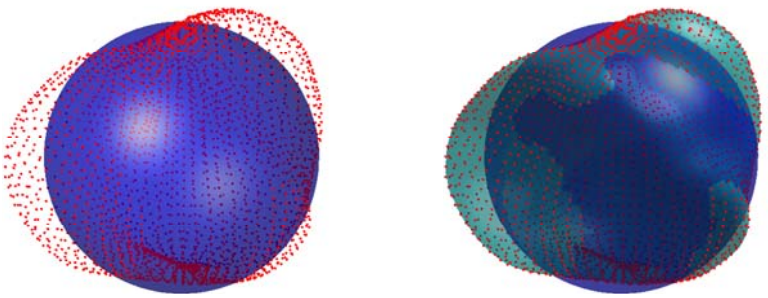


FIG. 3.3. *CHAMP geopotential data and ME spherical spline interpolant.*

cular orbit at an altitude of 450 km and orbital inclination of 87°. The simulated geopotential values are computed using a global geopotential model, EGM96 [11]. In Figure 3.3 we show the set of CHAMP scattered potential values which are normalized by the radius of the Earth and plotted over the unit sphere. The minimal energy homogeneous spherical interpolatory spline surface is shown in Figure 3.3 (right). It is clear that the spherical spline interpolates the given data and produces a reasonable fitting of the given geopotential values.

4. Discrete least squares fitting.

4.1. Existence and uniqueness. When the given data set is extremely large, e.g., $n \geq 10{,}000$, and highly redundant, it is suitable to find a discrete least squares (DLS) fitting to the given data instead of computing an interpolating spherical spline. The DLS approximation problem in $S_d^\tau(\Delta)$ can be described as follows. Let $\mathcal{V} = \{v_\ell, \ell = 1, \dots, n\}$ be the given data sites over $\mathbb{S}^2$ and Δ be a triangulation whose vertices may not relate to the data locations. For a given degree d, we assume that the data sites are rich enough in the following sense.

DEFINITION 4.1. [illegible] v_ℓ $\ell = 1, \dots, n$ [illegible] Δ [illegible] d [illegible]

$$\left[B_{ijk}^{d,\tau}(v_\ell)\right]_{i+j+k=d, v_\ell \in \tau}$$

[illegible] $\tau \in \Delta$

Suppose that the given data values are from a function f, i.e., $f(v_\ell) =: f_\ell$, $\ell = 1, \ldots, n$ are given. The least squares functional is defined by

$$\mathcal{L}(s) = \sum_{\ell=1}^{n} (s(v_\ell) - f_\ell)^2. \tag{4.1}$$

A DLS spherical spline $s_f \in S_d^r(\Delta)$ is the function minimizing $\mathcal{L}(s)$ over $S_d^r(\Delta)$, i.e.,

$$\mathcal{L}(s_f) = \min\{\mathcal{L}(s),\ s \in S_d^r(\Delta)\}.$$

THEOREM 4.2. [illegible] v_ℓ $\ell = 1, \ldots, n$ [illegible] Δ [illegible] d [illegible] s_f [illegible] d [illegible] r [illegible] (4.1) [illegible]. Recall that any $s \in S_d^r(\Delta)$ can be written as

$$s(v)|_\tau = \sum_{i+j+k=d} c_{ijk}^\tau B_{ijk}^{d,\tau}(v)$$

on a spherical triangle $\tau \in \Delta$. Let $\mathbf{c} = (c_{ijk}^\tau,\ i+j+k=d,\ \tau \in \Delta)$ be the coefficient vector of s. Note that

$$L(\mathbf{c}) := \mathcal{L}(s) = \sum_{\ell=1}^{n} |s(v_\ell) - f_\ell|^2 = \sum_{\tau \in \Delta} \sum_{v_\ell \in \tau} \left(\sum_{i+j+k=d} c_{ijk}^\tau B_{ijk}^{d,\tau}(v_\ell) - f_\ell \right)^2$$

is a continuous convex function of $\mathbf{c}$, $L(0) = \|\mathbf{f}\|_2^2$ with $\mathbf{f} = (f_\ell,\ \ell = 1, \ldots, n)$ the data value vector and $\|\mathbf{f}\|_2 := (\sum_{\ell=1}^n |f_\ell|^2)^{1/2}$ denoting the standard ℓ_2 norm of the vector $\mathbf{f}$. Consider $A = \{\mathbf{c},\ L(\mathbf{c}) \le \|\mathbf{f}\|_2^2\}$ and let us show that A is a bounded and closed set.

Fix any triangle $\tau \in \Delta$. For any $\mathbf{c} \in A$ we have

$$\left| \sum_{i+j+k=d} c_{ijk}^\tau B_{ijk}^{d,\tau}(v_\ell) - f_\ell \right|^2 \le \|\mathbf{f}\|_2^2 \quad \text{for all } v_\ell \in \tau.$$

It follows that

$$\left| \sum_{i+j+k=d} c_{ijk}^\tau B_{ijk}^{d,\tau}(v_\ell) \right| \le 2\|\mathbf{f}\|_2 \quad \text{for all } v_\ell \in \tau.$$

Since the data sites are evenly distributed with respect to d, the matrix

$$\left[B_{ijk}^{d,\tau}(v_\ell) \right]_{i+j+k=d, v_\ell \in \tau}$$

is of full rank and hence, there exists an index set $I_\tau \subset \{1, \ldots, n\}$ such that the square matrix $B^\tau = [B_{ijk}^{d,\tau}(v_\ell)]_{i+j+k=d, \ell \in I_\tau}$ is invertible. Therefore $\|(c_{ijk,i+j+k=d}^\tau)\|_2 \le C_\tau$, with C_τ a positive constant depending only on $\|\mathbf{f}\|_2$ and the norm of the inverse matrix of B^τ. Hence $\|\mathbf{c}\|_2$ is bounded above and A is bounded. It is easy to see that A is closed and that $A_s := \{\mathbf{c} : \mathbf{M}\mathbf{c} = 0\}$ is also closed. Here $\mathbf{M}\mathbf{c} = 0$ is the linear system representing the smoothness conditions for $S_d^r(\Delta)$. Hence, the set $A \cap A_s$ is compact, and there exists a $\mathbf{c}_f \in A \cap A_s$ minimizing $L(\mathbf{c})$.

To show uniqueness of the solution $\mathbf{c}_f$, we assume that there exist two solutions $\mathbf{c}_f$ and $\widehat{\mathbf{c}}_f$. The convexity of L implies that for any $0 \le \nu \le 1$ a convex combination $\mathbf{c}_f + \nu(\widehat{\mathbf{c}}_f - \mathbf{c}_f)$ also minimizes L. Thus

$$\begin{aligned}
&\frac{d}{d\nu} L(\mathbf{c}_f + \nu(\widehat{\mathbf{c}}_f - \mathbf{c}_f)) \\
&= \sum_{\tau\in\Delta}\sum_{v_\ell\in\tau}\left(\sum_{i+j+k=d}(\mathbf{c}^\tau_{ijk} + \nu(\widehat{\mathbf{c}}^\tau_{ijk} - \mathbf{c}^\tau_{ijk}))B^\tau_{ijk}(v_\ell) - f_\ell)(\widehat{\mathbf{c}}^\tau_{ijk} - \mathbf{c}^\tau_{ijk})B^\tau_{ijk}(v_\ell)\right) \\
&= \nu\sum_{\tau\in\Delta}\sum_{v_\ell\in\tau}\left(\sum_{i+j+k=d}(\widehat{\mathbf{c}}^\tau_{ijk} - \mathbf{c}^\tau_{ijk})^2 B^\tau_{ijk}(v_\ell)^2\right) \\
&+ \sum_{\tau\in\Delta}\sum_{v_\ell\in\tau}\left(\sum_{i+j+k=d}\mathbf{c}^\tau_{ijk}(\widehat{\mathbf{c}}^\tau_{ijk} - \mathbf{c}^\tau_{ijk})B^\tau_{ijk}(v_\ell)\right)^2 \\
&- \sum_{\tau\in\Delta}\sum_{v_\ell\in\tau}\left(\sum_{i+j+k=d} f_\ell(\widehat{\mathbf{c}}^\tau_{ijk} - \mathbf{c}^\tau_{ijk})B^\tau_{ijk}(v_\ell)\right) = 0
\end{aligned}$$

for any $0 \le \nu \le 1$. Since this expression is independent of ν, we have

$$\sum_{\tau\in\Delta}\sum_{v_\ell\in\tau}\left(\sum_{i+j+k=d}(\widehat{\mathbf{c}}^\tau_{ijk} - \mathbf{c}^\tau_{ijk})^2 B^\tau_{ijk}(v_\ell)\right)^2 = 0.$$

Since the data sites are evenly distributed over Δ, $\mathbf{c}_f = \widehat{\mathbf{c}}_f$. □

For nonhomogeneous spherical splines DLS approximation can be treated similarly. We seek a function $s = s_1 + s_0 \in N^r_d(\Delta)$ minimizing $\mathcal{L}(s)$.

For the nonhomogeneous case, Definition 4.1 has to be modified to take into account that the basis functions in $N^r_d(\Delta)$ consist of homogeneous SBB-basis polynomials of degrees d and $d-1$.

DEFINITION 4.3. [illegible] v_ℓ $\ell = 1, \dots, n$ [illegible] Δ [illegible] d [illegible]

$$\begin{bmatrix} B^{d,\tau}_{ijk}(v_\ell) \\ B^{d-1,\tau}_{i'j'k'}(v_\ell) \end{bmatrix}_{i+j+k=d, i'+j'+k'=d-1, v_\ell\in\tau}$$

[illegible] $\tau \in \Delta$

THEOREM 4.4. [illegible] v_ℓ $\ell = 1, \dots, n$ [illegible] d [illegible] $s_f \in N^r_d(\Delta)$ [illegible]
(4.1)

[illegible]. The proof is similar to the proof of Theorem 4.2. □

4.2. Computational algorithms. We first explain a computational algorithm for the DLS fit in $S^r_d(\Delta)$. The Lagrange multipliers method leads to the linear system

$$\begin{bmatrix} \mathbf{L}^T\mathbf{L} & \mathbf{M}^T \\ \mathbf{M} & 0 \end{bmatrix}\begin{bmatrix} \mathbf{c} \\ \eta \end{bmatrix} = \begin{bmatrix} \mathbf{L}^T\mathbf{F} \\ 0 \end{bmatrix},$$

where $\mathbf{L}$ is the observation matrix with entries $\mathbf{L}_{ij} = B_j(v_i)$, $i = 1, \dots, n$, and j runs from 1 to $\#\tau(d+1)(d+2)/2$, where $\#\tau$ denotes the number of triangles in Δ.

Here, $\mathbf{F}$ is a vector of function values ordered as the spherical points v_ℓ, $\ell = 1, \ldots, n$, $\mathbf{M}$ is the C^r smoothness matrix, and η is a vector of Lagrange multipliers. The least squares solution of this system is a vector $\mathbf{c}$ of coefficients of a homogeneous spline s of degree d and smoothness r defined with respect to the spherical triangulation Δ minimizing (4.1).

To find the DLS spline in $N_d^r(\Delta)$ we construct the observation matrix

$$\tilde{\mathbf{L}} = \begin{bmatrix} \mathbf{L}_1 & \mathbf{L}_0 \end{bmatrix}$$

and smoothness conditions $\tilde{\mathbf{M}}\tilde{\mathbf{c}} = 0$ as in the previous section. Here $\mathbf{L}_1$ and $\mathbf{L}_0$ are the observation matrices containing values of SBB-basis polynomials at the data sites for the spaces $S_d^{-1}(\Delta)$ and $S_{d-1}^{-1}(\Delta)$. We therefore solve the linear system

$$\begin{bmatrix} \tilde{\mathbf{L}}^T\tilde{\mathbf{L}} & \tilde{\mathbf{M}}^T \\ \tilde{\mathbf{M}} & 0 \end{bmatrix} \begin{bmatrix} c \\ \eta \end{bmatrix} = \begin{bmatrix} \tilde{\mathbf{L}}^T\mathbf{F} \\ 0 \end{bmatrix},$$

where $\mathbf{F}$ contains given data values and η is a vector of Lagrange multiplier coefficients.

THEOREM 4.5. [illegible] v_ℓ $\ell = 1, \ldots, n$ [illegible] Δ [illegible] d [illegible] $\tilde{\mathbf{L}}^T\tilde{\mathbf{L}}$ [illegible] $\tilde{\mathbf{M}}\tilde{\mathbf{c}} = 0$

[illegible] If $\tilde{\mathbf{c}}^T\tilde{\mathbf{L}}^T\tilde{\mathbf{L}}\tilde{\mathbf{c}} = 0$, then $\tilde{\mathbf{L}}\tilde{\mathbf{c}} = 0$. Then $\tilde{\mathbf{L}}\tilde{\mathbf{c}}|_\tau = 0$ for every $\tau \in \Delta$, where $\tilde{\mathbf{L}}|_\tau$ is of full rank and therefore $\tilde{\mathbf{c}} = 0$. □

Note that similar considerations apply to the homogeneous case; i.e. it is easy to see that the matrix $\mathbf{L}^T\mathbf{L}$ is positive definite with respect to $\mathbf{Mc} = 0$.

Hence Theorem 2.2 can be applied to find the DLS fittings in both homogeneous and nonhomogeneous spherical spline spaces.

4.3. Numerical experiments with DLS splines. The following are examples of DLS approximation on the sphere.

[illegible] 4.1. First we conduct experiments similar to the ones for ME splines in $S_3^1(\Delta_1)$, $S_4^1(\Delta_1)$, and $N_4^1(\Delta_1)$. The total number of scattered data points is 1006. Evaluation points and computation of errors are the same as in section 3.3. In addition we test higher degree polynomials to demonstrate the ability of $N_d^r(\Delta)$ to reproduce nonhomogeneous polynomials of degree d. Results are shown in Table 4.1.

[illegible] 4.2. We illustrate the convergence of DLS splines approximating $f(x, y, z) = 1 + 0.3x^8 + e^{0.2y^3}$ in Table 4.2.

[illegible] 4.3. We continue working with the data in Example 3.5. Recall Δ_1 is a triangulation based on six vertices and eight triangles as in Examples 3.1, 3.2, and 4.1.

TABLE 4.1
Polynomial reproduction over the unit sphere.

$S_d^r(\Delta_1) \setminus f$	1	$x+z$	$z+1$
$S_3^1(\Delta_1)$	$4.1063e-01$	$5.3912e-10$	$2.0543e-01$
$S_4^1(\Delta_1)$	$2.4365e-09$	$6.3255e-02$	$2.8532e-02$
$N_4^1(\Delta_1)$	$9.4194e-14$	$3.3859e-12$	$9.9751e-14$
$S_d^r(\Delta_1) \setminus f$	y^2+z	y^3+z+1	x^4+z+1
$S_3^1(\Delta_1)$	$1.5315e-01$	$1.8931e-01$	$1.7120e-01$
$S_4^1(\Delta_1)$	$4.5673e-02$	$2.8735e-02$	$2.6810e-02$
$N_4^1(\Delta_1)$	$1.1709e-13$	$1.2950e-13$	$1.5834e-13$

TABLE 4.2
Relative errors for splines approximating f.

$S_d^r(\Delta)$ \ $e(\Delta)$	$e(\Delta_1)$	$e(\Delta_2)$	$e(\Delta_3)$
$S_3^1(\Delta)$	$3.4124e-01$	$4.1755e-02$	$3.6864e-03$
$S_4^1(\Delta)$	$2.3321e-02$	$1.8815e-03$	$7.4771e-04$
$N_4^1(\Delta)$	$1.0102e-02$	$1.8007e-03$	$3.6840e-04$

TABLE 4.3
The relative errors for geodata approximating splines.

$S_d^r(\Delta)$ \ $e(\Delta)$	$e(\Delta_1)$	$e(\Delta_2)$	$e(\Delta_3)$
$S_3^1(\Delta)$	$3.7228e-01$	$2.0086e-01$	$8.6921e-02$
$S_4^1(\Delta)$	$2.9349e-01$	$9.6814e-02$	$4.5916e-02$
$N_4^1(\Delta)$	$2.0711e-01$	$9.5309e-02$	$3.1303e-02$

TABLE 4.4
The relative standard deviations for geodata approximating splines.

$S_d^r(\Delta)$ \ $\mathrm{s}(\Delta)$	$\mathrm{s}(\Delta_1)$	$\mathrm{s}(\Delta_2)$	$\mathrm{s}(\Delta_3)$
$S_3^1(\Delta)$	$7.020e-02$	$3.5489e-02$	$1.3247e-02$
$S_4^1(\Delta)$	$4.7324e-02$	$1.4640e-02$	$4.4683e-03$
$N_4^1(\Delta)$	$3.5713e-02$	$1.1194e-02$	$3.2933e-03$

Then Δ_1 is refined uniformly twice to obtain Δ_2 and Δ_3. For each triangulation we compute DLS spline solutions in the spaces $S_3^1(\Delta_i)$, $S_4^1(\Delta_i)$, and $N_4^1(\Delta_i)$, $i = 1, 2, 3$. In Table 4.3 we list errors of

$$e(\Delta_i) := \frac{\max_{v\in\mathcal{V}}|s(v) - f(v)|}{\max_{v\in\mathcal{V}}|f(v)|}$$

for each of the computed splines over triangulation Δ_i. In Table 4.4 we list their relative standard deviation values

$$\mathrm{s}(\Delta_i) := \frac{\mathrm{std}|s(v) - f(v)|}{\max_{v\in\mathcal{V}}|f(v)|}.$$

5. Penalized least squares approximation.

5.1. Existence and uniqueness. Again we let $\mathcal{V} := \{v_\ell,\, \ell = 1, \ldots, n\}$ be a set of sites on $\mathbb{S}^2$ and $\{f_\ell,\, \ell = 1, \ldots, n\}$ be the corresponding values for some function f. We need to find a smooth surface resembling f. Another commonly used method in this situation is called a penalized least squares (PLS) fit.

Let Δ be a regular triangulation of $\mathbb{S}^2$ whose vertices $\mathcal{W}$ form a subset of the data sites $\mathcal{V}$. Consider the spline space $S_d^r(\Delta)$ of degree d and smoothness r. We look for a spline solution $s_f \in S_d^r(\Delta)$ satisfying

$$\mathcal{P}_\lambda(s_f) = \min\{\mathcal{P}_\lambda(s) : s \in S_d^r(\Delta)\}, \tag{5.1}$$

where λ is a positive weight and

$$\mathcal{P}_\lambda(s) := \mathcal{L}(s) + \lambda\mathcal{E}_\delta(s). \tag{5.2}$$

Here the DLS and ME functionals are as defined in (4.1) and (3.2), respectively. It is clear that for large $\lambda \gg 1$, s_f is close to ME splines, and for small $\lambda \ll 1$ the solution s_f is close to the DLS spline. One way to choose λ is by the cross validation method [15]. We choose a small value for λ to get a good approximation, such as that of the DLS fitting which in the planar setting has high approximation power [9].

THEOREM 5.1. [illegible] $\lambda > 0$ [illegible] $\mathcal{W}$ [illegible] Δ [illegible] $\mathcal{V}$ [illegible] $\leq 1/2$ [illegible] $|\tau|$ [illegible] 1 [illegible] $s_f \in S_d^r(\Delta)$ [illegible] (5.2)

[illegible]. Recall that any $s \in S_d^r(\Delta)$ can be written as

$$s(v)|_\tau = \sum_{i+j+k=d} c_{ijk}^\tau B_{ijk}^{d,\tau}(v)$$

on a spherical triangle $\tau \in \Delta$. Let $\mathbf{c} = (c_{ijk}^\tau,\ i+j+k=d,\ \tau \in \Delta)$ be the coefficient vector of s. Recall that the energy functional $\mathcal{E}(s)$ can be expressed in terms of $\mathbf{c}$ as

$$\mathcal{E}(s) = \mathbf{c}^T \mathbf{E} \mathbf{c}$$

with the entries of $\mathbf{E}$ defined in (3.4). The DLS functional $\mathcal{L}(s)$ is expressed as

$$\begin{aligned}\mathcal{L}(s) &= \sum_{\ell=1}^n |s(v_\ell) - f_\ell|^2 = \sum_{\tau\in\Delta} \sum_{v_\ell \in \tau} \left(\sum_{i+j+k=d} c_{ijk}^\tau B_{ijk}^\tau(v_\ell) - f_\ell \right)^2 \\ &= \mathbf{c}^T \mathbf{L}^T \mathbf{L} \mathbf{c} - 2\mathbf{f}^T \mathbf{L} \mathbf{c} + \|\mathbf{f}\|_2^2,\end{aligned}$$

with $\mathbf{f} = (f_\ell,\ \ell = 1, \ldots, n)$ being the vector of data values. Thus

$$\mathcal{P}_\lambda(s) = \lambda \mathbf{c}^T \mathbf{E} \mathbf{c} + \mathbf{c}^T \mathbf{L}^T \mathbf{L} \mathbf{c} - 2\mathbf{f}^T \mathbf{L} \mathbf{c} + \|\mathbf{f}\|_2^2.$$

Note that $\mathcal{P}_\lambda(0) = \|\mathbf{f}\|_2^2$. Consider $A = \{\mathbf{c}, \mathcal{P}_\lambda(s) \leq \|\mathbf{f}\|_2^2\}$. Let us show that A is a bounded and closed set so that the continuous function $\mathcal{P}_\lambda(s)$ has a minimum in A.

Fix $\mathbf{c} \in A$ and let s be the corresponding spline. Then $\mathcal{P}_\lambda(s) \leq \|\mathbf{f}\|_2^2$. By the definition of $\mathcal{P}_\lambda$ we must have $\lambda \mathcal{E}_\delta(s) \leq \|\mathbf{f}\|_2^2$. By Lemma 10 in [6] and Lemma 3.9 in [5] the energy of a spline is equivalent to the square of its second order Sobolev seminorm on every triangle of Δ and all norms of a spline on each triangle are equivalent; i.e., we have

$$|s|_{2,\infty,\tau} \leq \frac{C_1}{\sqrt{\lambda}} \|\mathbf{f}\|_2 \text{ and } |s|'_{2,\infty,\tau} \leq \frac{C_2}{\sqrt{\lambda}} \|\mathbf{f}\|_2$$

with C_1, C_2 depending on degree d of the spline space and the smallest angle in τ. Let r_τ denote the center of the smallest spherical cap containing τ. Let $\mathbf{T}_\tau$ be a plane tangent to τ at r_τ. Define $\bar{\tau}$ in this plane as a set of points $\{w : \frac{w}{|w|} \in \tau\}$. Define $s_\delta(w) = |w|^\delta s(\frac{w}{|w|})$ to be a homogeneous extension of s of degree δ and $\bar{s}_\delta$ to be its restriction to $\bar{\tau}$. Similarly define f_δ and $\bar{f}_\delta$. By Proposition 2.26 in [5]

$$|\bar{s}_1|_{2,\infty,\bar{\tau}} \leq C_3 |s|_{2,\infty,\tau} \text{ and } |\bar{s}_0|_{2,\infty,\bar{\tau}} \leq C_4 |s|'_{2,\infty,\tau}.$$

Therefore

$$|\bar{s}_\delta|_{2,\infty,\bar{\tau}} \leq \frac{C_5}{\sqrt{\lambda}} \|\mathbf{f}\|_2. \tag{5.3}$$

Since the vertices, say v_1, v_2, v_3, of τ belong to $\mathcal{W}$,

$$\begin{aligned}|\bar{s}_\delta(\bar{v}_i)| &\le |\bar{s}_\delta(\bar{v}_i) - \bar{f}_\delta(\bar{v}_i)| + |\bar{f}_\delta(\bar{v}_i)| \\ &\le |v_i|^\delta(|s(v_i) - f(v_i)| + |f(v_i)|) \\ &\le C_6((\mathcal{P}_\lambda(s))^{1/2} + \|\mathbf{f}\|_2) \le 2C_6\|\mathbf{f}\|_2,\end{aligned}$$

for $C_6 = \max\{|v_\ell|^\delta : v_\ell \in \tau\} \le 1/\cos(1/2)$ and $i = 1, 2, 3$. For any point $\bar{v}$ in $\bar{\tau}$ we need to show that $\bar{s}_\delta(\bar{v})$ is bounded. Use Taylor expansion to get

$$\bar{s}_\delta(\bar{v}_1) = \bar{s}_\delta(\bar{v}) + \nabla\bar{s}_\delta(\bar{v}) \cdot (\bar{v}_1 - \bar{v}) + O(|\bar{s}_\delta|_{2,\infty,\bar{\tau}}|\bar{\tau}|^2). \tag{5.4}$$

Using similar expressions for $\bar{v}_2$ and $\bar{v_3}$ we get

$$\begin{aligned}\bar{s}_\delta(\bar{v}_1) - \bar{s}_\delta(\bar{v}_2) &= \nabla\bar{s}_\delta(\bar{v}) \cdot (\bar{v}_1 - \bar{v}_2) + O(|\bar{s}_\delta|_{2,\infty,\bar{\tau}}|\bar{\tau}|^2), \\ \bar{s}_\delta(\bar{v}_2) - \bar{s}_\delta(\bar{v_3}) &= \nabla\bar{s}_\delta(\bar{v}) \cdot (\bar{v}_2 - \bar{v_3}) + O(|\bar{s}_\delta|_{2,\infty,\bar{\tau}}|\bar{\tau}|^2).\end{aligned}$$

Solving this linear system for $\nabla\bar{s}_\delta$, we get

$$\begin{aligned}D_x\bar{s}_\delta(\bar{v}) &= O(|\bar{\tau}|^3|\bar{s}_\delta|_{2,\infty,\bar{\tau}}/A_{\bar{\tau}}) + |\bar{s}_\delta(\bar{v}_1)| + |\bar{s}_\delta(\bar{v}_2)||\bar{\tau}|/A_{\bar{\tau}}, \\ D_y\bar{s}_\delta(\bar{v}) &= O(|\bar{\tau}|^3|\bar{s}_\delta|_{2,\infty,\bar{\tau}}/A_{\bar{\tau}}) + |\bar{s}_\delta(\bar{v}_1)| + |\bar{s}_\delta(\bar{v}_2)||\bar{\tau}|/A_{\bar{\tau}},\end{aligned}$$

where $A_{\bar{\tau}}$ denotes the area of $\bar{\tau}$. Using these estimates for $\nabla\bar{s}_\delta$ we get

$$|\bar{s}_\delta(\bar{v})| \le C_7((1 + |\bar{\tau}| + |\bar{\tau}|^2/A_{\bar{\tau}})\|\mathbf{f}\|_2 + |\bar{\tau}|^4|\bar{s}_\delta|_{2,\infty,\bar{\tau}}/A_{\bar{\tau}}).$$

Hence we use (5.3) and (5.4) to get

$$|\bar{s}_\delta(\bar{v})| \le C_8\|\mathbf{f}\|_2$$

with C_8 depending on the size of τ. By the definition,

$$|s(v)| = |\bar{v}|^{-\delta}|\bar{s}_\delta(\bar{v})| \le C_{\bar{\tau}}\|\mathbf{f}\|_2$$

is bounded since $|\bar{v}| \ge 1$ and $|\bar{\tau}|$ is bounded. By the stability of SBB-basis [12], $\mathbf{c}$ is bounded, and hence A is a bounded set. Since A is closed, it is compact. By the definition, $\mathcal{P}_\lambda(s)$ is a continuous function of $\mathbf{c}$ and therefore $\mathcal{P}_\lambda$ attains its minimum in A.

To show uniqueness of the minimizer s_f suppose there exists $\hat{s}_f$ with $\mathcal{P}_\lambda(s_f) = \mathcal{P}_\lambda(\hat{s}_f)$. Since $\mathcal{P}_\lambda$ is a convex functional for any $0 \le \nu \le 1$,

$$\mathcal{P}_\lambda(\nu s_f + (1-\nu)\hat{s}_f) \le \nu\mathcal{P}_\lambda(s_f) + (1-\nu)\mathcal{P}_\lambda(\hat{s}_f) = \mathcal{P}_\lambda(s_f).$$

On the other hand, since $\mathcal{P}_\lambda$ achieves minimum value at s_f,

$$\mathcal{P}_\lambda(s_f) \le \mathcal{P}_\lambda(\nu s_f + (1-\nu)\hat{s}_f).$$

Therefore $\mathcal{P}_\lambda(\hat{s}_f + \nu(s_f - \hat{s}_f))$ is a constant function of ν on $[0, 1]$. It follows that $\frac{d}{d\nu}\mathcal{P}_\lambda(\hat{s}_f + \nu(s_f - \hat{s}_f)) = 0$ for all $0 \le \nu \le 1$, i.e.,

$$\begin{aligned}0 &= \frac{d}{d\nu}\mathcal{P}_\lambda(\hat{s}_f + \nu(s_f - \hat{s}_f)) \\ &= 2\lambda(\hat{\mathbf{c}}_f + \nu(\mathbf{c}_f - \hat{\mathbf{c}}_f))^T\mathbf{E}(\mathbf{c}_f - \hat{\mathbf{c}}_f) \\ &\quad + 2(\hat{\mathbf{c}}_f + \nu(\mathbf{c}_f - \hat{\mathbf{c}}_f))^T\mathbf{L}^T\mathbf{L}(\mathbf{c}_f - \hat{\mathbf{c}}_f) - 2\mathbf{f}^T\mathbf{L}(\mathbf{c}_f - \hat{\mathbf{c}}_f).\end{aligned}$$

The above holds for any ν in $[0,1]$. It follows, as in Theorem 4.2, that

$$0 = 2\lambda(\mathbf{c}_f - \hat{\mathbf{c}}_f)^T\mathbf{E}(\mathbf{c}_f - \hat{\mathbf{c}}_f) + 2(\mathbf{c}_f - \hat{\mathbf{c}}_f)^T\mathbf{L}^T\mathbf{L}(\mathbf{c}_f - \hat{\mathbf{c}}_f).$$

Hence, we must have $(\mathbf{c}_f - \hat{\mathbf{c}}_f)^T\mathbf{E}(\mathbf{c}_f - \hat{\mathbf{c}}_f) = 0$ and $(\mathbf{c}_f - \hat{\mathbf{c}}_f)^T\mathbf{L}^T\mathbf{L}(\mathbf{c}_f - \hat{\mathbf{c}}_f) = 0$ since both $\mathbf{E}$ and $\mathbf{L}^T\mathbf{L}$ are nonnegative definite. Then $\mathcal{E}(s_f - \hat{s}_f) = 0$ and therefore $s_f - \hat{s}_f$ is a linear homogeneous polynomial, and $s_f(v_\ell) - \hat{s}_f(v_\ell) = 0$ at every vertex v_ℓ of Δ. Therefore $s_f = \hat{s}_f$. □

To solve the PLS problem using nonhomogeneous splines we work in $N_d^r(\Delta)$. We have to replace the energy functional (3.2) in (5.2) by (3.3). For a spherical spline function $s = s_1 + s_0$ define

$$\mathcal{P}_\lambda(s) = \mathcal{L}(s_1 + s_0) + \lambda_1 \int_{\mathbb{S}^2} \sum_{|\alpha|=2} (D^\alpha s_1)^2 d\mu + \lambda_0 \int_{\mathbb{S}^2} \sum_{|\alpha|=2} (D^\alpha s_0)^2 d\mu, \tag{5.5}$$

with $\lambda_1 > 0$, $\lambda_0 > 0$.

THEOREM 5.2. [illegible] $\lambda_1 > 0$ [illegible] $\lambda_0 > 0$ [illegible] $\mathcal{W}$ [illegible] Δ [illegible] $\mathcal{V}$ [illegible] $\max\{|\tau|, \tau \in \Delta\} =: |\Delta| \leq 1$ [illegible] $s_f \in N_d^r(\Delta)$ [illegible] (5.5)

[illegible]. The proof is similar to the proof of Theorem 5.1. □

5.2. Computational algorithms. We first consider PLS fitting in $S_d^r(\Delta)$. By the method of Lagrange multipliers, minimization of (5.2) over $S_d^{-1}(\Delta)$ subject to the smoothness C^r conditions in the matrix form $\mathbf{Mc} = 0$ results in a system of linear equations

$$\begin{bmatrix} \mathbf{P} & \mathbf{M}^T \\ \mathbf{M} & 0 \end{bmatrix} \begin{bmatrix} \mathbf{c} \\ \eta \end{bmatrix} = \begin{bmatrix} \mathbf{L}^T\mathbf{F} \\ 0 \end{bmatrix}.$$

Here $\mathbf{P} = \mathbf{L}^T\mathbf{L} + \lambda\mathbf{E}$ and $\mathbf{f}$ is a vector of function values ordered as the spherical points v_ℓ, $\ell = 1, \ldots, n$, $\mathbf{M}$ is a smoothness matrix, and η is a vector of Lagrange multiplier coefficients. The solution of this system is a vector $\mathbf{c}$ of coefficients of a homogeneous spline s of degree d and smoothness r defined over the spherical triangulation Δ minimizing (5.2). Note that the linear system is of the same form as the one in section 2. We use the iterative scheme to compute an approximation of $\mathbf{c}$.

To find the PLS spline in $N_d^r(\Delta)$ we construct the observation matrix $\tilde{\mathbf{L}}$ and smoothness conditions $\tilde{\mathbf{M}}\tilde{\mathbf{c}} = 0$ as in the setting of DLS splines. We also assemble the energy matrices $\mathbf{E}_1$ and $\mathbf{E}_0$ and solve the linear system

$$\begin{bmatrix} \tilde{\mathbf{P}} & \tilde{\mathbf{M}}^T \\ \tilde{\mathbf{M}} & 0 \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{c}} \\ \eta \end{bmatrix} = \begin{bmatrix} \tilde{\mathbf{L}}^T\mathbf{f} \\ 0 \end{bmatrix},$$

where $\tilde{\mathbf{P}}$ in this case is

$$\tilde{\mathbf{L}}^T\tilde{\mathbf{L}} + \begin{bmatrix} \lambda_1\mathbf{E}_1 & 0 \\ 0 & \lambda_0\mathbf{E}_0 \end{bmatrix}.$$

COROLLARY 5.3. [illegible] $\tilde{\mathbf{P}}$ [illegible] $\tilde{\mathbf{M}}\tilde{\mathbf{c}} = 0$ [illegible] $\tilde{\mathbf{c}}^T\tilde{\mathbf{P}}\tilde{\mathbf{c}} = 0$ implies that $\tilde{\mathbf{c}}^T\tilde{\mathbf{E}}\tilde{\mathbf{c}} = 0$. Therefore $\mathcal{P}_\lambda(s) = 0$ implies that s is a linear polynomial. $\mathcal{L}(s) = 0$ as well, and therefore s interpolates $f = 0$ at every vertex of Δ. Hence $s = 0$ and so is $\tilde{\mathbf{c}} = 0$. □

Note that considerations of Corollary 5.3 apply to the homogeneous case, and therefore we can efficiently obtain homogeneous PLS spline solutions as well.

TABLE 5.1
The relative errors for PLS splines.

$S_d^r(\Delta) \setminus e(\Delta)$	$e(\bar{\Delta}_1)$	$e(\bar{\Delta}_2)$	$e(\bar{\Delta}_3)$
$S_3^1(\Delta)$	$7.3001e-01$	$2.5346e-01$	$1.0829e-01$
$S_4^1(\Delta)$	$3.2211e-01$	$1.1959e-01$	$4.4371e-02$
$N_4^1(\Delta)$	$2.5707e-01$	$9.4001e-02$	$4.2157e-02$

TABLE 5.2
The relative standard deviations for PLS splines.

$S_d^r(\Delta) \setminus \mathrm{s}(\Delta)$	$\mathrm{s}(\bar{\Delta}_1)$	$\mathrm{s}(\bar{\Delta}_2)$	$\mathrm{s}(\bar{\Delta}_3)$
$S_3^1(\Delta)$	$1.3024e-01$	$4.4337e-02$	$1.4585e-02$
$S_4^1(\Delta)$	$5.5051e-02$	$1.7656e-02$	$4.5250e-03$
$N_4^1(\Delta)$	$3.9482e-02$	$1.3479e-02$	$4.1267e-03$

5.3. Numerical experiments with PLS splines. In this section we present an example of a PLS fitting similar to Example 4.3.

Example 5.1. Note that for PLS fit we require the data at the vertices of a triangulation to be given. To deal with this requirement we modify the triangulation Δ_1 by replacing its vertices by points in $\mathcal{V}$ closest to them. Denote the new triangulation by $\bar{\Delta}_1$. After refining $\bar{\Delta}_1$ we again may not have the values of geopotential values available at the vertices of the refined triangulation. We replace these vertices by the points in $\mathcal{V}$ closest to them where values of geopotential are available. We call this new triangulation $\bar{\Delta}_2$. Similarly we obtain $\bar{\Delta}_3$. That is, $\bar{\Delta}_{i+1}$ is not exactly a uniform refinement of $\bar{\Delta}_i$. For each triangulation we compute PLS spline solutions in the spaces $S_3^1(\bar{\Delta}_i)$, $S_4^1(\bar{\Delta}_i)$, and $N_4^1(\bar{\Delta}_i)$, $i = 1, 2, 3$, with $\lambda = \lambda_1 = \lambda_2 = 10^{-6}$. In Table 5.1 we list relative error values

$$e(\bar{\Delta}_i) := \frac{\max_{v\in\mathcal{V}}|s(v) - f(v)|}{\max_{v\in\mathcal{V}}|f(v)|}$$

for each of the computed splines over triangulation Δ_i. In Table 5.2 we list their relative standard deviation values

$$\mathrm{s}(\bar{\Delta}_i) := \frac{\mathrm{std}|s(v) - f(v)|}{\max_{v\in\mathcal{V}}|f(v)|}.$$

REFERENCES

[1] P. ALFELD, M. NEAMTU, AND L. L. SCHUMAKER, *Bernstein-Bézier polynomials on spheres and sphere-like surfaces*, Comput. Aided Geom. Design, 13 (1996), pp. 333–349.

[2] P. ALFELD, M. NEAMTU, AND L. L. SCHUMAKER, *Fitting scattered data on sphere-like surfaces using spherical splines*, J. Comput. Appl. Math., 73 (1996), pp. 5–43.

[3] P. ALFELD, M. NEAMTU, AND L. L. SCHUMAKER, *Dimension and local bases of homogeneous spline spaces*, SIAM J. Math. Anal., 27 (1996), pp. 1482–1501.

[4] G. AWANOU AND M. J. LAI, *On convergence rate of the augmented Lagrangian algorithms for nonsymmetric saddle point problems*, Appl. Numer. Math., 54 (2005), pp. 122–134.

[5] V. BARAMIDZE, *Spherical Splines for Scattered Data Fitting*, Ph.D. dissertation, Department of Mathematics, The University of Georgia, Athens, GA, 2005.

[6] V. BARAMIDZE AND M. J. LAI, *Error bounds for minimal energy interpolatory spherical splines*, in Approximation Theory XI, C. K. Chui, M. Neamtu, and L. L. Schumaker, eds., Nashboro Press, Brentwood, 2005, pp. 25–50.

[7] G. Fasshauer and L. L. Schumaker, *Scattered data fitting on the sphere*, in Mathematical Methods for Curves and Surfaces II, M. Daehlen, T. Lyche, and L. Schumaker, eds., Vanderbilt University Press, Nashville, TN, 1998, pp. 117–166.

[8] M. von Golitschek, M. J. Lai, and L. L. Schumaker, *Error bounds for minimal energy bivariate polynomial splines*, Numer. Math., 93 (2002), pp. 315–331.

[9] M. von Golitschek and L. L. Schumaker, *Bounds on projections onto bivariate polynomial spline spaces with stable bases*, Constr. Approx., 18 (2002), pp. 241–254.

[10] A. Gomide and J. Stolfi, *Nonhomogeneous Polynomial C^k Splines on the Sphere S^n*, Relatorio Tecnico IC-00-13, Institute of Computing, University of Campinas, Brazil, 2000.

[11] F. Lemoine, N. Pavlis, S. Kenyon, R. Rapp, E. Pavlis, and B. Chao, *New high-resolution model developed for Earth's gravitational field*, EOS Trans. Amer. Geophys. Union, 79 (1998), pp. 117–118.

[12] M. Neamtu and L. L. Schumaker, *On the approximation order of splines on spherical triangulations*, Adv. Comput. Math., 21 (2004), pp. 3–20.

[13] C. Reigber, H. Jochmann, J. Wünsch, S. Petrovic, P. Schwintzer, F. Barthelmes, K. H. Neumayer, R. König, C. Förste, G. Balmino, R. Biancale, J. M. Lemonie, S. Loyer, and F. Perosanz, *Earth gravity field and seasonal variability from CHAMP*, in Earth Observation with CHAMP, C. Reigber, H. Lühr, P. Schwintzer, and J. Wickert, eds., Springer, New York, 2005, pp. 25–30.

[14] B. Tapley, S. Bettadpur, J. Ries, P. Thompson, and M. Watkins, *GRACE measurements of mass variability in the Earth system*, Science, 305 (2004), pp. 503–505.

[15] G. Wahba, *Spline Models for Observational Data*, SIAM, Philadelphia, 1990.

SIAM J. SCI. COMPUT.
Vol. 28, No. 1, pp. 260–277
© 2006 Society for Industrial and Applied Mathematics

LINEAR PDEs AND NUMERICAL METHODS THAT PRESERVE A MULTISYMPLECTIC CONSERVATION LAW*

JASON FRANK†, BRIAN E. MOORE‡, AND SEBASTIAN REICH§

Abstract. Multisymplectic methods have recently been proposed as a generalization of symplectic ODE methods to the case of Hamiltonian PDEs. Their excellent long time behavior for a variety of Hamiltonian wave equations has been demonstrated in a number of numerical studies. A theoretical investigation and justification of multisymplectic methods is still largely missing. In this paper, we study linear multisymplectic PDEs and their discretization by means of numerical dispersion relations. It is found that multisymplectic methods in the sense of Bridges and Reich [*Phys. Lett. A*, 284 (2001), pp. 184–193] and Reich [*J. Comput. Phys.*, 157 (2000), pp. 473–499], such as Gauss–Legendre Runge–Kutta methods, possess a number of desirable properties such as nonexistence of spurious roots and conservation of the sign of the group velocity. A certain CFL-type restriction on $\Delta t/\Delta x$ might be required for methods higher than second order in time. It is also demonstrated by means of the explicit midpoint method that multistep methods may exhibit spurious roots in the numerical dispersion relation for any value of $\Delta t/\Delta x$ despite being multisymplectic in the sense of discrete variational mechanics [J. E. Marsden, G. P. Patrick, and S. Shkoller, *Commun. Math. Phys.*, 199 (1999), pp. 351–395].

Key words. wave equations, dispersion relations, discretizations, multisymplectic methods, discrete conservation laws

AMS subject classifications. 65P10, 65M99

DOI. 10.1137/050628271

1. Introduction.

The field of numerical analysis has experienced a recent, as well as growing, interest in geometric integration. When solving differential equations numerically it is important to preserve as much of the qualitative solution behavior as possible. In particular, it is advantageous to numerically preserve the symplectic structure of Hamiltonian systems.

The topic of symplectic integration for Hamiltonian ODEs has been thoroughly analyzed. The concept of symplectic time integration has also been extended to Hamiltonian PDEs (see, e.g., [16]). The concept of space-time multisymplectic integration is, however, much more recent and its development is still in the beginning stages. In some respects, even the proper definition of a multisymplectic integrator is still not clear. The original approach, presented by Marsden, Patrick, and Shkoller [15], defines a multisymplectic method as one which can be derived from a Lagrangian formulation using a discrete variational principle. The original approach was limited to first-order field theories (see [12] for an extension to second-order field theories). The discrete variational approach allows the systematic derivation of one-step and multistep multisymplectic methods.

*Received by the editors April 1, 2005; accepted for publication (in revised form) August 19, 2005; published electronically March 10, 2006.

http://www.siam.org/journals/sisc/28-1/62827.html

†CWI, P.O. Box 94079, 1090 GB Amsterdam, The Netherlands (jason@cwi.nl). This author's work was supported by an Innovational Research grant from the Netherlands Organization for Scientific Research (NWO).

‡Department of Mathematics and Statistics, McGill University, Montreal, Canada (moore@math.mcgill.ca).

§Institut für Mathematik, Universität Potsdam, Postfach 60 15 53, D-14415 Potsdam, Germany (sreich@math.uni-potsdam.de).

The second approach, which was put forth by Bridges and Reich [7, 19], defines a multisymplectic integrator as a method which exactly preserves a multisymplectic conservation law by applying a symplectic one-step method such as Störmer–Verlet or implicit midpoint in space and time.

It is our intent to study multisymplectic integration methods in the context of linear multisymplectic PDEs. Based on a discussion of numerical dispersion relations, we will find that both conservation of a multisymplectic conservation law and discretization by one-step methods in space and time should be considered essential ingredients of a multisymplectic method.

According to Bridges [2, 3], a PDE of the general form

$$\mathbf{K}\mathbf{z}_t + \mathbf{L}\mathbf{z}_x = \nabla_{\mathbf{z}} S(\mathbf{z}) \tag{1.1}$$

is called multisymplectic provided $\mathbf{K}$ and $\mathbf{L}$ are (constant) $d \times d$ skew-symmetric matrices, and $S : \mathbb{R}^d \to \mathbb{R}$ is a smooth function. Here and throughout, subscripts are used to denote differentiation in the standard way, and we define $\mathbf{z} = \mathbf{z}(x,t)$ to be a d-dimensional vector of state variables. Many Hamiltonian PDEs can be formulated in this way, and the multisymplectic formulation has become useful for understanding these problems (cf. [2, 3, 4, 5, 6, 8]).

As one would expect, however, the nonlinearity of (1.1) makes the study of numerical solutions for the problem more difficult, and there is still much to understand with regard to multisymplectic discretization methods. Hence, the aim of this paper is to conduct an extensive analysis of multisymplectic discretization methods for the linear equation

$$\mathbf{K}\mathbf{z}_t + \mathbf{L}\mathbf{z}_x = \mathbf{A}\mathbf{z}, \tag{1.2}$$

where $\mathbf{A}$ is symmetric. This allows one to examine solutions and dispersion relations for both the continuous and discrete versions of this equation, providing a thorough understanding of the behavior of our numerical methods for linear problems.

We start by making the standard (complex-valued) separation of variables ansatz

$$\mathbf{z}(x,t) = u(t)v(x)\mathbf{a},$$

where $u(t)$ and $v(x)$ are complex-valued functions and $\mathbf{a}$ is a d-dimensional complex-valued vector. Substituting this ansatz into the linear PDE (1.2) leads to the linear system

$$\left(\frac{u'}{u}\mathbf{K} + \frac{v'}{v}\mathbf{L} - \mathbf{A}\right)\mathbf{a} = \mathbf{0}.$$

Solutions are found by setting

$$\lambda = \frac{u'}{u} \qquad \text{and} \qquad \mu = \frac{v'}{v}$$

with the complex numbers λ and μ satisfying

$$\det\left(\lambda\mathbf{K} + \mu\mathbf{L} - \mathbf{A}\right) = 0.$$

Since we are interested in wave-like solutions we furthermore seek (complex-valued) solutions with $\lambda = \mathtt{i}\omega$ and $\mu = \mathtt{i}k$, i.e.,

$$\mathbf{z}(x,t) = \mathrm{e}^{\mathtt{i}(kx+\omega t)}\mathbf{a}. \tag{1.3}$$

Here, k denotes the wave number and ω denotes the wave frequency, which must satisfy the dispersion relation [3]

$$\mathcal{D}(\omega, k) := \det\left(\mathtt{i}\omega\mathbf{K} + \mathtt{i}k\mathbf{L} - \mathbf{A}\right) = 0. \tag{1.4}$$

It is important to note that the matrix used in this calculation is self-adjoint, i.e., $\overline{\mathcal{D}(\omega, k)} = \mathcal{D}(\overline{\omega}, \overline{k})$, from which it is immediate that $0 = \mathcal{D}(\omega, k) = \mathcal{D}(\overline{\omega}, k)$ for real k. Hence, for any given real k, solutions ω of the dispersion relation are either real or come in complex-conjugate pairs meaning there is no diffusion [20, Chapter 11].

With the dispersion relation, one can write the frequency as a function of the wave number, such that

$$\mathcal{D}(\omega, k) = 0 \Longleftrightarrow \omega = \omega(k),$$

at least locally. Then, depending on the function $\mathcal{D}$, there may be multiple frequencies ω_i for every k, corresponding to different modes. One could also pose the reverse question and ask how many different wave numbers k can give rise to a given frequency ω.

It is worthwhile to note that taking the complex-conjugate in (1.3) also yields a solution and hence

$$\mathbf{z}(x,t) = \frac{1}{2}[\mathbf{a}\,\mathrm{e}^{\mathtt{i}(kx+\omega t)} + \overline{\mathbf{a}}\,\mathrm{e}^{-\mathtt{i}(kx+\omega t)}] \tag{1.5}$$

yields real-valued solutions. The dispersion relation is not restricted to single mode solutions (1.3). In fact, the most general oscillatory solution of the linear PDE (1.2) can be stated as a superposition of solutions

$$\begin{aligned}\mathbf{z}(x,t) = &\frac{1}{2}\sum_i \int_0^\infty \mathbf{a}_i(k)\,\mathrm{e}^{\mathtt{i}(kx-\omega_i(k)t)}\mathrm{d}k \\ &+ \frac{1}{2}\sum_i \int_0^\infty \overline{\mathbf{a}}_i(k)\,\mathrm{e}^{-\mathtt{i}(kx-\omega_i(k)t)}\mathrm{d}k, \qquad -\infty < x < \infty,\end{aligned}$$

which is obtained through Fourier analysis [20, Chapter 11]. Here the index i runs over all solution branches of the dispersion relation.

An important derived quantity is the group velocity defined by

$$v_g = \frac{d}{dk}\omega(k) = -\frac{\partial\mathcal{D}}{\partial k}\Big/\frac{\partial\mathcal{D}}{\partial\omega}.$$

The group velocity characterizes the speed of energy transport in wave packets [20, Chapter 11] and, hence, is fundamental to the understanding of linear waves. Following Whitham [20, Chapter 11], we introduce the Lagrangian density

$$\mathcal{L} = \frac{1}{2}\overline{\mathbf{a}}^T\left[\mathtt{i}\omega\mathbf{K} + \mathtt{i}k\mathbf{L} - \mathbf{A}\right]\mathbf{a} \tag{1.6}$$

and define the energy density

$$E = \omega\frac{\partial\mathcal{L}}{\partial\omega} = \frac{\mathtt{i}\omega}{2}\overline{\mathbf{a}}^T\mathbf{K}\mathbf{a}$$

and the energy flux

$$F = -\omega\frac{\partial\mathcal{L}}{\partial k} = -\frac{\mathtt{i}\omega}{2}\overline{\mathbf{a}}^T\mathbf{L}\mathbf{a}.$$

It holds that $F = v_g E$ since [2]

$$v_g = -\frac{\partial \mathcal{L}}{\partial k} \Big/ \frac{\partial \mathcal{L}}{\partial \omega} = -\frac{\bar{\mathbf{a}}^T \mathbf{L}\mathbf{a}}{\bar{\mathbf{a}}^T \mathbf{K}\mathbf{a}}.$$

We wish to extend the linear dispersion analysis to multisymplectic integration methods. As multisymplectic methods are obtained as a space-time concatenation of symplectic methods, we start the discussion with a review of results for symplectic ODE methods.

2. Symplectic integration of Hamiltonian ODEs. We recall some basic properties of symplectic integration methods for Hamiltonian ODEs of the form

$$\mathbf{K}\mathbf{z}_t = \nabla_{\mathbf{z}} S(\mathbf{z}), \tag{2.1}$$

where $\mathbf{K}$ is a (constant) $d \times d$ skew-symmetric matrix and $S : \mathbb{R}^d \to \mathbb{R}$ a smooth function. A numerical one-step method

$$\mathbf{z}^{n+1} = \Psi_{\Delta t}(\mathbf{z}^n) \tag{2.2}$$

is called symplectic if

$$\mathbf{d}\mathbf{z}^{n+1} \wedge \mathbf{K}\mathbf{d}\mathbf{z}^{n+1} = \mathbf{d}\mathbf{z}^n \wedge \mathbf{K}\mathbf{d}\mathbf{z}^n. \tag{2.3}$$

(See [13] for details and use of wedge product notation.) Let us discuss implications for the linearized equations

$$\mathbf{K}\mathbf{z}_t = \mathbf{A}\mathbf{z}, \tag{2.4}$$

where $\mathbf{A}$ is symmetric. We assume that the one-step method (2.2) is a Runge–Kutta (RK) method with stability function $R(z)$ [11]. Then the (complex-valued) numerical solution of (2.4) can be written in the form

$$\mathbf{z}^n = \sum_i R(\lambda_i \Delta t)^n \mathbf{a}_i,$$

where λ_i and $\mathbf{a}_i$ have to satisfy the generalized eigenvalue problem

$$[\lambda_i \mathbf{K} - \mathbf{A}]\, \mathbf{a}_i = 0 \tag{2.5}$$

associated with (2.4). Note that if λ is an eigenvalue of (2.5), then $-\lambda$ is also an eigenvalue. Furthermore, a symplectic RK method satisfies $R(\lambda)R(-\lambda) = 1$. The same property holds for all (not necessarily symplectic) symmetric RK methods [11].

If we now restrict the analysis to the symplectic Gauss–Legendre Runge–Kutta (GL RK) methods, then we conclude from $R(\lambda)R(-\lambda) = 1$ and A-stability of the method that $R(z)$ is on the unit circle for purely imaginary valued $\lambda = \mathtt{i}\omega$. We write this in the form

$$R(\mathtt{i}\omega\Delta t) = \mathrm{e}^{\mathtt{i}\phi},$$

with an appropriate phase function $\phi(\omega\Delta t)$ continuous in $\omega\Delta t$.

Let us explicitly derive the phase function ϕ for the implicit midpoint rule

$$\mathbf{K}\frac{\mathbf{z}^{n+1} - \mathbf{z}^n}{\Delta t} = \mathbf{A}\frac{\mathbf{z}^{n+1} + \mathbf{z}^n}{2}.$$

We write $\phi = \Omega\Delta t$ and obtain

$$\frac{\mathrm{e}^{\mathtt{i}\Omega\Delta t}-1}{\Delta t}\mathbf{K}\mathbf{a}_i = \frac{\mathrm{e}^{\mathtt{i}\Omega\Delta t}+1}{2}\mathbf{A}\mathbf{a}_i$$

for each solution mode and, using (2.5) with $\lambda_i = \mathtt{i}\omega_i$, this equation implies

$$\frac{\mathrm{e}^{\mathtt{i}\Omega\Delta t}-1}{\Delta t}\mathbf{K}\mathbf{a} = \mathtt{i}\omega_i\frac{\mathrm{e}^{\mathtt{i}\Omega\Delta t}+1}{2}\mathbf{K}\mathbf{a}.$$

Hence we must have

$$\frac{\mathrm{e}^{\mathtt{i}\Omega\Delta t}-1}{\Delta t} = \mathtt{i}\omega_i\frac{\mathrm{e}^{\mathtt{i}\Omega\Delta t}+1}{2},$$

which yields the definition for the stability function of the implicit midpoint method, i.e.,

$$\mathrm{e}^{\mathtt{i}\Omega\Delta t} = R(\mathtt{i}\omega\Delta t) = \frac{1+\mathtt{i}\omega\Delta t/2}{1-\mathtt{i}\omega\Delta t/2},$$

and

$$\phi(\omega\Delta t) = 2\tan^{-1}(\omega\Delta t/2) \in (-\pi, +\pi) \tag{2.6}$$

is a monotone function in $\omega\Delta t$.

The fourth-order GL RK method has stability function

$$R(z) = \frac{1+z/2+z^2/12}{1-z/2+z^2/12}.$$

Setting $z = \mathtt{i}\omega\Delta t$ and $R(\mathtt{i}\omega\Delta t) = e^{\mathtt{i}\phi}$ results in

$$\tan\frac{\phi}{2} = \frac{\omega\Delta t}{2-\frac{\omega^2\Delta t^2}{6}}.$$

For positive ω, the function $\phi(\omega\Delta t)$ can be chosen to increase monotonically from zero for $\omega = 0$ to $\phi(\omega\Delta t) = 2\pi$ for $\omega\Delta t \to \infty$. A similar statement holds for negative frequencies. Of course, from a sampling point of view, we have $\mathrm{e}^{\mathtt{i}\Omega\Delta t} = \mathrm{e}^{\mathtt{i}\phi}$ with

$$\Omega = \frac{1}{\Delta t}\left[\mathrm{mod}\left(\phi(\omega\Delta t)+\pi, 2\pi\right)-\pi\right] \in \left(\frac{-\pi}{\Delta t}, \frac{+\pi}{\Delta t}\right]. \tag{2.7}$$

Note that the implicit midpoint method simply yields

$$\Omega = \frac{1}{\Delta t}\phi(\omega\Delta t)$$

with phase function ϕ defined by (2.6).

For a general s-stage GL RK method of order $p = 2s$ it holds that $\phi(\omega\Delta t)$ is monotone with values in the interval $(-s\pi, +s\pi)$, $\phi(-y) = -\phi(y)$, and $\phi(y) < y$ for $y > 0$ [10]. Due to sampling on a discrete grid, i.e., (2.7), it can nevertheless happen that there are values of Δt for which $\Omega_i = \Omega_j$ even though $\omega_i \neq \omega_j$. These numerically induced 1-1 resonances can occur for $s \geq 2$ and

$$\max_i |\phi(\omega_i\Delta t)| > \pi.$$

Numerical resonance can lead to instabilities of the scheme under small perturbations [13]. Note that the implicit midpoint method and higher-order GL RK methods can be subject to other numerically induced resonance instabilities. See, e.g., Mandziuk and Schlick [14].

Our discussion of GL RK methods applied to (linear) Hamiltonian ODEs may be summarized as follows:

(i) Oscillatory solutions of (2.4) with frequency ω are represented numerically by an oscillatory solution with frequency Ω.
(ii) There is one-to-one correspondence between numerical and analytic[1] solutions modes (no spurious computational modes as for linear multistep methods).
(iii) The phase function ϕ is monotone. However, numerical resonances due to (2.7) are possible for GL RK methods with more than one stage.

It should be noted that the same statements would be true for any symmetric and A-stable implicit RK method such as Lobatto IIIA and Lobatto IIIB [11]. However, the family of Lobatto IIIA and IIIB methods is not symplectic in the sense of (2.3).

We will now investigate how these properties translate to multisymplectic discretization methods. We start with a discussion of the Preissman box scheme.

3. The Preissman box scheme. Applying the implicit midpoint rule to the PDE (1.1) in both space and time yields

$$\mathbf{K}\delta_t^+ \mathbf{z}^{n,m+1/2} + \mathbf{L}\delta_x^+ \mathbf{z}^{n+1/2,m} = \nabla S\big(z^{n+1/2,m+1/2}\big), \tag{3.1}$$

where we define the edge midpoint approximations

$$\mathbf{z}^{n+1/2,m} = \frac{1}{2}\left(\mathbf{z}^{n+1,m} + \mathbf{z}^{n,m}\right), \qquad \mathbf{z}^{n,m+1/2} = \frac{1}{2}\left(\mathbf{z}^{n,m+1} + \mathbf{z}^{n,m}\right),$$

the cell center approximation

$$\mathbf{z}^{n+1/2,m+1/2} = \frac{1}{4}\left(\mathbf{z}^{n+1,m+1} + \mathbf{z}^{n+1,m} + \mathbf{z}^{n,m+1} + \mathbf{z}^{n,m}\right),$$

and the finite difference approximations

$$\delta_t^+ \mathbf{z}^{n,m+1/2} = \frac{1}{\Delta t}\big(\mathbf{z}^{n+1,m+1/2} - \mathbf{z}^{n,m+1/2}\big),$$
$$\delta_x^+ \mathbf{z}^{n+1/2,m} = \frac{1}{\Delta x}\big(\mathbf{z}^{n+1/2,m+1} - \mathbf{z}^{n+1/2,m}\big).$$

In addition, define

$$\tau = \frac{\Delta t}{2} \qquad \text{and} \qquad \chi = \frac{\Delta x}{2}$$

through the remainder of the text.

PROPOSITION 1. [illegible] (3.1) [illegible]

$$\delta_t^+\big(\mathbf{dz}^{n,m+1/2} \wedge \mathbf{K}\mathbf{dz}^{n,m+1/2}\big) + \delta_x^+\big(\mathbf{dz}^{n+1/2,m} \wedge \mathbf{L}\mathbf{dz}^{n+1/2,m}\big) = \mathbf{0}.$$

[illegible] Proof of this result is due to Bridges and Reich [7]. □

[1]The qualifier *analytic* will be used throughout the paper to refer to a property of the given continuum equations (e.g., analytic solutions, analytic dispersion relation, and analytic group velocity) as opposed to properties of the numerical scheme (e.g., numerical solutions, numerical dispersion relation, and numerical group velocity).

Applying the Preissman box scheme to the linear PDE (1.2) yields

$$\mathbf{K}\delta_t^+ \mathbf{z}^{n,m+1/2} + \mathbf{L}\delta_x^+ \mathbf{z}^{n+1/2,m} = \mathbf{A}\mathbf{z}^{n+1/2,m+1/2}. \tag{3.2}$$

Based upon the results of [17, 18], the modified equation for the linear PDE (1.2) can be stated explicitly as

$$\mathbf{K}\left(\mathbf{z}_t - \frac{\tau^2}{3}\mathbf{z}_{ttt} + \frac{2\tau^4}{15}\mathbf{z}_{ttttt} - \cdots\right) + \mathbf{L}\left(\mathbf{z}_x - \frac{\chi^2}{3}\mathbf{z}_{xxx} + \frac{2\chi^4}{15}\mathbf{z}_{xxxxx} - \cdots\right) = \mathbf{A}\mathbf{z}. \tag{3.3}$$

This is a higher-order linear PDE and we may substitute a standard solution ansatz of the form

$$\mathbf{z}(x,t) = \mathrm{e}^{\mathtt{i}(\Omega t + Kx)}\mathbf{a}$$

into the equation. This yields

$$\left(\mathbf{K}\mathtt{i}\left(\Omega + \frac{\tau^2\Omega^3}{3} + \frac{2\tau^4\Omega^5}{15} + \cdots\right) + \mathbf{L}\mathtt{i}\left(K + \frac{\chi^2 K^3}{3} + \frac{2\chi^4 K^5}{15} + \cdots\right) - \mathbf{A}\right)\mathbf{a} = \mathbf{0}.$$

Then, due to the identity

$$\tan(\theta) = \theta + \frac{\theta^3}{3} + \frac{2\theta^5}{15} + \cdots,$$

we obtain

$$\left(\frac{\mathtt{i}}{\tau}\tan(\tau\Omega)\mathbf{K} + \frac{\mathtt{i}}{\chi}\tan(\chi K)\mathbf{L} - \mathbf{A}\right)\mathbf{a} = \mathbf{0}. \tag{3.4}$$

The modified equation can also be stated in closed form because the expansions in (3.3) converge; it is given by

$$\frac{1}{\tau}\mathbf{K}\tanh(\tau\partial_t)\mathbf{z} + \frac{1}{\chi}\mathbf{L}\tanh(\chi\partial_x)\mathbf{z} = \mathbf{A}\mathbf{z}.$$

Furthermore, the following theorem follows immediately from (3.4).

THEOREM 1. [illegible] (3.2) [illegible]

$$\mathcal{D}(\omega, k) = 0, \tag{3.5}$$

[illegible] $\mathcal{D}$ [illegible] (1.4) [illegible]

$$\omega = \frac{\tan(\tau\Omega)}{\tau} \qquad \textit{and} \qquad k = \frac{\tan(\chi K)}{\chi}, \tag{3.6}$$

[illegible] $-\frac{\pi}{2} < \tau\Omega < \frac{\pi}{2}$ [illegible] $-\frac{\pi}{2} < \chi K < \frac{\pi}{2}$ [illegible]

$$\mathcal{D}_N(\Omega, K) := \mathcal{D}\left(\frac{\tan(\tau\Omega)}{\tau}, \frac{\tan(\chi K)}{\chi}\right) = 0.$$

This result was first published by Ascher and McLachlan [1]; see also [17]. Three important conclusions are to be drawn. (i) The numerical group velocity V_g is defined by

$$V_g = \frac{d}{dK}\Omega(K) = -\frac{\partial \mathcal{D}}{\partial k}\frac{\partial k}{\partial K}\Big/\frac{\partial \mathcal{D}}{\partial \omega}\frac{\partial \omega}{\partial \Omega} = v_g\frac{\partial k}{\partial K}\Big/\frac{\partial \omega}{\partial \Omega}.$$

Since both

$$\frac{\partial \omega}{\partial \Omega} > 0 \qquad \text{and} \qquad \frac{\partial k}{\partial K} > 0,$$

we have derived the following.

COROLLARY 1. [illegible] v_g [illegible]

(ii) The monotonicity and invertibility of (3.6) implies the following corollary.

COROLLARY 2. [illegible] ω [illegible] k [illegible]

(iii) We consider the Lagrangian density (1.6) with ω and k now functions of Ω and K, respectively, and define the numerical energy density and flux for a single mode solution by

$$E_N = \frac{\mathtt{i}\Omega}{2}\frac{\partial \omega}{\partial \Omega}\bar{\mathbf{a}}^T\mathbf{K}\mathbf{a}, \qquad F_N = -\frac{\mathtt{i}\Omega}{2}\frac{\partial k}{\partial K}\bar{\mathbf{a}}^T\mathbf{L}\mathbf{a}. \tag{3.7}$$

COROLLARY 3. [illegible]

$$F_N = V_g E_N$$

[illegible] V_g [illegible]

[illegible]. The statement follows from the definitions (3.7) and the total derivative of

$$\bar{\mathbf{a}}^T\left(\frac{\mathtt{i}}{\tau}\tan(\tau\Omega)\mathbf{K} + \frac{\mathtt{i}}{\chi}\tan(\chi K)\mathbf{L} - \mathbf{A}\right)\mathbf{a} = 0$$

with respect to K. □

4. Numerical dispersion relation for multisymplectic GL RK methods. We now discuss the discretization of a linear multisymplectic PDE (1.2) using GL RK methods (possibly of different orders) in space and time. We apply an s^0-stage GL RK method in time with Butcher arrays $A^0 = (a^0_{ij})$, $b^0 = (b^0_i)$, $i, j = 1, \ldots, s^0$, and a similarly denoted s^1-stage GL RK method in space.

Reserve the indices m and n for the spatial and temporal meshes, respectively, and the indices i and j for spatial and temporal stage indices, respectively. The index k is a dummy index. Summation is implied by repeated indices in a product.

Let $\mathbf{Z} \approx \mathbf{z}$ at interior stages and $\boldsymbol{\zeta}, \boldsymbol{\sigma} \approx \mathbf{z}$ on horizontal and vertical edges, respectively. Let $\mathbf{T} \approx \mathbf{z}_t$ at interior stages and $\boldsymbol{\tau} \approx \mathbf{z}_t$ on edges. Let $\mathbf{X} \approx \mathbf{z}_x$ at interior stages and $\boldsymbol{\xi} \approx \mathbf{z}_x$ on edges. Let $\mathbf{U} \approx \mathbf{z}_{xt}$ at interior stages. (See Figure 1.)

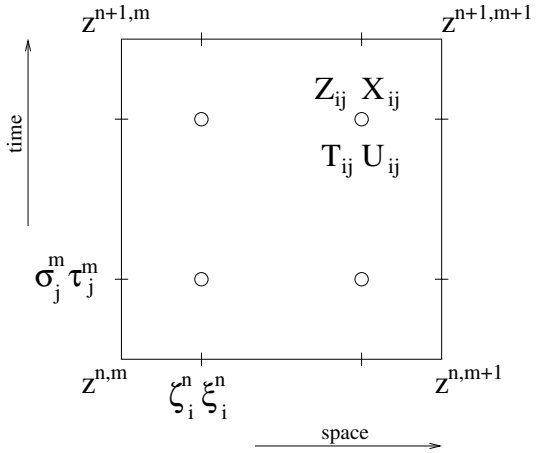


FIG. 1. *Arrangement of approximate quantities in* (4.1)–(4.11).

$$\mathbf{K}\mathbf{T}_{ij} + \mathbf{L}\mathbf{X}_{ij} = \mathbf{A}\mathbf{Z}_{ij}, \tag{4.1}$$

$$\mathbf{X}_{ij} = \boldsymbol{\xi}_i^n + \Delta t\, a_{jk}^0 \mathbf{U}_{ik}, \tag{4.2}$$

$$\mathbf{T}_{ij} = \boldsymbol{\tau}_j^m + \Delta x\, a_{ik}^1 \mathbf{U}_{kj}, \tag{4.3}$$

$$\mathbf{Z}_{ij} = \boldsymbol{\zeta}_i^n + \Delta t\, a_{jk}^0 \mathbf{T}_{ik}, \tag{4.4}$$

$$= \boldsymbol{\sigma}_j^m + \Delta x\, a_{ik}^1 \mathbf{X}_{kj}, \tag{4.5}$$

$$\boldsymbol{\xi}_i^{n+1} = \boldsymbol{\xi}_i^n + \Delta t\, b_j^0 \mathbf{U}_{ij}, \tag{4.6}$$

$$\boldsymbol{\zeta}_i^{n+1} = \boldsymbol{\zeta}_i^n + \Delta t\, b_j^0 \mathbf{T}_{ij}, \tag{4.7}$$

$$\boldsymbol{\tau}_j^{m+1} = \boldsymbol{\tau}_j^m + \Delta x\, b_i^1 \mathbf{U}_{ij}, \tag{4.8}$$

$$\boldsymbol{\sigma}_j^{m+1} = \boldsymbol{\sigma}_j^m + \Delta x\, b_i^1 \mathbf{X}_{ij}, \tag{4.9}$$

$$\mathbf{z}^{n+1,m+1} = \mathbf{z}^{n,m+1} + \Delta t\, b_j^0 \boldsymbol{\tau}_j^{m+1} \tag{4.10}$$

$$= \mathbf{z}^{n+1,m} + \Delta x\, b_i^1 \boldsymbol{\xi}_i^{n+1}. \tag{4.11}$$

Equations (4.1), (4.4), (4.5), (4.7), and (4.9) form a closed set (same number of equations as unknowns). To include the grid point data, either {(4.2), (4.6), and (4.11)} or {(4.3), (4.8), and (4.10)} should be added. The remaining relations are redundant.

We will concentrate on (4.1), (4.4), (4.5), (4.7), and (4.9) in the following. Let us assume that we have ω, k, and $\mathbf{a}$ satisfying the plane wave solution

$$[\mathtt{i}\omega\mathbf{K} + \mathtt{i}k\mathbf{L} - \mathbf{A}]\mathbf{a} = \mathbf{0}.$$

We look for a (complex-valued) numerical solution of the form

$$\mathbf{z}^{n,m} = \mathbf{a}^{nm} := \mathrm{e}^{\mathtt{i}\Omega n\Delta t}\mathrm{e}^{\mathtt{i}Km\Delta x}\mathbf{a}.$$

Let us therefore make the ansatz that all stage vectors are proportional to $\mathbf{a}^{nm}$:

$$\mathbf{Z}_{ij}^{nm} = \gamma_{ij}\mathbf{a}^{nm}, \quad \mathbf{X}_{ij}^{nm} = x_{ij}\mathbf{a}^{nm}, \quad \mathbf{T}_{ij}^{nm} = t_{ij}\mathbf{a}^{nm}, \quad \boldsymbol{\zeta}_i^n = \alpha_i\mathbf{a}^{nm}, \quad \boldsymbol{\sigma}_j^m = \beta_j\mathbf{a}^{nm}.$$

Substituting this solution into (4.1) gives

$$[t_{ij}\mathbf{K} + x_{ij}\mathbf{L} - \gamma_{ij}\mathbf{A}]\mathbf{a}^{nm} = 0,$$

for which the dispersion relation yields

$$\frac{t_{ij}}{\gamma_{ij}} = \mathtt{i}\omega, \qquad \frac{x_{ij}}{\gamma_{ij}} = \mathtt{i}k,$$

that is, $\mathbf{T}_{ij} = \mathtt{i}\omega\mathbf{Z}_{ij}$ and $\mathbf{X}_{ij} = \mathtt{i}k\mathbf{Z}_{ij}$.

Next, (4.4) yields

$$\gamma_{ij}\mathbf{a}^{nm} = \alpha_i\mathbf{a}^{nm} + \mathtt{i}\omega\Delta t\, a^0_{jk}\gamma_{ik}\mathbf{a}^{nm}.$$

For each i we obtain the solution

$$(I - \mathtt{i}\omega\Delta t\, A^0)\boldsymbol{\gamma}_{i:} = \alpha_i \mathbb{1}, \qquad \boldsymbol{\gamma}_{i:} = (\gamma_{i1}, \ldots, \gamma_{is^1})^T,$$

from which it follows that $\gamma_{ij} = \lambda_i\mu_j$ with

$$\boldsymbol{\mu} = (I - \mathtt{i}\omega\Delta t\, A^0)^{-1}\mathbb{1}, \qquad \boldsymbol{\alpha} = \boldsymbol{\lambda}.$$

Similarly, (4.5) yields

$$\boldsymbol{\lambda} = (I - \mathtt{i}k\Delta x\, A^1)^{-1}\mathbb{1}, \qquad \boldsymbol{\beta} = \boldsymbol{\mu}.$$

Finally, substitution into (4.7) gives

$$\lambda_i(\mathrm{e}^{\mathtt{i}\Omega\Delta t} - 1)\mathbf{a}^{nm} = \mathtt{i}\omega\Delta t b^0_j\lambda_i\mu_j\mathbf{a}^{nm},$$

or

$$(\mathrm{e}^{\mathtt{i}\Omega\Delta t} - 1) = \mathtt{i}\omega\Delta t\,(b^0)^T\boldsymbol{\mu} \Rightarrow e^{\mathtt{i}\Omega\Delta t} = 1 + \mathtt{i}\omega\Delta t\,(b^0)^T(I - \mathtt{i}\omega\Delta t\, A^0)^{-1}\mathbb{1} = R^0(\mathtt{i}\omega\Delta t). \tag{4.12}$$

Also, (4.9) similarly yields

$$\mathrm{e}^{\mathtt{i}K\Delta x} = R^1(\mathtt{i}k\Delta x). \tag{4.13}$$

We have effectively proven the following theorem.

THEOREM 2. [illegible] $R^0(z)$ [illegible] $R^1(z)$ [illegible] (1.2) [illegible] (1.3) [illegible] (1.2) [illegible]

$$\mathbf{z}^{n,m} = \mathrm{e}^{\mathtt{i}\Omega n\Delta t}\mathrm{e}^{\mathtt{i}Km\Delta x}\mathbf{a} = R^0(\mathtt{i}\omega\Delta t)^n\, R^1(\mathtt{i}k\Delta x)^m\mathbf{a},$$

[illegible] ω [illegible] k [illegible] $\mathcal{D}(\omega, k) = 0$ [illegible]

$$\mathcal{D}_N(\Omega, K) = 0$$

[illegible] $\omega(\Omega)$ [illegible] $k(K)$ [illegible] (4.12) [illegible] (4.13) [illegible]

$$\overline{\mathcal{D}_N(\Omega, K)} = \mathcal{D}_N(\overline{\Omega}, \overline{K}).$$

Since any symmetric and A-stable implicit RK method behaves like a symplectic method on the linear equation level, Theorem 2 also applies to those RK methods as well and to Lobatto IIIA and IIIB methods in particular.

We may write $R^0(\mathtt{i}y) = \mathrm{e}^{\mathtt{i}\phi(y)}$ and $R^1(\mathtt{i}y) = \mathrm{e}^{\mathtt{i}\theta(y)}$, where ϕ and θ are monotone phase functions as defined in section 2. Hence the practical computation of the numerical dispersion relation is as follows. Given a computational mode with wave number $K \in (-\pi/\Delta x, +\pi/\Delta x]$, one first finds the uniquely defined wave number k from $\mathrm{e}^{\mathtt{i}K\Delta x} = R^1(\mathtt{i}k\Delta x)$. The analytic dispersion relation $\mathcal{D}(\omega, k) = 0$ implies a corresponding frequency ω. One then uses $R^0(\mathtt{i}\omega\Delta t) = \mathrm{e}^{\mathtt{i}\phi}$ to determine the uniquely defined phase $\phi(\omega\Delta t)$. Finally, the modulo operation (2.7) gives the computational frequency Ω.

A number of important conclusions may be drawn. (i) Monotonicity of ϕ and θ implies $\phi' \geq 0$, $\theta' \geq 0$, and we obtain the following.

COROLLARY 4. [illegible]

$$V_g = -\frac{\partial \mathcal{D}_N}{\partial K} \Big/ \frac{\partial \mathcal{D}_N}{\partial \Omega}$$

[illegible] (K, Ω) [illegible]

$$v_g = -\frac{\partial \mathcal{D}}{\partial k} \Big/ \frac{\partial \mathcal{D}}{\partial \omega}$$

[illegible] (k, ω)

(ii) Monotonicity of ϕ and θ implies that we can define the inverses of ϕ and θ and the two explicit relations

$$\omega = \frac{1}{\Delta t}\phi^{-1}(\Delta t\Omega), \qquad k = \frac{1}{\Delta x}\theta^{-1}(\Delta x K).$$

Using these two relations, we find that a numerical single mode solution satisfies

$$\bar{\mathbf{a}}^T \left(\mathtt{i}\omega\mathbf{K} + \mathtt{i}k\mathbf{L} - \mathbf{A}\right)\mathbf{a} = 0.$$

Definition (3.7) for the energy density and flux implies now the following corollary.

COROLLARY 5. [illegible] (3.7) [illegible]

$$F_N = V_g E_N$$

[illegible] V_g [illegible]

(iii) Because of (2.7), nonintersecting branches of the analytic dispersion relation may intersect in the (K, Ω) plane unless either the implicit midpoint method is applied in time or Δt is chosen such that

$$\sup_{k \in S} |\phi(\omega(k)\Delta t)| \leq \pi,$$

where k is to be restricted to the proper domain S determined by the possible numerical wave numbers $-\pi < K\Delta x \leq \pi$. Intersecting branches (resonances) of the numerical dispersion relation may lead to instabilities of the method under small perturbations (see [2] for a discussion of the analytic case). Exclusion of intersecting branches leads to a CFL-type step-size restriction (except for an implicit midpoint time discretization). This will be discussed further in the following section.

5. An example: Fourth-order GL RK method in space and time. The linear wave equation

$$u_{tt} = u_{xx}$$

can be reformulated as a multisymplectic PDE of the form

$$-v_t - w_x = 0, \qquad u_t = v, \qquad u_x = -w \tag{5.1}$$

with $\mathbf{z} = (u, v, w)^T$,

$$\mathbf{K} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{L} = \begin{bmatrix} 0 & 0 & -1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}, \quad \mathbf{A} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix}.$$

The analytic dispersion relation is $\omega^2 = k^2$. The application of the fourth-order GL RK method in space yields

$$\tan \frac{K\Delta x}{2} = \frac{k\Delta x}{2 - \frac{k^2 \Delta x^2}{6}},$$

and we find that $k\Delta x$ must be in the range of $0 \le k\Delta x < \sqrt{12}$ for computational wave numbers K satisfying $0 \le K\Delta x < \pi$. More specifically, we obtain

$$k\Delta x = -\frac{3}{b} + \sqrt{\frac{9}{b^2} + 12}, \qquad b = \tan \frac{K\Delta x}{2}.$$

Let us apply the same fourth-order method as the temporal discretization. Upon introducing the step-size mesh ratio $\beta = \Delta t/\Delta x$, one finds that $\omega\Delta t$ satisfies the inequalities

$$|\omega\Delta t| < \beta\sqrt{12}.$$

Branches of the numerical dispersion relation intersect whenever $\beta > 1$. However, $\phi' \ge 0$ (i.e., preservation of the sign of the group velocity) is guaranteed for any value of β. See Figure 2 for both cases $\beta < 1$ and $\beta > 1$.

Intersecting branches can be avoided altogether by an implicit midpoint discretization in time. Higher-order temporal discretization based upon the implicit midpoint method can be constructed using, e.g., Yoshida's fourth-order composition method [21]. The composed method still gives rise to a stability function of the form

$$R(i\omega\Delta t) = \mathrm{e}^{\mathrm{i}\phi}, \qquad \phi(\omega\Delta t) = 2\phi_{\mathrm{IM}}(\gamma\omega\Delta t) + \phi_{\mathrm{IM}}((1-2\gamma)\omega\Delta t),$$

where $\gamma = 1/(2-2^{1/3})$ and ϕ_{IM} is the phase function for the implicit midpoint method. It can be verified that $|\phi(\omega\Delta t)| \le \pi$ and that ϕ is a monotone function. See Figure 2 for numerical results.

6. The explicit midpoint scheme. This section discusses the effect of symplectic noncompact differencing methods by means of the explicit midpoint scheme.

Applying the explicit midpoint rule in both space and time yields a simple second-order two-step method. Introducing

$$\delta_t^{1/2}\mathbf{z}^{n,m} = \frac{1}{2\Delta t}\left(\mathbf{z}^{n+1,m} - \mathbf{z}^{n-1,m}\right) \quad \text{and} \quad \delta_x^{1/2}\mathbf{z}^{n,m} = \frac{1}{2\Delta x}\left(\mathbf{z}^{n,m+1} - \mathbf{z}^{n,m-1}\right),$$

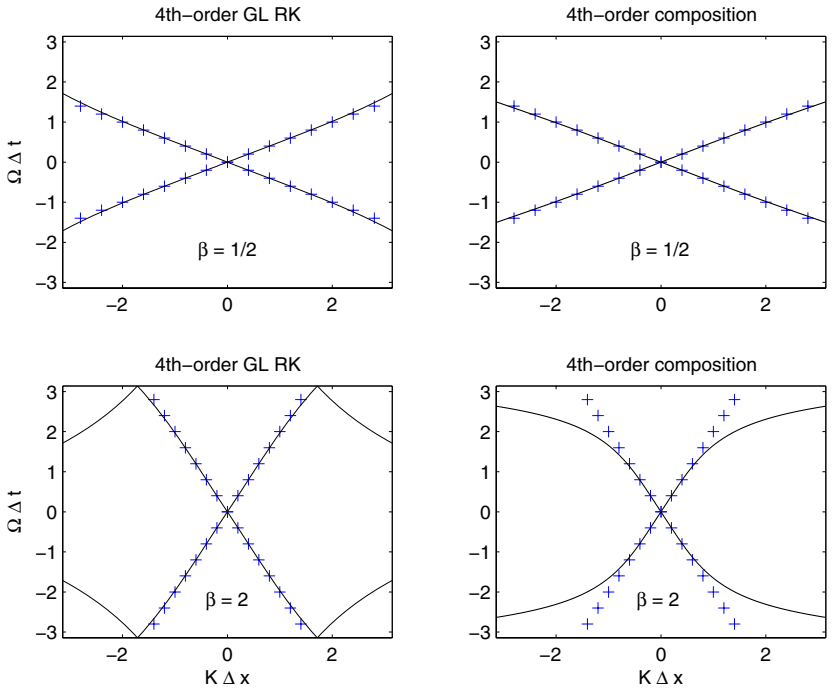


FIG. 2. *The numerical dispersion relations displayed in the left two plots are for a fourth-order GL RK space and time discretization for different values of* $\beta = \Delta t/\Delta x$. *The intersecting branches can clearly be seen for* $\beta = 2$. *These intersecting branches are no longer presented in a fourth-order space and time discretization with the time discretization now being a fourth-order composition method based on the implicit midpoint method. These results are displayed in the two plots to the right.*

in order to simplify notation, we present the explicit midpoint scheme in the following proposition.

PROPOSITION 2.

$$\mathbf{K}\delta_t^{1/2}\mathbf{z}^{n,m} + \mathbf{L}\delta_x^{1/2}\mathbf{z}^{n,m} = \nabla_{\mathbf{z}} S\left(\mathbf{z}^{n,m}\right)$$

$$\delta_t^+\left(\mathbf{dz}^{n,m} \wedge \mathbf{K}\mathbf{dz}^{n-1,m}\right) + \delta_x^+\left(\mathbf{dz}^{n,m} \wedge \mathbf{L}\mathbf{dz}^{n,m-1}\right) = 0.$$

The associated variational equation is given by

$$\mathbf{K}\delta_t^{1/2}\mathbf{dz}^{n,m} + \mathbf{L}\delta_x^{1/2}\mathbf{dz}^{n,m} = S_{zz}(\mathbf{z}^{n,m})\mathbf{dz}^{n,m}.$$

After taking the wedge product with $\mathbf{dz}^{n,m}$, we get

$$\mathbf{dz}^{n,m} \wedge \mathbf{K}\delta_t^{1/2}\mathbf{dz}^{n,m} + \mathbf{dz}^{n,m} \wedge \mathbf{L}\delta_x^{1/2}\mathbf{dz}^{n,m} = 0,$$

which is equivalent to

$$0 = \frac{1}{\Delta t}\left(\mathbf{dz}^{n,m} \wedge \mathbf{K}\left(\mathbf{dz}^{n+1,m} - \mathbf{dz}^{n-1,m}\right)\right) + \frac{1}{\Delta x}\left(\mathbf{dz}^{n,m} \wedge \mathbf{L}\left(\mathbf{dz}^{n,m+1} - \mathbf{dz}^{n,m-1}\right)\right).$$

Since

$$\begin{aligned}\mathbf{dz}^{n,m} \wedge \mathbf{K}\left(\mathbf{dz}^{n+1,m} - \mathbf{dz}^{n-1,m}\right) &= \mathbf{dz}^{n,m} \wedge \mathbf{K}\mathbf{dz}^{n+1,m} - \mathbf{dz}^{n,m} \wedge \mathbf{K}\mathbf{dz}^{n-1,m} \\ &= \mathbf{dz}^{n+1,m} \wedge \mathbf{K}\mathbf{dz}^{n,m} - \mathbf{dz}^{n,m} \wedge \mathbf{K}\mathbf{dz}^{n-1,m}\end{aligned}$$

and a similar result for the spatial discretization part, this gives the desired result. □

Applying this discretization to the linear equation yields

$$\mathbf{K}\delta_t^{1/2}\mathbf{z}^{n,m} + \mathbf{L}\delta_x^{1/2}\mathbf{z}^{n,m} = \mathbf{A}\mathbf{z}^{n,m}. \tag{6.1}$$

LEMMA 1. [illegible] (6.1) [illegible]

$$\mathcal{D}_{\mathbf{A}}(\omega, k) = 0$$

[illegible] $\mathcal{D}_{\mathbf{A}}$ [illegible] (1.4) [illegible]

$$\omega = \frac{\sin(\Omega\Delta t)}{\Delta t} \qquad \textit{and} \qquad k = \frac{\sin(K\Delta x)}{\Delta x}$$

[illegible] $-\pi < \Delta t\Omega \leq \pi$ [illegible] $-\pi < \Delta x K \leq \pi$
[illegible]. Substituting the (complex-valued) numerical solution ansatz

$$\mathbf{z}^{n,m} = \mathrm{e}^{\mathtt{i}(\Omega n\Delta t + Km\Delta x)}\mathbf{a}$$

into (6.1) gives

$$\left(\mathbf{K}\frac{\mathrm{e}^{\mathtt{i}\Omega\Delta t} - \mathrm{e}^{-\mathtt{i}\Omega\Delta t}}{2\Delta t} + \mathbf{L}\frac{\mathrm{e}^{\mathtt{i}K\Delta x} - \mathrm{e}^{-\mathtt{i}K\Delta x}}{2\Delta x} - \mathbf{A}\right)\mathbf{a} = 0.$$

Now, using the trigonometric identities

$$\mathrm{e}^{\mathtt{i}\Omega\Delta t} - \mathrm{e}^{-\mathtt{i}\Omega\Delta t} = 2\mathtt{i}\sin(\Delta t\Omega) \qquad \text{and} \qquad \mathrm{e}^{\mathtt{i}K\Delta x} - \mathrm{e}^{-\mathtt{i}K\Delta x} = 2\mathtt{i}\sin(\Delta x K),$$

the proof is completed. □

A first important conclusion is that the relation $R(\mathtt{i}\omega\Delta t) = \mathrm{e}^{\mathtt{i}\phi}$ implies that $\phi(\omega\Delta t)$, defined by

$$\sin\phi = \omega\Delta t,$$

is no longer monotone and is not defined for $|\omega\Delta t| > 1$ (instability of the method). Furthermore, the nonmonotonicity implies the existence of spurious roots in the numerical dispersion relation. A similar observation applies to the midpoint spatial discretization. Nonmonotonicity also implies that the sign of the numerical group velocity can be different from the sign of the analytic value for the corresponding (ω, k) pair.

We now investigate the behavior of the explicit midpoint space-time discretization for the linear wave equation (5.1). The discretization can be derived from the discrete Lagrangian

$$L = \frac{1}{2}\sum_{n,m}\left[\left(\frac{u^{n+1,m} - u^{n-1,m}}{2\Delta t}\right)^2 + \left(\frac{u^{n,m+1} - u^{n,m-1}}{2\Delta x}\right)^2\right]$$

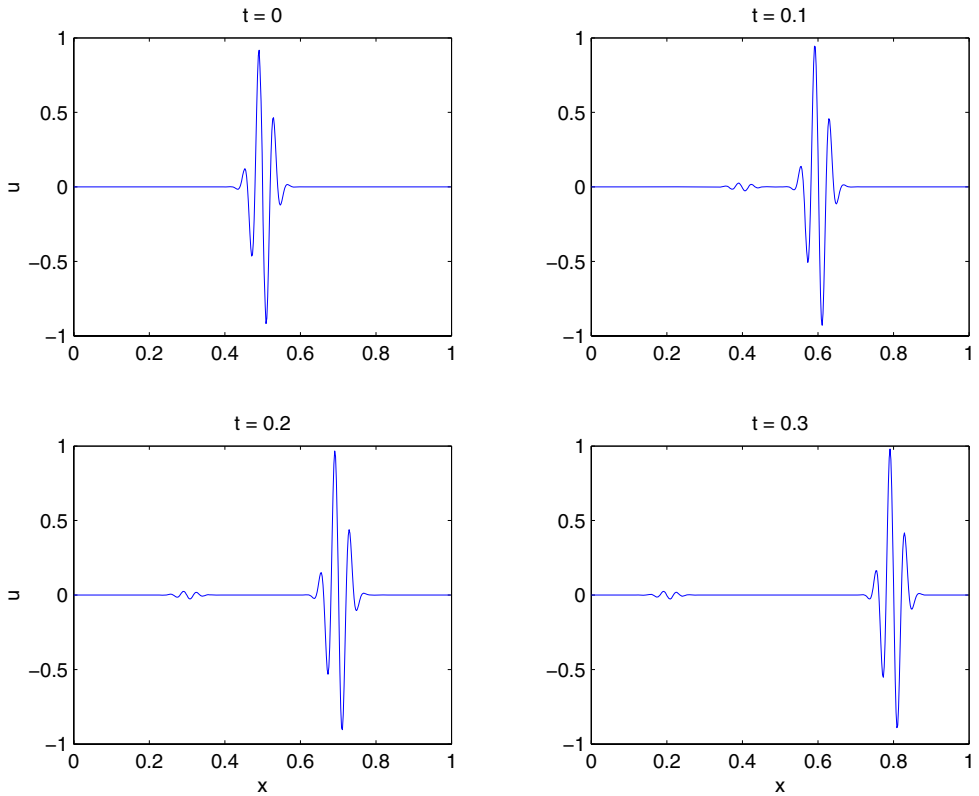


FIG. 3. *Numerical propagation of a wave packet under the space-time EMP method. The analytic solution consists of a wave packet moving entirely to the right. The displayed small perturbation moving towards the left is a numerical artifact due to the nonmonotone numerical dispersion relation of the EMP method.*

following the principle of discrete variational mechanics [15]. The numerical dispersion relation is given by

$$\sin^2(\Omega\Delta t) = \frac{\Delta t^2}{\Delta x^2}\sin^2(K\Delta x)$$

and is real-valued provided $\Delta t/\Delta x \leq 1$ (CFL condition). However, even for $\Delta t/\Delta x \leq 1$, the numerical dispersion relation has spurious roots for given K and Ω, respectively.

Let us first demonstrate the implication of the nonmonotone numerical dispersion relation for solutions of the linear wave equation (5.1). We solve the wave equation with the explicit midpoint (EMP) method in space and time over a spatial domain $x \in [0, 1)$ with periodic boundary conditions and initial data

$$u(x,0) = f(x), \quad v(x,0) = -f'(x), \quad f(x) = \exp(-30^2(x-1/2)^2)\sin(50\pi x).$$

The analytic solution consists of a wave packet moving to the right. The numerical solution for $\Delta x = 1/450$ and $\Delta t = 0.9\Delta x$ is displayed in Figure 3. The small perturbation moving towards the left is due to the nonmonotone numerical dispersion relation and would be absent for a GL RK discretization in space and time.

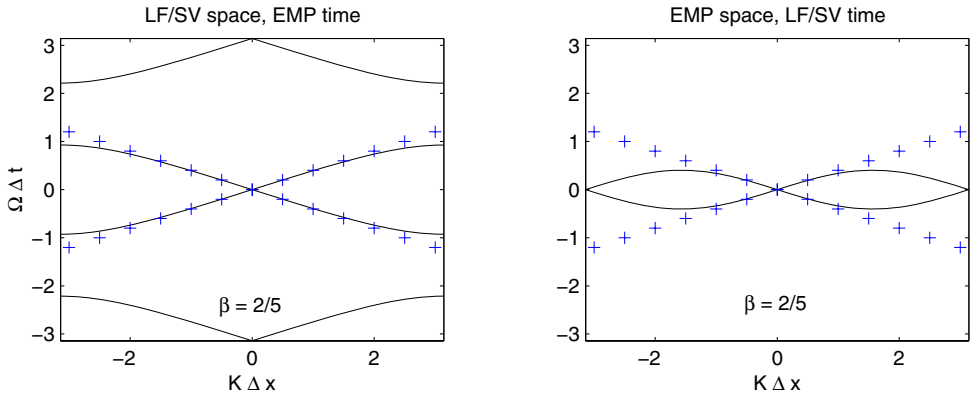


FIG. 4. *The numerical dispersion relations displayed on the left are for an EMP discretization in time and a Störmer–Verlet (LF/SV) discretization in space. The spurious branches (roots) in the numerical dispersion relation are due to the multistep EMP method. On the contrary, an LF/SV discretization in time combined with an EMP discretization in space yields a nonmonotone dispersion relation implying spurious roots.*

The spurious roots for K and/or Ω in the numerical dispersion relation of the space-time EMP discretization can be partially eliminated by substituting a symplectic one-step method in space or in time. As an example we consider the symplectic leapfrog/Störmer–Verlet (LF/SV) method. Applying the LF/SV method in space yields the numerical dispersion relation

$$\sin^2(\Omega\Delta t) = \frac{4\Delta t^2}{\Delta x^2}\sin^2\left(\frac{K\Delta x}{2}\right),$$

while application in time implies

$$\sin^2\left(\frac{\Omega\Delta t}{2}\right) = \frac{\Delta t^2}{4\Delta x^2}\sin^2(K\Delta x).$$

As shown in Figure 4 both combinations still imply spurious roots in their numerical dispersion relations. The same statement is true if the SV method is replaced in time by the implicit midpoint (IMP) method. The only difference is that there is no longer a stability restriction on β. See Figure 5.

It should be noted that there are spatial multistep discretizations with monotone phase function [9]. However, those methods do not seem to satisfy a discrete variational principle and, hence, do not give rise to multisymplectic discretization methods.

7. Summary: Towards a characterization of multisymplectic integration methods. There are a variety of numerical discretizations that satisfy a discrete multisymplectic conservation law. When applied to linear multisymplectic PDEs with wave-like solutions, all these methods yield real-valued numerical dispersion relations

$$\mathcal{D}_N(\Omega, K) = 0$$

provided the ratio $\Delta t/\Delta x$ satisfies a CFL-type condition $\Delta t/\Delta x \leq C$, where C is a positive number. This follows from the fact that $R(z)$, i.e., $\mathbf{z}^{n+1} = R(z)\mathbf{z}^n$, of a

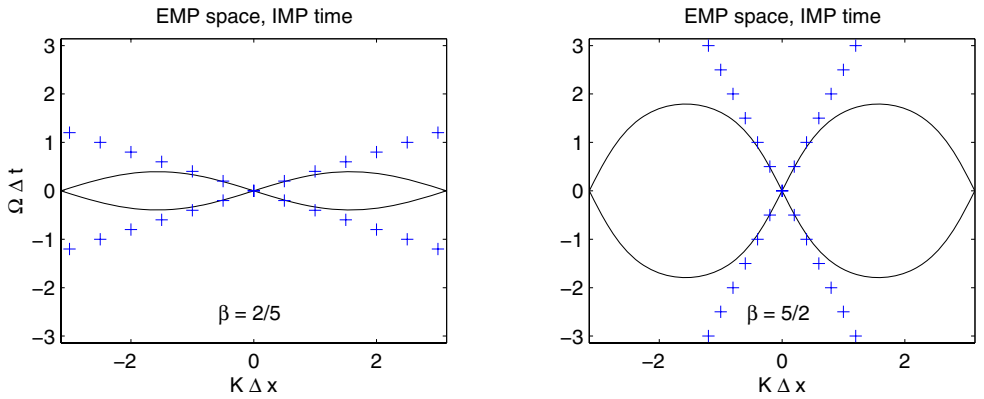


FIG. 5. *The displayed numerical dispersion relations are for an EMP discretization in space and an IMP discretization in time. Results are shown for $\beta < 1$ and $\beta > 1$.*

(stable and consistent) symplectic (one-step or multistep) method has to satisfy

$$|R(\mathtt{i}\sigma)| = 1$$

for sufficiently small $\sigma \geq 0$ because of conservation of volume.

One-step multisymplectic discretizations in space and time, such as GL RK, lead to (complex-valued) numerical approximations of the form

$$\mathbf{z}^{n,m} = R^0(\mathtt{i}\omega\Delta t)^n \, R^1(\mathtt{i}k\Delta x)^m \, \mathbf{a},$$

where $R^0(z)$ and $R^1(z)$ are the stability functions for the temporal and spatial discretization, respectively. It follows that multisymplectic GL RK discretizations in space and time are unconditionally stable, i.e., $C = \infty$.

Another key aspect of numerical methods is the existence or nonexistence of spurious roots in the numerical dispersion relation. The existence of spurious modes can be excluded provided (i)

$$R^0(\mathtt{i}y) = \mathrm{e}^{\mathtt{i}\phi(y)}, \qquad R^1(\mathtt{i}y) = \mathrm{e}^{\mathtt{i}\theta(y)}$$

for monotone phase functions ϕ, θ; and (ii) $\Delta t/\Delta x$ satisfies a CFL-type condition $\Delta t/\Delta x \leq c$. While all GL RK methods have monotone phase functions [10], the IMP time discretization is the only GL RK method for which $c = \infty$. Monotonicity of the phase function $\phi(\sigma)$ alone implies conservation of the sign of the group velocity under numerical discretization. Hence wave packets propagate numerically at a perturbed speed but in the correct direction. A multisymplectic discretization with

$$c < \frac{\Delta t}{\Delta x} \leq C$$

will, in general, lead to numerically induced 1-1 resonances (intersections of solution branches of the dispersion relation). Those spurious resonances can lead to instabilities of the method under small perturbations in the same manner as analytic resonances can give rise to instabilities [2]. It was found that higher-order time discretizations with $c = \infty$ can be constructed by proper composition of the IMP method. It should be noted that multisymplectic methods can be subject to additional numerically induced resonances. (Compare, e.g., Mandziuk and Schlick [14] for the case

of Hamiltonian ODEs.) This issue will be investigated in detail in a forthcoming publication.

Acknowledgment. The authors thank an anonymous referee for suggestions that led to a more thorough treatment of the energy transport under multisymplectic discretization methods.

REFERENCES

[1] U. M. Ascher and R. I. McLachlan, *Multisymplectic box schemes and the Korteweg-de Vries equation*, Appl. Numer. Math., 48 (2004), pp. 255–269.
[2] T. J. Bridges, *A geometric formulation of the conservation of wave action and its implications for signature and the classification of instabilities*, Proc. Roy. Soc. London Ser. A, 453 (1997), pp. 1365–1395.
[3] T. J. Bridges, *Multi-symplectic structures and wave propagation*, Math. Proc. Cambridge Philos. Soc., 121 (1997), pp. 147–190.
[4] T. J. Bridges and G. Derks, *Unstable eigenvalues and the linearization about solitary waves and fronts with symmetry*, R. Soc. Lond. Proc. Ser. A Math. Phys. Eng. Sci., 455 (1999), pp. 2427–2469.
[5] T. J. Bridges and G. Derks, *Linear instability of solitary wave solutions of the Kawahara equation and its generalizations*, SIAM J. Math. Anal., 33 (2002), pp. 1356–1378.
[6] T. J. Bridges, G. Derks, and G. Gottwald, *Stability and instability of solitary waves of the fifth-order KdV equation: A numerical framework*, Phys. D, 172 (2002), pp. 190–216.
[7] T. J. Bridges and S. Reich, *Multi-symplectic integrators: Numerical schemes for Hamiltonian PDEs that conserve symplecticity*, Phys. Lett. A, 284 (2001), pp. 184–193.
[8] T. J. Bridges and S. Reich, *Multi-symplectic spectral discretizations for the Zakharov-Kuznetsov and shallow-water equations*, Phys. D, 152 (2001), pp. 491–504.
[9] J. Frank and S. Reich, *Energy-Conserving Semi-discretizations and Spurious Numerical Reflections*, CWI Report MAS-E0608, CWI, Amsterdam, The Netherlands, 2006.
[10] N. Guglielmi and E. Hairer, *Order starts and stability for delay differential equations*, Numer. Math., 83 (1999), pp. 371–383.
[11] E. Hairer, Ch. Lubich, and G. Wanner, *Geometric Numerical Integration: Structure Preserving Algorithms for Ordinary Differential Equations*, Springer-Verlag, Berlin, 2002.
[12] S. Kournabaeva and S. Shkoller, *A variational approach to second-order multisymplectic field theories*, J. Geom. Phys., 35 (2000), pp. 333–366.
[13] B. Leimkuhler and S. Reich, *Simulating Hamiltonian Dynamics*, Cambridge University Press, Cambridge, UK, 2005.
[14] M. Mandziuk and T. Schlick, *Resonance in the dynamics of chemical systems simulated by the implicit-midpoint scheme*, Chem. Phys. Lett., 237 (1995), pp. 525–535.
[15] J. E. Marsden, G. P. Patrick, and S. Shkoller, *Multi-symplectic geometry, variational integrators, and nonlinear PDEs*, Commun. Math. Phys., 199 (1999), pp. 351–395.
[16] R. I. McLachlan, *Symplectic integration of Hamiltonian wave equations*, Numer. Math., 66 (1994), pp. 465–492.
[17] B. E. Moore, *A Modified Equations Approach for Multi-symplectic Integration Methods*, Ph.D. thesis, University of Surrey, Guildford, UK, 2003.
[18] B. E. Moore and S. Reich, *Multi-symplectic integrators for Hamiltonian PDEs*, Future Generation Comp. Syst., 19 (2003), pp. 395–402.
[19] S. Reich, *Multi-symplectic Runge-Kutta collocation methods for Hamiltonian wave equations*, J. Comput. Phys., 157 (2000), pp. 473–499.
[20] G. B. Whitham, *Linear and Nonlinear Waves*, Wiley-Interscience, New York, 1974.
[21] H. Yoshida, *Construction of higher order symplectic integrators*, Phys. Lett. A, 150 (1990), pp. 262–268.

SIAM J. SCI. COMPUT.
Vol. 28, No. 1, pp. 278–302
© 2006 Society for Industrial and Applied Mathematics

A REAL GHOST FLUID METHOD FOR THE SIMULATION OF MULTIMEDIUM COMPRESSIBLE FLOW*

C. W. WANG†, T. G. LIU‡, AND B. C. KHOO§

Abstract. In the previous ghost fluid methods (GFMs) developed, the focus is on the definition of ghost fluid states while the pressure and velocity in the real fluid sides are taken for granted, except for the correction made to the density at the real fluid nodes next to the interface to overcome the possible problems related to overheating. It has been found that such GFMs encounter many difficulties when applied to shock impedance matching (-like) problems due to the inability of accurately imposing interfacial conditions. By predicting the flow states for the real fluid nodes just next to the interface and the ghost fluid nodes using the Riemann problem solver, a more accurate interface boundary condition can be imposed and the said difficulties are mitigated to a large extent. This leads to the development of a proposed real-GFM in this work. A simple yet efficient extension of the present method to multidimensions is also introduced. In order to overcome issues associated with the severe bunching of level set contours due to the large flow velocity gradient, an extension (artificial) velocity field is constructed in the computation of the level set function. The present method is applied to various one- and two-dimensional problems involving strong shock-interface interaction and complex flow physics.

Key words. Riemann problem, level set method, multimedium flow

AMS subject classifications. 35L45, 65C20, 76T10

DOI. 10.1137/030601363

1. Introduction.

Essentially nonoscillatory (ENO) schemes, first introduced by Harten and Osher [1] and Harten et al. [2], can achieve uniformly higher-order accuracy with sharp, essentially nonoscillatory shock transition. ENO schemes have shown their robustness and have worked very successfully in solving hyperbolic conservation laws [3, 4, 5] for single medium compressible flows. However, when such a scheme is used for simulating multimedium compressible flows, nonphysical oscillations occur or it may even fail to work. Additional work and techniques usually become necessary to somehow accommodate the change of the equation of state across the interface. One may refer to the conservative two-fluid capturing method developed for barotropic flows [6, 7], the nonconservative ghost fluid method (GFM) [8, 9] and its variants [10, 11], or others [12, 13, 14, 15, 16], to name some.

Of these methods, the GFM [8, 9] presented a fairly simple and flexible way to treat multimedium flows and is easily extended to multidimensions. The original GFM developed in [8] defines for the first medium the pressure and velocity at the ghost fluid nodes across the interface as the local pressure and velocity of the second medium, and the density is obtained via isobaric fixing. It has been found, however, that such a definition of ghost fluid status is not very robust when applied to gas-water flows [9]. Therefore, in [9], a new definition of the ghost fluid status was proposed

*Received by the editors October 23, 2003; accepted for publication (in revised form) August 29, 2005; published electronically March 10, 2006.
http://www.siam.org/journals/sisc/28-1/60136.html

†College of Science, Nanjing University of Aeronautics and Astronautics, Jiangsu, Nanjing 210016, PRC. Current address: Institute of High Performance Computing, The Capricorn, Science Park II, 117528, Singapore (wangcw@ihpc.a-star.edu.sg).

‡Institute of High Performance Computing, The Capricorn, Science Park II, 117528, Singapore (liutg@ihpc.a-star.edu.sg).

§Corresponding author. Department of Mechanical Engineering, National University of Singapore, 10 Kent Ridge Crescent, Singapore 119260 (mpekbc@nus.edu.sg).

especially for treating gas-water flows. More specifically, to obtain the ghost fluid status for the gas medium, the ghost fluid velocity is directly duplicated from the water medium, while the ghost fluid pressure is extrapolated from the gas medium. In the definition of ghost fluid status for the water medium, the ghost fluid pressure is directly duplicated from the gas medium, while the ghost fluid velocity is extrapolated from the water medium. Hereafter, for ease of reference, we shall call the GFM developed in [9] the new GFM in this work. With the ghost nodes defined across the interface, computation is thereby carried out as if in a single medium without the necessity to change the numerical scheme employed and hence the method is of the great versatility. The original and new GFMs have been successfully applied to various areas such as the treatment of discontinuities in compressible and incompressible flows [17, 18, 19], flame and detonation fronts [20, 21], and fluid-structure coupling [9]. The success of the GFM depends on the implicit capture and imposition of boundary conditions at the moving interface. Since the flow states at the interface are usually material properties related, the boundary conditions at the interface are therefore usually nonlinearly affected by the flow material properties from both sides of the interface. This implies that the definition of ghost fluid states by simply duplicating from the real fluid is not strictly sufficient to take into account such effects arising from the different material properties present on both sides of the interface. This is one of the reasons why the GFM presented in [8] has some limitations in its applications and is found to be problem related [9].

A strong wave interaction can occur at the interface such as in the situation of a strong shock impacting on a material interface. As such, the pressure or the velocity can experience a large gradient or sudden jump across the interface. In such a situation, the ghost fluid states have to be predicted to take into consideration the influence of both wave interaction and material properties on the interfacial evolution as analyzed in [22]. In fact, because the GFM provides a solution via essentially solving two single medium GFM Riemann problems, some conditions (i.e., Conditions I and II as proposed in [23]) have to be satisfied in the definition of the ghost fluid states to ensure that the two GFM Riemann problems indeed give the correct solution in the respective real fluid sides. This has led to the development of a modified ghost fluid method (MGFM) proposed by in [22]. In the MGFM, a Riemann problem is defined and solved approximately to predict the interfacial status. The predicted interfacial status is then used to define the ghost fluid status at the ghost nodes across the interface. The MGFM has been shown to be robust and much less problem related and can be applied to various equations of state (EOS).

In the original GFM, the new GFM, and the MGFM, the focus is on defining the ghost fluid states while the pressure and velocity in the real fluid side are taken for granted, except for the correction made to the density at the real fluid nodes next to the interface to overcome the overheating. It has been found that such an approach adopted by the GFM algorithm is not very accurate when applied to those problems with a critical condition like the shock impedance matching (-like) problem [22, 23]. Because such a critical condition is usually related to the flow material properties on both sides of the interface, a single-medium computational method like the GFM can encounter many difficulties in enforcing and maintaining this critical condition. This will remain so unless this condition is equivalently imposed in advance before the computation is carried out as shown in [22]. Furthermore, the flow motion relative to the moving interface is subsonic. As a result, the influence of material properties from both fluids and the wave interaction at the interface can propagate both upward and downward simultaneously in relation to the interface motion. This implies

that a correction to the real fluid nodes next to the interface is deemed necessary to accurately take into account such influence (especially for those critical problems of shock impedance matching). On the other hand, in all the previously developed GFM algorithms, only the ghost fluid states are defined or modified to take into account such influence. This means that the consideration of such influence on the real fluid is artificially delayed by at least one time step. Therefore, one motivation of the present work is to develop a technique to consider the influence mentioned above on the fluids (lying on either side of the interface) simultaneously to better reflect the real/true flow physics. To do so, the real fluid states next to the interface instead of the ghost fluid states will be predicted by solving a Riemann problem and the ghost fluid states are then obtained by solving an advection equation. This will result in the complete modification and redefinition of the real fluid next to the interface. In doing so, one will find that the nonphysical reflection for the shock impedance matching problem can be greatly and further suppressed. At the same time, all the good features of the MGFM are still maintained. The algorithm based on the present implementation is called real ghost fluid method (rGFM) for ease of reference and discussion in this work. Furthermore, we will show theoretically that the rGFM indeed enables a better imposition of the boundary conditions at the interface and less conservative errors are incurred in comparison to the previous GFMs.

The paper is organized as follows. In section 2, the governing equations, EOS for gases and water, and the level set equation are provided. To avoid or prevent the appearance of hugely distorted level set contours caused by (potential) large flow velocity gradient, an extension artificial velocity field is constructed for solving the level set equation. In section 3, the detailed implementation of the present method based on the third-order ENO-LF scheme [3, 4] for one- and two-dimensional problems is presented. Some conclusions based on analysis for the rGFM are also obtained in this section. Tests on the accuracy, convergence, and conservation errors incurred for the rGFM are given in section 4. In section 5 extensive comparisons are made between the simulations and analytical results. A brief conclusion is given in section 6.

2. Equations.

2.1. Governing equations. The Euler equations governing two-dimensional compressible flows are written as

$$\frac{\partial U}{\partial t} + \frac{\partial F(U)}{\partial x} + \frac{\partial G(U)}{\partial y} = 0, \tag{2.1}$$

where $U = [\rho, \rho u, \rho v, E]^T$, $F(U) = [\rho u, \rho u^2 + p, \rho uv, (E+p)u]^T$, and $G(U) = [\rho v, \rho uv, \rho v^2 + p, (E+p)v]^T$. Here ρ is the density, u and v are the velocity components in the respective x and y directions, p is the pressure, and E is the total energy per unit volume. The total energy is the sum of internal energy and kinetic energy,

$$E = \rho e + \frac{1}{2}\rho(u^2 + v^2), \tag{2.2}$$

where e is the internal energy per unit mass.

2.2. Equations of state(EOS). For closure of (2.1), EOS is required. In the present work, our interest centered on the compressible gas and water media. The EOS for compressible gases and water can be expressed in the following consistent form as

$$p = (\gamma - 1)\rho e - \gamma B, \tag{2.3}$$

where γ and B are treated as constants. For the air,

$$\gamma = 1.4 \quad \text{and} \quad B = 0. \tag{2.4}$$

For the water medium (Tait's equation),

$$\gamma = 7.15 \quad \text{and} \quad B = 3309. \tag{2.5}$$

2.3. Level set equation. To track the moving interface, we employ the level set technique [24, 25, 26]. The level set equation for two-dimensional flows can be written as

$$\frac{\partial \phi}{\partial t} + u\frac{\partial \phi}{\partial x} + v\frac{\partial \phi}{\partial y} = 0. \tag{2.6}$$

In general $\phi(x, y, t)$ starts off as a signed distance function. For complex interface evolution, it has been found that a higher-order accurate discretization can better help to maintain the accurate position of the interface [8]. In this work, the fifth-order weighted ENO scheme [27, 28] is, thus, used to discretize the spatial derivatives together with the third-order Runge–Kutta time discretization [3] to advance the interface.

The possible disadvantage of using the level set technique is that solving equation (2.6) may lead to severely distorted contours when the velocity field experiences a large gradient in the vicinity of the interface. To remedy this situation, a redefinition of the velocity field is called forth via the extension velocity. There are several ways of constructing the extension velocity field in the literature. In [29], the velocity at the interface point, which is closest to the considered nodes, is taken as the extension velocity. In [30], the extension velocity is obtained by solving an equation, which ensures the level set function to be a signed distance function, and so the reinitialization step is obviated. In this work, a very simple yet efficient method for obtaining the extension velocity is proposed. In particular, the normal velocity at the interface, which is obtained from the Riemann problem solver, is extended to all the nodes in the fluid field as the extension velocity. As the Riemann problem solver provides the accurate interface moving velocity, the uniformly distributed level set contour and accurate interface position can be obtained without any need of reinitialization. The detail of implementation is discussed in the following section.

3. rGFM. In the rGFM, a Riemann problem is defined at the interface and solved for predicting the interfacial status. The predicted interfacial status is then used to redefine the flow states at the real fluid nodes next to the interface, and the ghost fluid states at the ghost nodes across the interface are obtained by solving an advection equation. In doing so, the isobaric fix which is used in the previous GFMs to overcome the overheating is not required. To show the method clearly, we first present the treatment of the interface boundary, where the Riemann problem is defined, and the implementation of the extension velocity for one-dimensional problems. Then we extend it to the two-dimensional problems. Finally, we carry out some analysis for the rGFM on the issue relating to the imposition of boundary conditions at the interface and on the conservation errors incurred by the algorithm. In the implementation of the present rGFM, the spatial discretization is either the first-order ENO-LF or the third-order ENO-LF [4, 5] (ENO scheme with Lax–Friedrichs flux) scheme; the resultant temporal ordinary differential equation (ODE) system is then solved using the third-order TVD Runge–Kutta discretization [3]. The former algorithm is termed the first-order ENO-LF while the latter is termed the third-order ENO-LF.

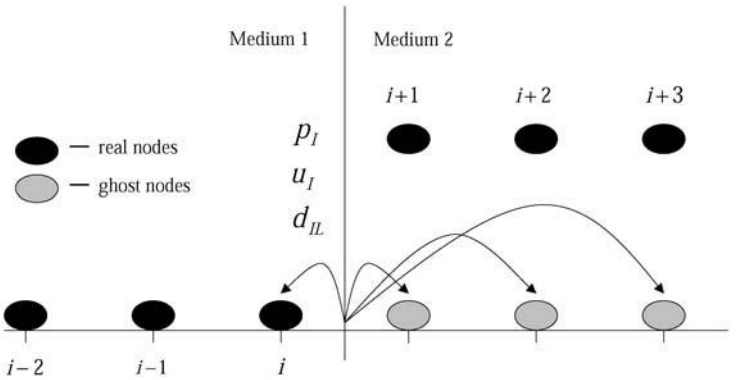


FIG. 1. *Updating the real and ghost nodes in the rGFM.* p_I*: interface pressure,* u_I*: interface velocity,* ρ_{IL}*: left interface density,* ρ_{IR}*: right interface density.*

3.1. For one-dimensional problems. Suppose that the interface lies between node i and node $i+1$ as depicted in Figure 1. To define the Riemann problem at the interface, the two initial constant states of the Riemann problem are simply given as U_L=U_i and U_R=U_{i+1} in the present paper. Such a simple definition works rather well due to the accurate imposition of boundary conditions at the interface (to be shown in section 3.4) and can be easily extended to two-dimensional cases on comparing to the previous GFMs. A better and perhaps simpler choice for U_L and U_R are U_{i-1} and U_{i+2}, respectively, which has been applied to the original GFM, the new GFM, and the MGFM due to employment of the isobaric fix (see [8] for more details). The latter has been found to also work equally well for the rGFM. Once the Riemann problem is defined, the Riemann problem solver [22, 26] can then be employed to provide intermediate interfacial states—p_I(pressure), u_I(velocity), and ρ_{IL} and ρ_{IR} representing the densities on the left and right sides of the interface, respectively. For the third-order ENO scheme [4], to update the states at the real fluid nodes next to the interface and the ghost fluid states at ghost nodes, we need to redefine the value for medium 1 at nodes from node i to node $i+3$. (Node i is a real fluid node, while $i+1$, $i+2$, and $i+3$ are ghost fluid nodes.) We take p_I, u_I, and ρ_{IL} as the pressure, velocity, and density at these said nodes. Likewise, for medium 2, we also take p_I, and u_I as the pressure and velocity at nodes $i+1$, i, $i-1$, and $i-2$. The density at these latter nodes is ρ_{IR}. Thus, the boundary conditions at the interface for medium 1 and medium 2 are imposed, and the solution can be advanced to the next time step.

The extension of the level set velocity field is very simple for one-dimensional problems. Essentially, one can just copy the velocity u_I from the Riemann problem solver to all the nodes in the fluid field. By solving for the one-dimensional equation (2.6), the level set function can be obtained at the new time step. Obviously, the signed distance function is maintained using the present extension velocity technique for the one-dimensional problems.

3.2. For two-dimensional problems. Defining the Riemann problem and obtaining the flow states at ghost nodes seems simple and straightforward for one-dimensional problems. However, for two-dimensional problems, it becomes much more involved because there is more than one velocity component to deal with. One has to construct the Riemann problem appropriately and then extrapolate the states to the associated ghost nodes accordingly. Using the level set function , we can define

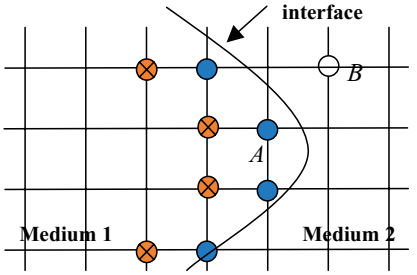


FIG. 2. *Construction of the Riemann problem.*

the unit normal at every grid node as

$$N = \frac{\nabla \phi}{|\nabla \phi|}. \tag{3.1}$$

The normal velocity at those nodes next to the interface is given by

$$u_N = (u, v) \cdot N. \tag{3.2}$$

First, we present the procedure of defining the Riemann problem and extrapolation for medium 1. Considering node A (see Figure 2), which is just bordering the interface, we have the normal velocity, u_N^A, the density, ρ^A, and the pressure, p^A. Next, we search for another node B also bordering the interface but on the other side of the medium such that the angle made by the respective normals at node A and node B is the minimum. In addition, we know the normal velocity, the density, and the pressure at node B, as denoted by u_N^B, ρ^B, and p^B, respectively. Therefore we can define a Riemann problem in the normal direction with the initial condition,

$$U_0 = \begin{cases} U_A, \\ U_B, \end{cases} \tag{3.3}$$

where $U_A = [\rho^A, u_N^A, p^A]^T$, $U_B = [\rho^B, u_N^B, p^B]^T$. We then solve for the Riemann problem to provide for an intermediate state ρ_L^I, ρ_R^I, u_N^I, and p^I.

The intermediate states are employed to redefine the flow states at those real fluid nodes next to the interface. After updating all the nodes bordering the interface in the real fluid, the ghost fluid states are then obtained by solving the following convection equation:

$$I_t \pm N \cdot \nabla I = 0 \tag{3.4}$$

(see also [8]). This is done by fixing the value of I in the real fluid region (where $\phi < 0$) and iterating (3.4) to steady state for all the ghost fluid nodes. Here I is a column vector comprising the normal velocity, tangential velocity, pressure, and density or entropy. A first-order ENO scheme is used to discretize the spatial derivatives. Likewise, for the calculation of medium 2, a node A in medium 1 can also be found corresponding to node B in medium 2 such that the angle made by the normal of node B and that of node A is the minimum. Therefore the Riemann problem can be constructed and then the ghost fluid states can be defined (in medium 1) by solving (3.4). The above procedure of defining and solving the Riemann problem gives the normal velocities for all the grid nodes next to the interface. By fixing the extension

velocity for all those nodes next to the interface and solving for (3.4) with a reverse convection direction to that for obtaining the ghost fluid states, the normal velocity field of the extension velocity is obtained. We use this normal velocity to define the extension velocity at any node C as

$$(u_1^C, v_1^C) = u_N^C \cdot N^C, \tag{3.5}$$

where u_N^C is the extended normal velocity and N^C is the normal vector at node C. We then advance the level set function to the next time step. The present technique is very simple and works rather efficiently, as the interface is predicted correctly. Unlike the one-dimensional implementation, the two-dimensional implementation of advancing the level set function theoretically does not guarantee that the level set function is maintained as a signed distance function. However, our applications have shown that the distance function is approximated and maintained quite well. The level set contours are almost uniformly distributed around the interface.

3.3. Summary of solution procedures. Here we briefly summarize the implementation procedures for the rGFM. The level set function is defined according to the initial condition of the problem at the first time step. Assuming that the flow states at $t = t^n$ have been obtained, the following steps are taken to calculate for the respective quantities at the next time step:

1. Construct the Riemann problem for medium 1, predict the interface states, and then redefine the flow states for those nodes next to the interface in the real fluid. Then obtain the ghost fluid states by solving (3.4).
2. After the boundary conditions are imposed, the third-order ENO-LF scheme is employed for discretizing the spatial derivatives of the governing equations. The solution in medium 1 is advanced to the next intermediate time level.
3. Repeat steps 1 and 2 for the next medium 2 until the solution in the whole domain is advanced to the next intermediate level.
4. Construct the extension velocity in the fluid field and solve for the level set equation to get the level set function at the next intermediate time step.
5. By repeating steps 1, 2, 3, and 4 at each intermediate time step of the third-order TVD Runge–Kutta method, the solution can be advanced from U^n to U^{n+1}.

3.4. Some conclusions on the rGFM. Numerical tests and applications to be given in sections 4 and 5 will show that the rGFM can give better numerical results even in comparison to the MGFM. In this subsection, analysis is carried out to better understand the underlying reasons; the analysis is based on one-dimensional flows. In the discussion below, the interface is supposed to lie between node i and node $i + 1$. We have the following findings.

THEOREM 1.

Physically, the initial singularity is instantly decomposed, and both flow states next to the interface change simultaneously to meet the interfacial conditions (the continuity of pressure and normal velocity). Numerically, this may not be captured faithfully by the previous GFM-based algorithms. As a result, numerical errors are introduced. This can happen as shown in [22] for the original GFM when applied to the strong shock impacting on the gas-water interface. Due to the usual employment of a high-resolution scheme, which is usually upwind based, the real fluid may not be affected if the initial flow state of this real fluid is supersonic. More specifically, for

illustration, the initial flow in medium 1 is supersonic as for a jet impact problem, and medium 2 is stationary and subject to low pressure. If the jet is of large magnitude, a full up-winded or Roe-type scheme will lead to numerical fluxes, $F_{i-\frac{1}{2}}=F_{i-1}$ and $F_{i+\frac{1}{2}}=F_i$. The fluid in medium 2 thereby exerts no influence on the flow at node i in the first step of computation for those one-step algorithms. Consequently, the decomposition of singularity in medium 1 will be at least one time step lagging behind that in medium 2, which results in numerical errors, while physically the decomposition of singularity at the interface occurs simultaneously in both the upstream and downstream directions. However, such nonsimultaneous decomposition does not occur for the rGFM. This is because the influence of fluid properties on the singularity decomposition is taken into account via solving an interfacial Riemann problem and the real fluids at the nodes next to the interface are also modified to consider such influence. Independent of the numerical scheme employed whether it be centered or up-winded, the numerical flux, $F_{i+\frac{1}{2}}$, will include the influence from medium 2 in updating the flow state at node i for medium 1. Therefore, the singularity is simultaneously decomposed in both fluids.

THEOREM 2.

There are easily two kinds of error arising from treatment of the interface using a previous GFM-based algorithm. One is that the interfacial location may not be captured accurately due to the nonconservative computation. The other is that the boundary conditions may not be enforced correctly. That is, the continuity of pressure and normal velocity across the interface may not be maintained. As pointed out in [22, 23], a GFM-based algorithm essentially consists of solving two GFM Riemann problems. If there is an initial singularity at the interface, neither the original GFM nor the new GFM can faithfully maintain the continuity of pressure and/or normal velocity across the interface in the very earlier stages of singularity decomposition. This is because there is no such mechanism for the pressure and/or normal velocity continuity initially using either GFMs, and the two GFM Riemann problems usually provide different flow status at the location of interface [22, 23]. On the other hand, the rGFM modifies the real fluid status at those nodes next to the interface via the predicted interfacial pressure and normal velocity and then extends to the ghost fluid nodes by solving equation (3.4) for each substep of the Runge–Kutta method. This ensures the pressure and normal velocity at the interfacial location are the same for the two GFM Riemann problems. In other words, the continuity of pressure and normal velocity at the interface is imposed. As for the MGFM, the ghost fluid status is directly defined using the predicted interfacial status, which is obtained via solving an interfacial Riemann problem. Because the interfacial location generally does not coincide with the ghost fluid nodes, the interfacial pressure and normal velocity in one GFM Riemann problem are not equal to the respective quantities in the other GFM Riemann problem, albeit they are quite close to each other, as shown in [22]. Consequently, the rGFM can better enforce the continuity of the interfacial pressure and normal velocity, while the other mentioned GFM algorithms do not strictly uphold such a property.

Because the present rGFM is nonconservative like all the other previous GFM algorithms, conservation errors are incurred. The occurrence of conservation errors is one of the causes for inaccurate results for some problems. Being so, the reduction of conservation errors is especially relevant and certainly critical in the construction of a viable method for treating the interface. For a GFM algorithm-based on the third-order TVD Runge–Kutta time discretization [3] (independent of the spatial discretization), we have the following findings.

THEOREM 3. [illegible] [3] [illegible] x_A, x_B [illegible]

$$TCE(t) = DETAT(t) + DETAS(t) + STEP1(t) + STEP2(t) + STEP3(t), \tag{3.6}$$

[illegible]

$$DETAT(t) = \frac{1}{\triangle x}\left(\int_{x_A}^{x_I^{n+1}} U_L^{n+1}dx - \int_{x_A}^{x_I^n} U_L^n dx + \int_{x_I^{n+1}}^{x_B} U_R^{n+1}dx - \int_{x_I^n}^{x_B} U_R^n dx\right),$$

$$DETAS(t) = \frac{1}{6\triangle x}\int_{t^n}^{t^{n+1}} \left[F_B^{(1)} - F_A^{(1)} + F_B^{(2)} - F_A^{(2)} + 4\left(F_B^{(3)} - F_A^{(3)}\right)\right] dt,$$

$$STEP1(t) = \frac{1}{6\triangle x}\int_{t^n}^{t^{n+1}} \left(F_{IL}^{(1)} - u_{IL}^{(1)}U_{IL}^{(1)} - F_{IR}^{(1)} + u_{IR}^{(1)}U_{IR}^{(1)}\right) dt,$$

$$STEP2(t) = \frac{1}{6\triangle x}\int_{t^n}^{t^{n+1}} \left(F_{IL}^{(2)} - u_{IL}^{(2)}U_{IL}^{(2)} - F_{IR}^{(2)} + u_{IR}^{(2)}U_{IR}^{(2)}\right) dt,$$

$$STEP3(t) = \frac{2}{3\triangle x}\int_{t^n}^{t^{n+1}} \left(F_{IL}^{(3)} - u_{IL}^{(3)}U_{IL}^{(3)} - F_{IR}^{(3)} + u_{IR}^{(3)}U_{IR}^{(3)}\right) dt.$$

Here $DETAT(t)$ denotes the increase within the fluid domain, $DETAS(t)$ is the flux out of the domain, while $STEP1(t)$, $STEP2(t)$, and $STEP3(t)$ are the errors associated with the interfacial treatment in the three intermediate time steps (or substeps) of the third-order TVD Runge–Kutta method, respectively. $F_{IL}^{(i)}$ and $F_{IR}^{(i)}$ are the fluxes on the left and right sides of the interface, and $U_{IL}^{(i)}$ and $U_{IR}^{(i)}$ are the value of the conservative variables on the left and right sides of the interface. $u_{IL}^{(i)}$ and $u_{IR}^{(i)}$ are the interfacial velocities provided by the respective GFM Riemann problems. Subscript L or R indicates the left or the right flow medium, while superscript i stands for the ith substep of the third-order Runge–Kutta method. We can obtain

$$\begin{aligned} F_{IL}^{(i)} - u_{IL}^{(i)}U_{IL}^{(i)} &= \left[0, p_{IL}^{(i)}, u_{IL}^{(i)}p_{IL}^{(i)}\right]^T, \\ F_{IR}^{(i)} - u_{IR}^{(i)}U_{IR}^{(i)} &= \left[0, p_{IR}^{(i)}, u_{IR}^{(i)}p_{IR}^{(i)}\right]^T, \end{aligned} \tag{3.7}$$

where $p_{IL}^{(i)}$ ($u_{IL}^{(i)}$) and $p_{IR}^{(i)}$ ($u_{IR}^{(i)}$) are the interfacial pressures (velocities) provided by the respective GFM Riemann problems. Theoretically, $u_{IL}^{(i)}$ should be equal to $u_{IR}^{(i)}$, and $p_{IL}^{(i)}$ should be equal to $p_{IR}^{(i)}$, and thus $STEP1(t)$, $STEP2(t)$, and $STEP3(t)$ should be equal to zero. However, because the GFM algorithm obtains solution essentially via computation in the single medium flow and both GFM Riemann problems are usually unable to provide identical pressure and normal velocity at the interface, $STEP1(t)$, $STEP2(t)$, and $STEP3(t)$ are not equal to zero. Consequently, there is strictly no conservation over the whole computation domain. Using the rGFM, however, $STEP1(t)$, $STEP2(t)$, or $STEP3(t)$ can be made to vanish. Thus, we have the following.

COROLLARY 1. [illegible]

$$TCE(t) = DETAT(t) + DETAS(t). \tag{3.8}$$

This is because the rGFM imposes the pressure and velocity boundary conditions at the interface in the process of defining the ghost fluid status as shown in Theorem 2. Therefore, Corollary 1 is obviously true.

To maintain conservation, the sum of $DETAT(t)$ and $DETAS(t)$ should be equal to zero. This is generally not true for the rGFM. The conservation errors incurred in the rGFM come primarily from the modification to the real fluid status at the nodes next to the interface. This is unlike the original GFM, the new GFM, and the MGFM, where besides the conservative errors associated with the isobaric fix imposed on the real fluid nodes, there are other conservative errors arising from the inaccurate imposition of boundary conditions at the interface (i.e., $STEP1(t)$, $STEP2(t)$, and are generally not equal to zero). Numerical tests in section 4 will show that the conservative errors arising from employing the rGFM are much less than those associated with using the original GFM, the new GFM, or even the MGFM.

4. Preliminary tests of the rGFM. To better understand the rGFM, some tests on accuracy and convergence of the rGFM will be done in this section.

4.1. On accuracy and convergence. We select a one-dimensional model of translating density profile first proposed by Davis [31] to test the accuracy and convergence of the rGFM. This is a linear model in which a specified density profile translates with a constant velocity and pressure. There are no reflection or transmission waves in this problem. The computation domain is $[-1, 1]$ and the CFL is equal to 0.5. The given initial density profile is

$$\rho(x,0) = \begin{cases} -0.8x^3 + 0.4, & x < 0, \\ -0.8x^3 + 2.0, & x > 0, \end{cases} \tag{4.1}$$

and the initial velocity and pressure are constant at $u = 1.0$ and $p = 2.0$. The heat conductivity ratio is $\gamma = 1.4$. The convergence and the accuracy of the density calculated with the rGFM-based third-order ENO-LF scheme are provided in Table 1 for different mesh sizes by doubly refining the mesh repeatedly. The numerical results show that the algorithm is third-order accurate in the L_∞ norm and even higher than third-order accurate in the L_1 norm. This means that the rGFM does not incur more errors that can lead to the degeneration of accuracy and convergence for this problem although the present method is not conservative. The density profile at $t = 0.2$ is given in Figure 3 when the interface has moved to $x = 0.2$ from the initial position of $x = 0.0$. The comparison to the theoretical solution shows that the rGFM captures the interface accurately for this problem.

4.2. The conservation errors. We shall use two cases to study the conservative errors of the rGFM. Case 1 is a high-speed air jet impacting on an air-water interface (i.e., Problem 4 in section 5). Case 2 is the oscillating water column problem proposed in [6]. In the discussion below, we again assume that the interface lies between node i and node $i + 1$.

TABLE 1
The convergence and the accuracy of density at $t = 0.2$.

Mesh size	L_∞ error	order	L_1 error	order
0.04	4.10E-04	-	5.27E-06	-
0.02	4.49E-05	3.19	4.04E-07	3.70
0.01	5.38E-06	3.06	3.46E-08	3.54
0.005	6.87E-07	2.97	3.17E-09	3.45
0.0025	8.65E-08	2.99	2.84E-10	3.48

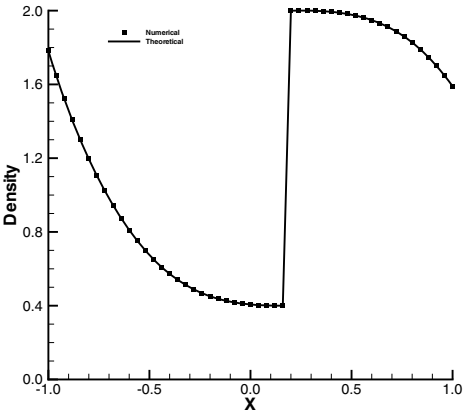


FIG. 3. *The density profile obtained at* $t = 0.2$.

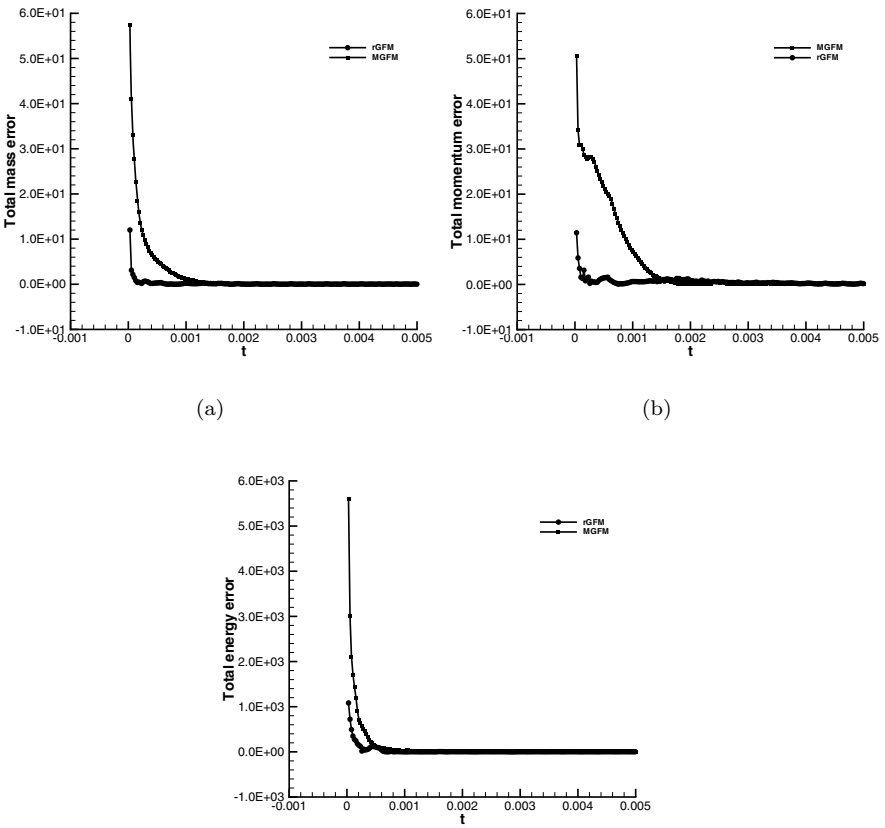


FIG. 4. *Comparison of conservation errors betwee the rGFM and the MGFM for Case* 1.

1. In this problem, the jet is supersonic. Figures 4(a), (b), and (c) shows the conservation errors for the total mass, momentum, and energy in the computational domain, respectively, for both the rGFM and the MGFM. The results are obtained

FIG. 5. *Initial status of the oscillating water column problem.*

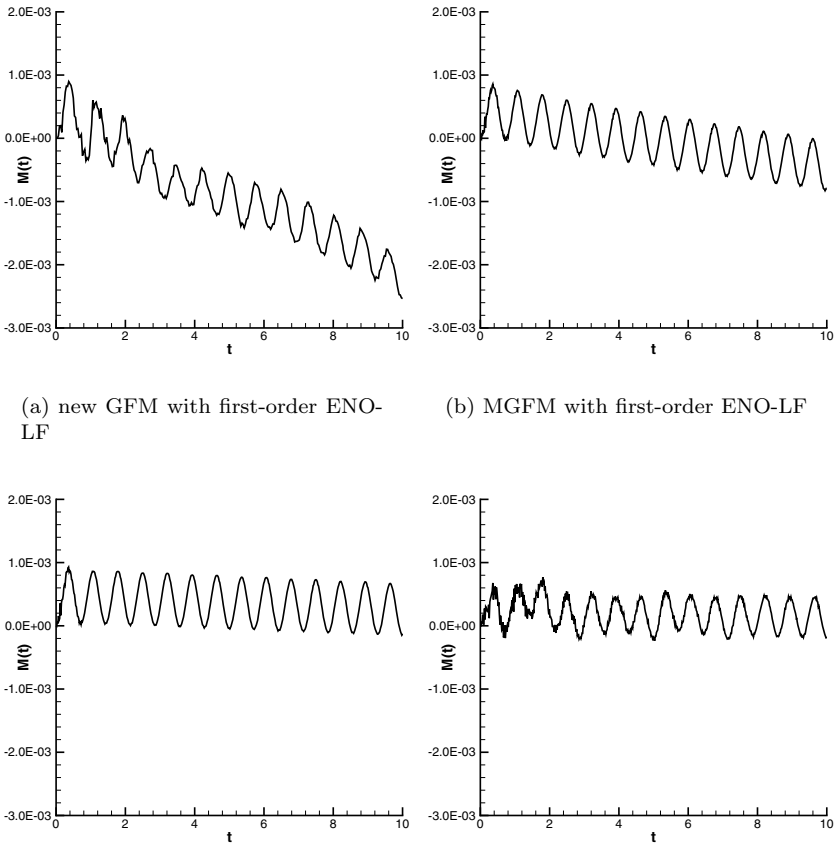
using formula (3.6). From these figures, it is apparent that large conservation errors are incurred in the first a few time steps of the computation via the MGFM. The errors, however, tend toward zero fairly quickly. The results also show that there are relatively much smaller errors incurred by the rGFM in the first several time steps of computation on comparing to the MGFM. These errors tend toward zero more quickly for the rGFM than for the MGFM. The results indicate that the conservative errors associated with the rGFM are even more contained and limited compared to the MGFM. It is clear that the redefinition of the real fluid states at the nodes bordering the interface and the accurate calculation of the numerical flux next to the interface are important at the beginning of the singularity decomposition. Since the flow states at the real fluid nodes in the rGFM are redefined with the Riemann problem solver rather than only for the ghost fluid nodes as carried out in the MGFM [22], the numerical flux $F_{i+\frac{1}{2}}$ is more accurately obtained and is independent of the numerical scheme selected as analyzed in Theorem 1. More important, according to Corollary 1, $STEP1(t)$, $STEP2(t)$, and $STEP3(t)$ are equal to zero in the rGFM, while these are not so in the very earlier stage of the singularity decomposition for the MGFM (or other previous GFMs). The far smaller conservation errors found support our analysis in section 3.4 that the rGFM imposes better and more accurate interfacial conditions than the previous GFMs mentioned. (Needless to say, the conservative errors pertaining to the previous GFMs in [8, 9] are even larger than the MGFM as already discussed and analyzed in [22, 23].)

EXAMPLE 2. This is a problem of a water column separated by the air and oscillating in the middle of a closed tube [6]. The problem is sketched in Figure 5. The initial status of the air and water are defined as $\gamma_a = 1.4$, $B_a = 0.0$, $\rho_a = 0.001$ and $\gamma_w = 7.0$, $B_w = 3000.0$, $\rho_w = 1.0$, respectively. The velocity is $u_a = u_w = u_0 = 1.0$, and the pressure is $p_a = p_w = p_0 = 1.0$. The two interfaces are located at $x_{IL} = -0.1$ and $x_{IR} = 0.1$.

With the fluids moving to the right, the air on the left of the water column expands while the air on the right of the water column is compressed. The pressure decreases at the former and increases at the latter; this leads to the deceleration and eventually stagnation of the fluids in the tube. Then the fluids move in the reverse direction due to the pressure gradient, and so on with the water column oscillating. Pressure waves are evident in the tube due to the impact of the air on the end boundaries and interaction with the water column. It is found and reported in [6] that the air mass has been erroneously converted into water. Figures 6(a) to (d) show the time evolution of the conservation errors of the air mass $M(t)$ computed with 80 grid points for the new GFM, the MGFM, and the rGFM. The said errors are given as

$$M(t) = \frac{m_a(t) - m_a(0)}{m_a(0)}, \tag{4.2}$$

where $m_a(t)$ is the total mass of the air in the tube at time t. It can indeed be observed that the conservation errors associated with the air mass appear to be oscillating similar to that observed in [6]. For the new GFM [9], which is especially formulated for the gas-water interface, the trend of the air mass loss looks large (see Figure 6(a)).



(a) new GFM with first-order ENO-LF

(b) MGFM with first-order ENO-LF

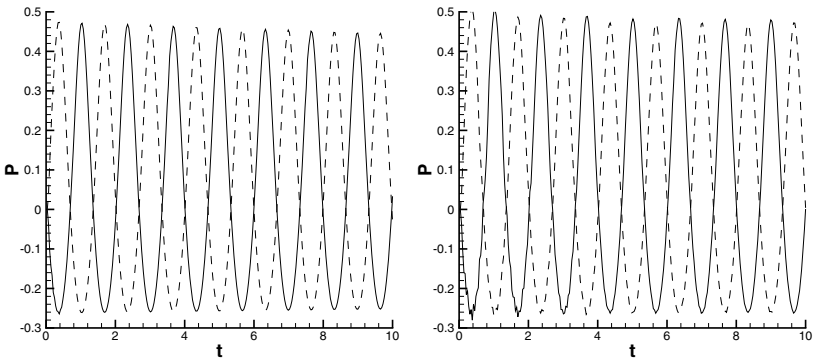
(c) rGFM with first-order ENO-LF

(d) rGFM with third-order ENO-LF

FIG. 6. *Comparison of conservation errors among the new GFM, the rGFM and the MGFM for Case* 2.

With employment of the MGFM, Figure 6(b) also depicts an oscillatory $M(t)$ but with a much gentler trend of net conversion of the air mass into water when compared to Figure 6(a). The results for the rGFM are shown in Figures 6(c) and 6(d) based on the first-order ENO-LF and the third-order ENO-LF, respectively. It is clear that while the amplitude of oscillations for Figure 6(b) is fairly comparable to Figure 6(c) the net conversion of the air mass with time as measured by the slope is obviously gentle for the latter. For Figure 6(d) computed with the third-order ENO-LF, the amplitude of oscillations is about the least and the rate of net conversion is the lowest; in fact, there is no considerable conversion of the air mass to water. The results further suggest that the conservative errors of the rGFM are limited and the rGFM espouses better characteristics than the previous GFMs in treating the interface. Figures 7(a) and (b) show the time evolution of the pressure coefficients at the wall boundaries calculated using the first-order ENO-LF and the third-order ENO-LF rGFM schemes, respectively. The wall pressure coefficients are given as

$$P(x=-1,t)=\frac{p(x=-1,t)-p_0}{p_0},\quad P(x=1,t)=\frac{p(x=1,t)-p_0}{p_0}. \tag{4.3}$$



(a) rGFM with first-order ENO-LF (b) rGFM with third-order ENO-LF

FIG. 7. *Time evolution of the pressure coefficients at boundaries. Solid line: at the left boundary; dashed line: at the right boundary.*

Both Figure 7(a) and Figure 7(b) show that there is a slight decrease of the pressure amplitude with time with the decrease in Figure 7(b) even smaller. It may be noted that an equivalent plot of P versus t in [6] also depicts a decrease of pressure amplitude of oscillations with time. Closer scrutiny suggests that the amplitude decrease for P is slightly smaller for the rGFM.

5. Applications to one- and two-dimensional problems. In this section, several problems are presented to test and further validate the present method and then applied to more challenging problems. In particular, Problem 1 is a gas-gas shock tube problem used previously in [8]. The next test carried out in Problem 2 is a gas shock impacting on a gas-water interface as also shown previously in [22]. Problems 3 and 4 with severe initial conditions are selected to show the ability of the rGFM to solve challenging cases. In Problem 3, the results obtained with the original GFM [8] and the MGFM [22] have been given for comparison. In the supersonic jet impacting case of Problem 4, the shock position as captured by the present method is identical to the theoretical solution. Problem 5, which is actually a one-dimensional axis-symmetric problem for single medium flow, is employed to validate the assembled two-dimensional rGFM code. Finally, the example on Problem 6 gives the interaction of two underwater high-pressure gas cylinders, which can be construed as two underwater explosions. All parameters in the problems are assumed to be nondimensional. It should be mentioned that although only four selected one-dimensional examples are presented below, the rGFM has been employed to test all other one-dimensional problems found in [8, 22], where analytical solutions are available. All results provided via the rGFM compared well to the respective analytical solutions (not shown here).

The approximate Riemann problem solver as detailed in [22, 26] is employed to supply the intermediate states. The calculations for the original GFM, the new GFM, the MGFM, and the present rGFM are carried out using the third-order ENO-LF. It is different from the schemes adopted in [8, 9] and [22], where the third-order ENO-LLF and MUSCL are adopted, respectively. The CFL number is taken as 0.5. For all the one-dimensional problems the computational domain is taken as [0, 1] with 201 nodes uniformly distributed unless stated otherwise.

Problem 1: Shock tube problem [8]. This is a gaseous shock tube problem taken from [8] and the initial condition is given as follows. The interface is set at the

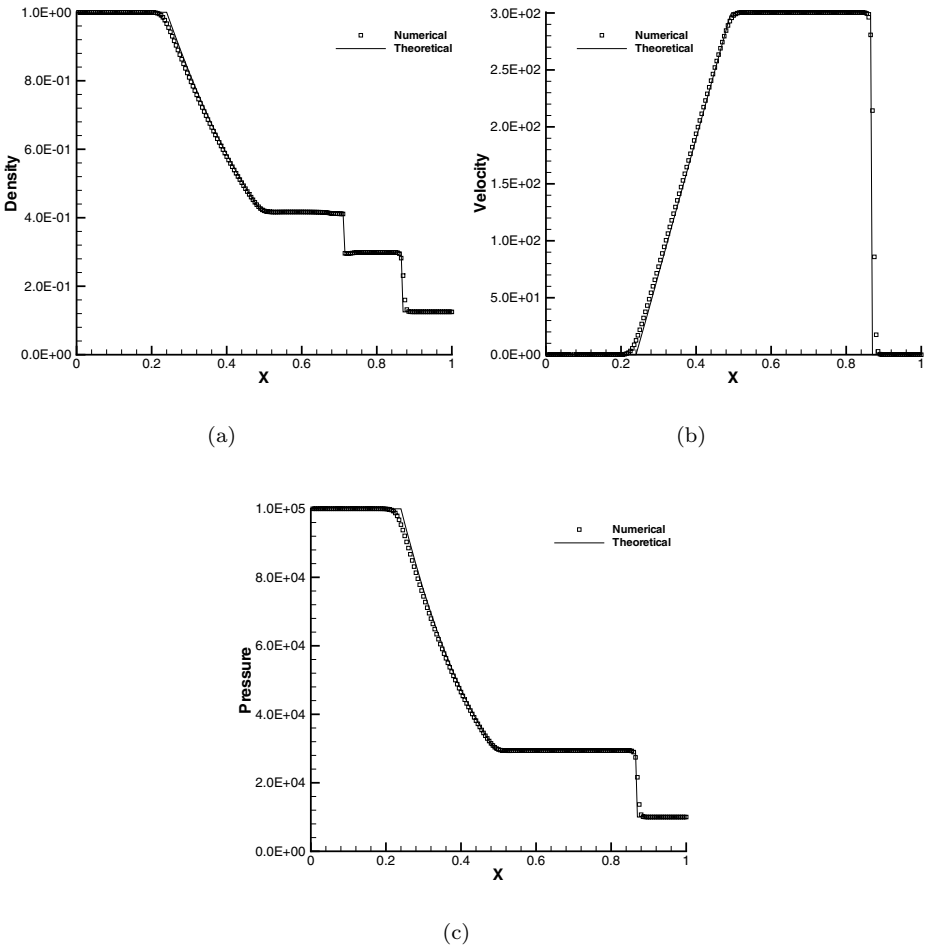


FIG. 8. *Problem* 1*: Comparison of the rGFM results with the theoretical solution at* $t = 0.0007$.

middle of the computational domain [0, 1], the left and the right states are defined respectively as $\gamma_L = 1.4$, $\gamma_R = 1.2$; $\rho_L = 1.0$, $\rho_R = 0.125$; $p_L = 10^5$, $p_R = 10^4$; $u_L = u_R = 0.0$ and $B_L = B_R = 0.0$. Figures 8(a), (b), and (c) show that the density, velocity, and pressure obtained by the rGFM compare reasonably to the theoretical results at time 0.0007. The position of the interface and the shock front are predicted accurately.

Problem 2: Strong shock impacting on a gas-water interface [22]. This is a case taken directly from Problem 7 in [22]. A shock is located at the same position as the interface, which lies between the 100th and 101th node points. The shock is defined as $\gamma_S = 1.4$, $\rho_S = 0.00596521$, $u_S = 911.8821$, $p_S = 1000$ and the states on the left and right sides of the interface are defined respectively as $\gamma_L = 1.4$, $\gamma_R = 7.15$; $\rho_L = 0.001$, $\rho_R = 1.0$; $p_L = 1.0$, $p_R = 1.0$; $u_L = u_R = 0.0$ and $B_L = 0.0$, $B_R = 3309.0$. Figures 9(a), (b), and (c) show that the numerical results obtained by the rGFM agree well with the theoretical results at time 0.0007. The locations of both the interface and the shock front are predicted accurately.

Problem 3: Gaseous shock in impedance-matching medium [22]. In this case a shock on the left side of the interface impacts on a gas-gas interface; this is

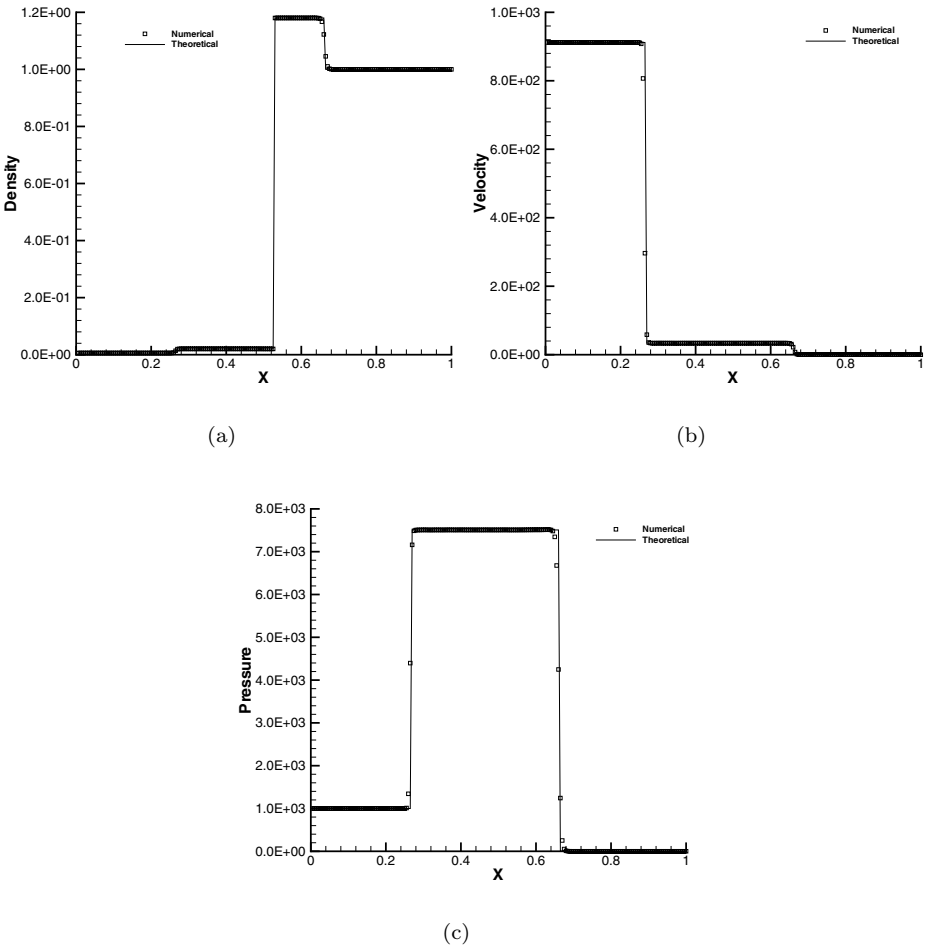


FIG. 9. *Problem 2: Comparison of the rGFM results with the theoretical solution at* $t = 0.0007$.

identical to Problem 6 in [22]. The shock strength is 100 and the initial position of the shock is same as the interface, which is located at $x_I = 0.2$ of the domain $[0, 1]$. The states on the left and right sides of the interface are defined respectively as $\gamma_L = 1.667$, $\gamma_R = 1.2$; $\rho_L = 0.82369077$, $\rho_R = 1.0$; $p_L = 1.0$, $p_R = 1.0$; $u_L = u_R = 0.0$ and $B_L = B_R = 0.0$. Based on the analysis on the interaction of the shock reflection at the interface, the shock is transmitted to the right side of the interface and there is no reflection of any kind. However, the calculation with the original GFM and the MGFM shows the obvious presence of nonphysical reflection, which occurs due to the inability of accurately imposing the shock impedance matching (-like) condition at the interface [22, 23]. Whether the nonphysical reflection is a rarefaction wave or compressive wave (shock) can be ascertained or determined beforehand for the original and new GFMs as discussed in [23]. Shown in Figures 10 and 11 are the respective results obtained by the original GFM and the MGFM at time $t = 0.06$. Already, there is marked improvement in the results of the MGFM where the nonphysical hump in the profile is greatly diminished compared to the original GFM [8]. For this same gas-gas problem, very severe nonphysical reflection occurs for the results using the new GFM [9] and is not presented here. Theoretically as discussed at length in [22],

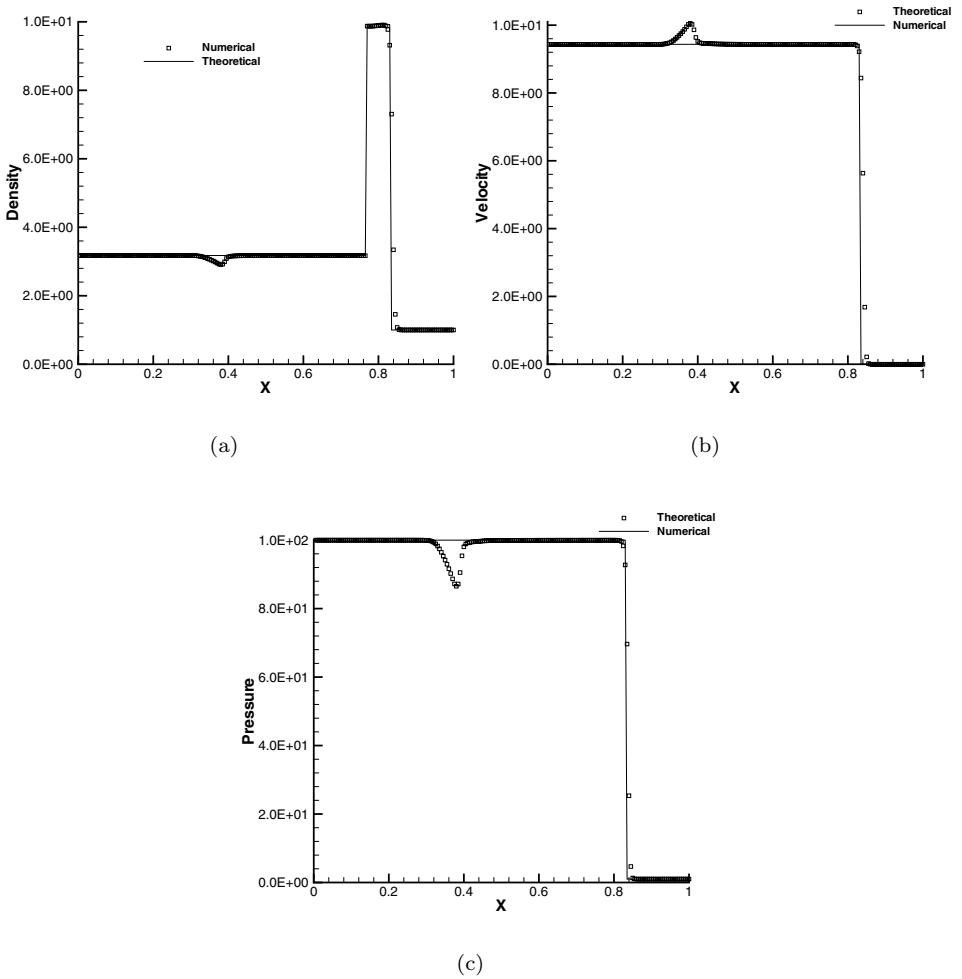


FIG. 10. *Problem 3: Comparison of the original GFM results with the theoretical solution at* $t = 0.06$.

such nonphysical reflection is indeed very difficult to remove completely. On the other hand, Figures 12(a), (b), and (c) show the density, velocity, and pressure obtained by the rGFM; these results clearly depict further improvement compared to the MGFM, although the reflection is still about perceptible.

Problem 4: Jet impact on the gas-water interface. This is an even more challenging problem. The initial conditions are given as follows: $\gamma_L = 1.4$, $\gamma_R = 7.15$; $\rho_L = 1.0$, $\rho_R = 1000.0$; $p_L = 1.0$, $p_R = 1.0$; $u_L = 90.0$, $u_R = 0.0$; and $B_L = 0.0$, $B_R = 3309$. The interface is located initially at $x_I = 0.6$. The results are obtained at time $t = 0.015$. Figure 13 shows the results calculated using the MGFM, while Figure 14 depicts the counterpart obtained via the rGFM. As analyzed in section 4.2, the not-so-perfect simultaneous decomposition of the singularity at the first few time steps of the shock interacting with the interface using the MGFM has resulted in a slightly faster left-moving shock wave, as can be observed in Figure 13(b) or (c). On the other hand, the computation with the present rGFM depicted in Figure 14 shows that the density, velocity, and pressure concur very well with the theoretical results.

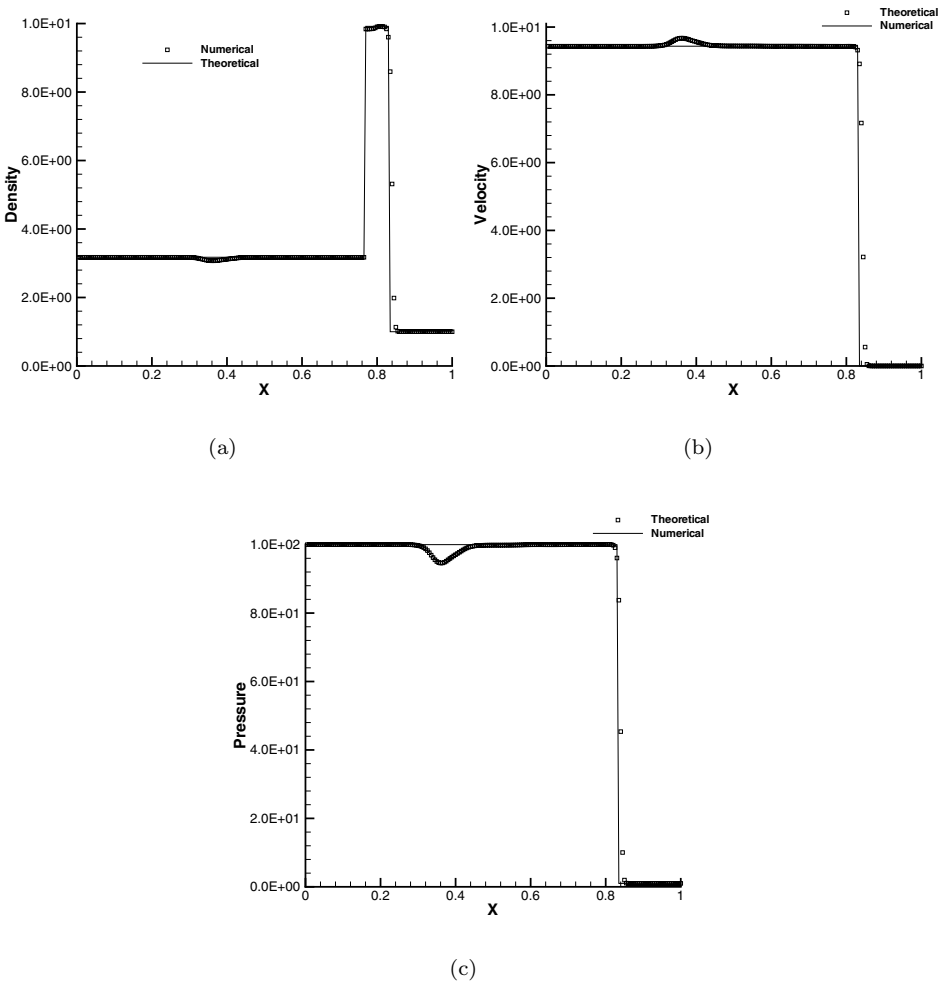


FIG. 11. *Problem 3: Comparison of the MGFM results with the theoretical solution at* $t = 0.06$.

Problem 5: A two-dimensional gaseous explosion in air. We consider a (pure air) single medium flow problem to test the assembled two-dimensional code for the rGFM. The computational domain is taken as $[0, 1] \times [0, 1]$. A gaseous bubble with a radius of 0.1 is located at the center of the domain. The state inside the bubble is $\gamma_1 = 1.4$, $\rho_1 = 1.0$, $u_1 = 0.0$, $v_1 = 0.0$, $p_1 = 100.0$; the state outside the bubble is $\gamma_2 = 1.4$, $\rho_2 = 1.0$, $u_R = 0.0$, $v_2 = 0.0$, $p_2 = 1.0$. This is actually a one-dimensional axis-symmetric problem for single medium flow. Hence we can simulate this problem using the third-order ENO-LF in conjunction with the third-order TVD Runge–Kutta method for the whole computational domain without involving any special treatment for the interface (the results are termed as ENO-LF in the plots) for comparison to the rGFM. The computational domain is divided uniformly by 201×201 nodes.

For this problem, once the diaphragm separating the high and low pressure sides is removed, a strong shock is generated and propagates radially outward. Simultaneously, a rarefaction wave is generated and traveling toward the center of the gas bubble, and a contact discontinuity (interface), across which a density jump occurs,

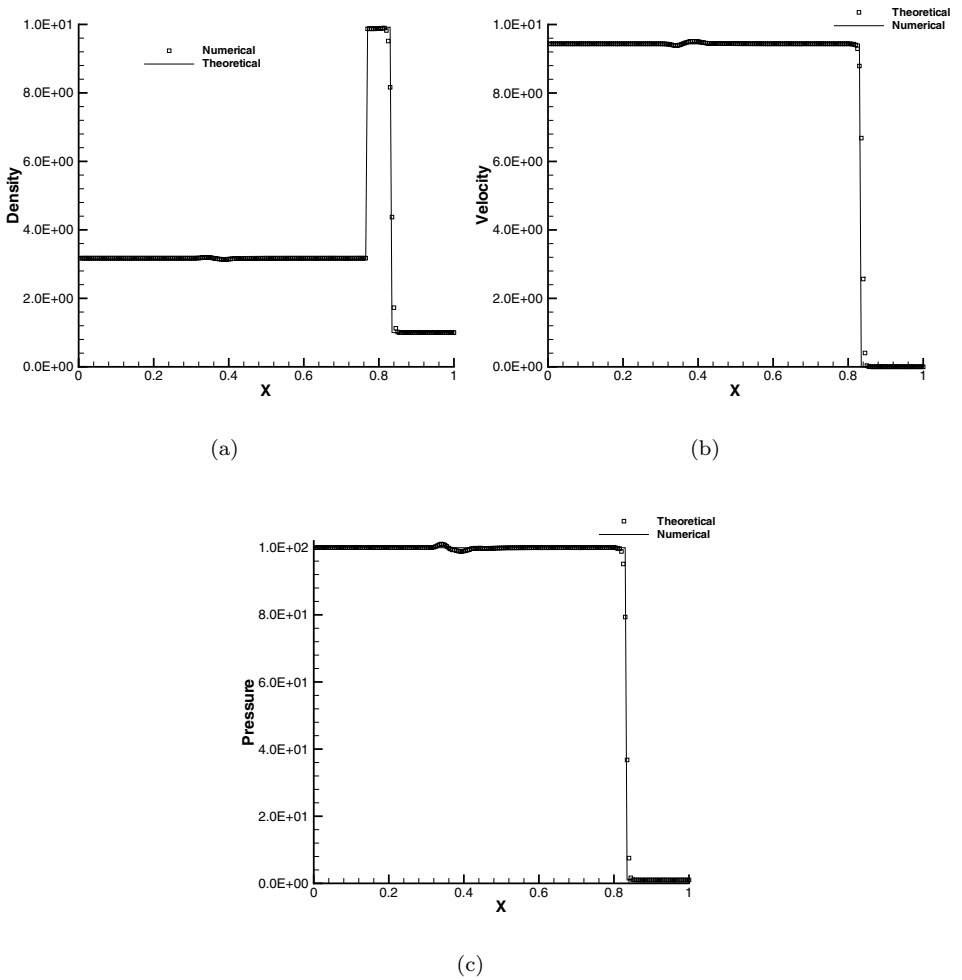


FIG. 12. *Problem 3: Comparison of the rGFM results with the theoretical solution at* $t = 0.06$.

is created. Due to overexpansion, a shock is generated inside the expanding gas bubble; this shock subsequently implodes and is reflected at the bubble center [32]. Figures 15(a), (b), and (c) show the results obtained by both methods at the time of $t = 0.04$. From the figure we can see that there is fair concurrence of results between the rGFM and the single medium ENO-LF except that the former provides a sharp interface while the latter indicates a rather smooth transition (see Figure 15(a)). One can also note that the velocity and pressure provided by the present rGFM is not so smooth across the interface as shown in Figures 15(b) and (c). Such phenomenon is physical and one should strictly detect in the flow a contact discontinuity in the density plot. In fact, the pressure gradient across a contact discontinuity should satisfy the following equality [33] for an axis-symmetric problem:

$$\frac{1}{\rho^-}\frac{\partial p^-}{\partial n} = \frac{1}{\rho^+}\frac{\partial p^+}{\partial n}, \tag{5.1}$$

where the superscript $+$ and $-$ indicate the flow states on each side of the interface, respectively, and n indicates the normal direction of the interface. Equation (5.1)

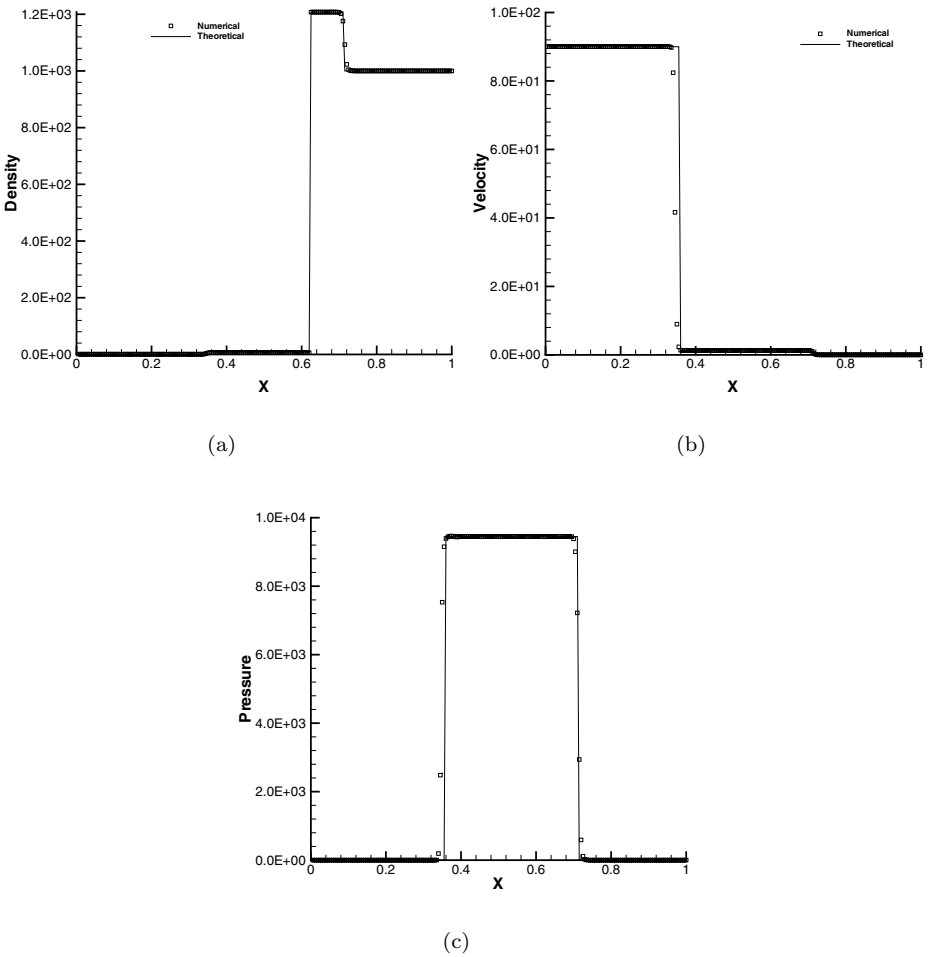


FIG. 13. *Problem 4: Comparison of the MGFM results with the theoretical solution at* $t = 0.015$.

reveals that the pressure gradient is discontinuous across the interface if the pressure gradient is not equal to zero (see Figure 15(c)) due to the density jump across the contact discontinuity. This in turn leads to the jump of the velocity gradient across the interface as shown in Figure 15(b). In Figure 15(c), it is clear that the pressure gradient is not quite equal to zero at the location of the interface.

Problem 6: A two-dimensional double underwater explosion. Two underwater explosions are initialized by two highly pressurized gas bubbles of equal dimensions. The computational domain measures $[0, 4] \times [0, 4]$ with 201×201 nodal points uniformly distributed, and the centers of two bubbles are located at $(1.4, 2)$ and $(2.6, 2)$ with an initial radius of 0.3 each. The flow state inside each bubble is defined as $\gamma_g = 1.4$, $\rho_g = 1000.0$, $p_g = 10^8$, $u_g = 0.0$, $v_g = 0.0$, and $B_g = 0.0$, and the region outside the two bubbles takes on the flow state $\gamma_w = 7.15$, $\rho_w = 1000.0$, $p_w = 10^5$, $u_w = 0.0$, $v_w = 0.0$, and $B_w = 3309.0$. A schematic of the problem is provided in Figure 16. Once the explosions are initiated, two main shock waves are generated and travel radially outward in the water. At time $t = 1.751 \times 10^{-4}$, the two main shock waves intersect and their subsequent interaction leads to two reflected shock waves traveling in the water but in the opposite directions. These two reflected

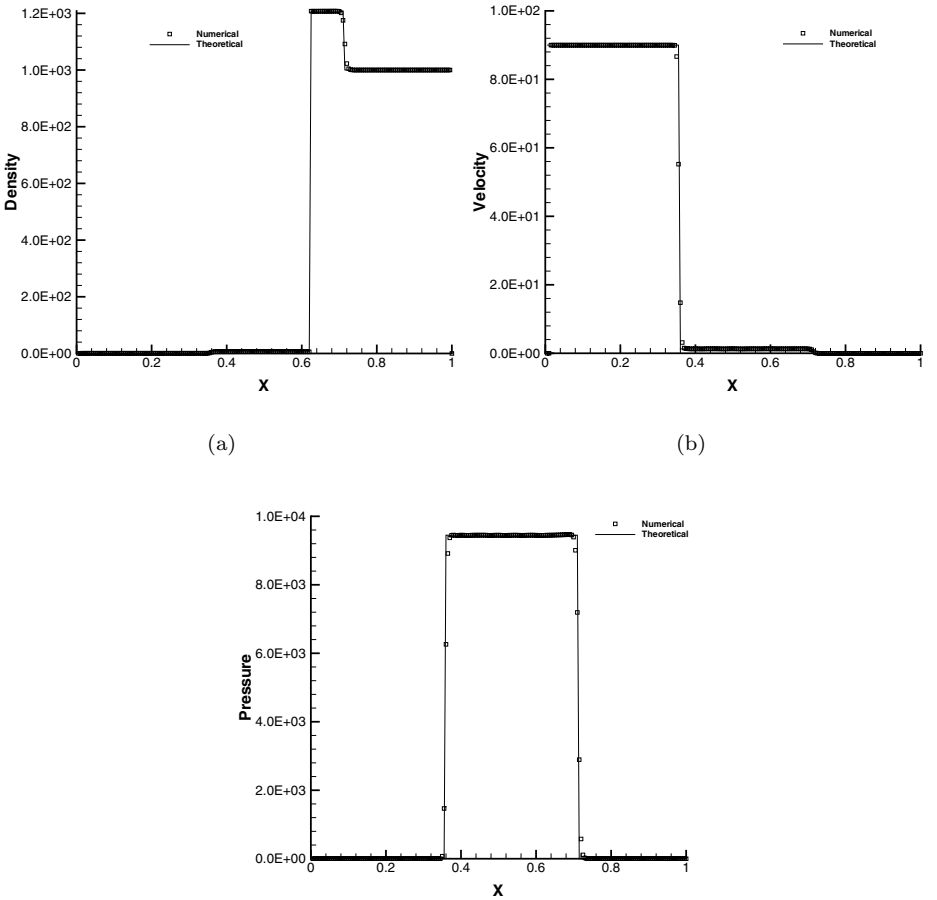


FIG. 14. *Problem 4: Comparison of the rGFM results with the theoretical solution at* $t = 0.015$.

shock waves reach the bubble interface at time $t = 3.450 \times 10^{-4}$. From [34] one may know that the reflection of the shock wave off each bubble surface is a rarefaction wave and the transmission into each gaseous bubble is a shock wave. A time sequence of pressure contours is presented in Figures 17(a), (b), and (c). Figure 17(a) (at $t = 1.172 \times 10^{-4}$) shows the two main shock waves traveling outward in the water before they intersect, and the two shock waves are reflected after their interaction as depicted in Figure 17(b) at $t = 2.888 \times 10^{-4}$. In Figure 17(c) the pressure contours at $t = 4.572 \times 10^{-4}$ indicate two rarefaction waves are reflected back into the water and two shock waves are transmitted inside each bubble due to the shock refraction at the bubble surfaces. Figure 18 is the plot of the level set contours at the same time as for Figure 17(c). One can notice that no contour distortion is found, and all the contour lines are nearly uniformly distributed; this demonstrates the efficiency of the present level set technique.

6. Conclusions. In this work, the rGFM algorithm has been presented. By predicting the flow states at the real fluid nodes next to the interface using the Rie-

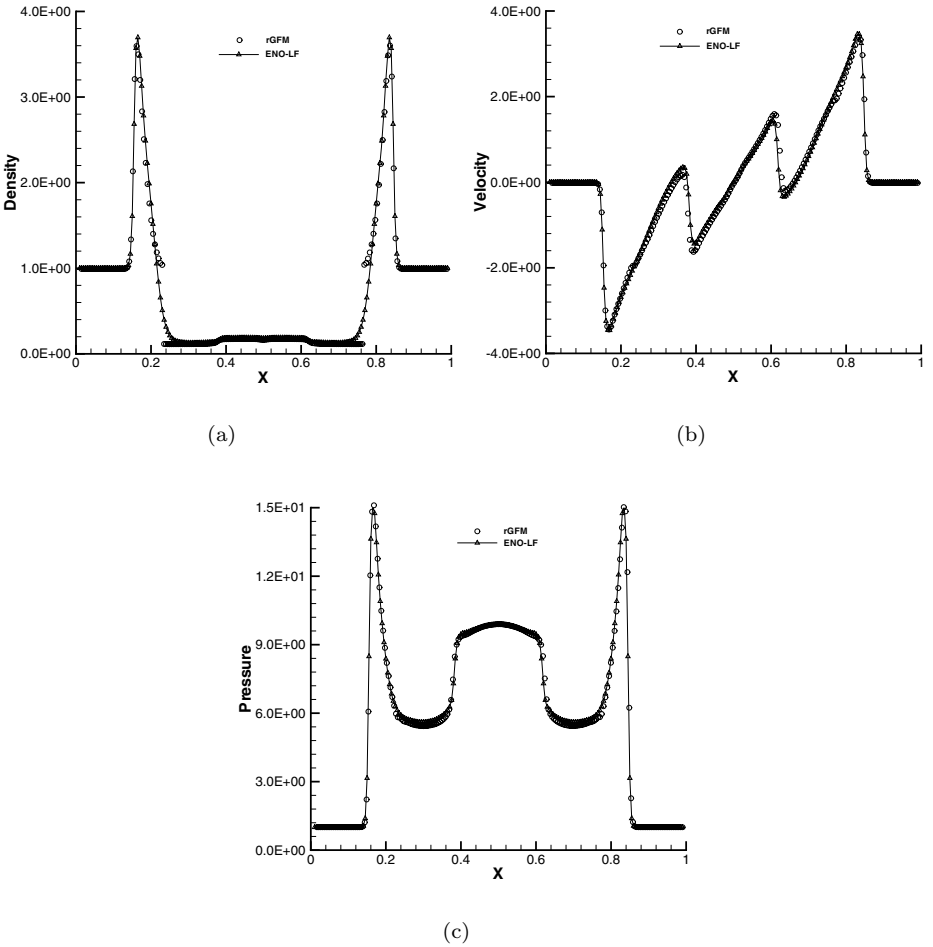


FIG. 15. *Problem 5: Comparison of the rGFM results with the ENO-LF results at* $t = 0.04$.

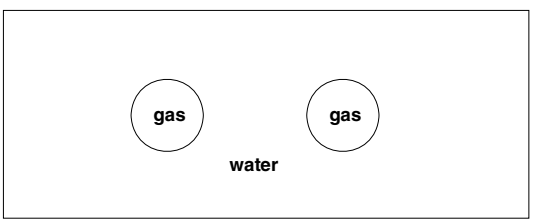
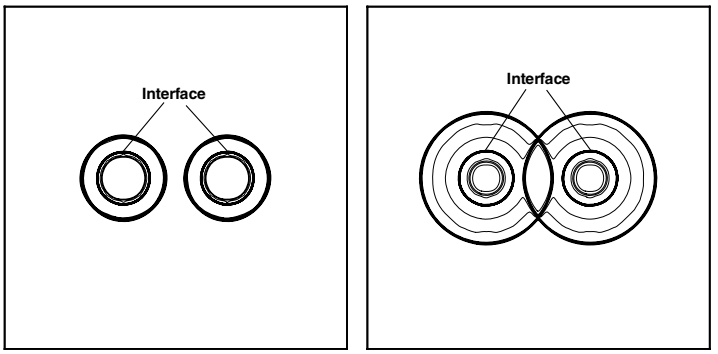
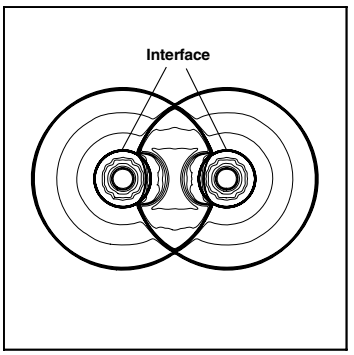


FIG. 16. *Schematics of the computational domain.*

mann problem solver, a more accurate interface boundary condition can be provided for which has been supported by both theoretical analysis and numerical results. A simple yet efficient extension of the present rGFM to multidimensions has also been introduced. In order to overcome the difficulties associated with the severe distortion of the level set contours in the region of the material interface where the large flow velocity gradient may exist, an extension velocity field was constructed to advance



(a) $t = 1.172 \times 10^{-4}$ (b) $t = 2.888 \times 10^{-4}$



(c) $t = 4.572 \times 10^{-4}$

FIG. 17. *Problem 6: Pressure contours at three different times.*

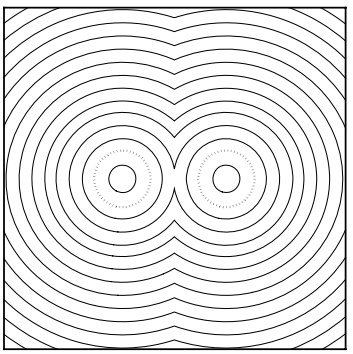


FIG. 18. *Problem 6: Level set contours at* $t = 4.572 \times 10^{-4}$. *The dotted line is the interface.*

the level set function. The present method has been applied to various problems involving strong shock-interface interaction and complex flow physics. The results obtained have clearly indicated that the present method is not problem related and can be employed to various equations of state.

REFERENCES

[1] A. Harten and S. Osher, *Uniformly high-order accurate nonoscillatory schemes* I, SIAM J. Numer. Anal., 24 (1987), pp. 279–309.

[2] A. Harten, B. Engquist, S. Osher, and R. Chakravarthy, *Uniformly high order accurate nonoscillatory schemes* III, J. Comput. Phys., 71 (1987), pp. 231–303.

[3] C. W. Shu and S. Osher, *Efficient implementation of essentially non-oscillatory shock capturing schemes*, J. Comput. Phys., 77 (1988), pp. 439–471.

[4] C. W. Shu and S. Osher, *Efficient implementation of essentially non-oscillatory shock capturing schemes* II, J. Comput. Phys., 83 (1989), pp. 32–78.

[5] C. W. Shu, T. A. Zang, G. Erlebacher, D. Whitaker, and S. Osher, *High order ENO schemes applied to two and three dimensional compressible flow*, Appl. Numer. Math., 9 (1992), pp. 45–71.

[6] B. Koren, M. R. Lewis, E. H. Van Brummelen, and B. Van Leer, *Riemann-problem and level-set approaches for homentropic two-fluid flow computations*, J. Comput. Phys., 181 (2002), pp. 654–674.

[7] E. H. van Brummelen and B. Koren, *A pressure-invariant conservative Godunov-type method for barotropic two-fluid flows*, J. Comput. Phys., 185 (2003), pp. 289–308.

[8] R. P. Fedkiw, T. Aslam, B. Merriman, and S. Osher, *A non-oscillatory Eulerian approach to interfaces in multimaterial flows (the Ghost Fluid Method)*, J. Comput. Phys., 152 (1999), pp. 457–492.

[9] R. P. Fedkiw, *Coupling an Eulerian fluid calculation to a Lagrangian solid calculation with the ghost fluid method*, J. Comput. Phys., 175 (2002), pp. 200–224.

[10] R. Abgrall and S. Karni, *Computations of compressible multifluids*, J. Comput. Phys., 169 (2001), pp. 594–623.

[11] T. D. Aslam, *A level set algorithm for tracking discontinuities in hyperbolic conservation laws* II: *Systems of equations*, J. Sci. Comput., 19 (2003), pp. 37–62.

[12] S. Karni, *Multicomponent flow calculations by a consistent primitive algorithm*, J. Comput. Phys., 112 (1994), pp. 31–43.

[13] R. Abgrall, *How to prevent pressure oscillations in multicomponent flow calculations: A quasi-conservative approach*, J. Comput. Phys., 125 (1996), pp. 150–160.

[14] S. Karni, *Hybrid multifluid algorithms*, SIAM J. Sci. Comput., 17 (1996), pp. 1019–1039.

[15] J. P. Cocchi and R. Saurel, *A Riemann problem based method for the resolution of compressible multimaterial flows*, J. Comput. Phys., 137 (1997), pp. 265–298.

[16] J. Glimm, X. L. Li, Y. Liu, and N. Zhao, *Conservative front tracking and level set algorithms*, PNAS, 98 (2001), pp. 14198–14201.

[17] M. Kang, R. P. Fedkiw, and X. D. Liu, *A boundary condition capturing method for multiphase incompressible flow*, J. Sci. Comput., 15 (2000), pp. 323–360.

[18] X. D. Liu, R. P. Fedkiw, and M. Kang, *A boundary condition capturing method for Poisson's equation on irregular domain*, J. Comput. Phys., 160 (2000), pp. 151–178.

[19] R. Caiden, R. P. Fedkiw, and C. Anderson, *A numerical method for two-phase flow consisting of separate compressible and incompressible regions*, J. Comput. Phys., 166 (2001), pp. 1–27.

[20] R. P. Fedkiw, T. Aslam, and S. Xu, *The ghost fluid method for deflagration and detonation discontinuities*, J. Comput. Phys., 154 (1999), pp. 393–427.

[21] D. Q. Nguyen, R. P. Fedkiw, and M. Kang, *A boundary condition capturing method for incompressible flame discontinuities*, J. Comput. Phys., 172 (2001), pp. 71–98.

[22] T. G. Liu, B. C. Khoo, and K. S. Yeo, *Ghost fluid method for strong shock impacting on material interface*, J. Comput. Phys., 190 (2003), pp. 651–681.

[23] T. G. Liu, B. C. Khoo, and C. W. Wang, *The ghost fluid method for gas-water simulation*, J. Comput. Phys., 204 (2005), pp. 193–221.

[24] D. Adalsteinsson and J. A. Sethian, *A fast level set method for propagating interfaces*, J. Comput. Phys., 118 (1995), pp. 269–277.

[25] J. A. Sethian and J. D. Strain, *Crystal growth and dendritic solidification*, J. Comput. Phys., 98 (1992), pp. 231–253.

[26] T. G. Liu, B. C. Khoo, and K. S. Yeo, *The simulation of compressible multi-medium flow. Part* I: *A new methodology with applications to* 1*D gas-gas and gas-water cases*, Comput. & Fluids, 30 (2001), pp. 291–314.

[27] G. S. Jiang and C. W. Shu, *Efficient implementation of weighted ENO schemes*, J. Comput. Phys., 126 (1996), pp. 202–228.

[28] G. S. Jiang and D. Peng, *Weighted ENO schemes for Hamilton-Jacobi equations*, SIAM J. Sci. Comput., 21 (2000), pp. 2126–2143.

[29] R. Malladi, J. A. Sethian, and B. C. Vemuri, *Shape modeling with front propagation: A level set approach*, IEEE. Trans. Pattern Anal. Mach. Intell., 17 (1995), pp. 158–175.
[30] D. Adalsteinsson and J. A. Sethian, *The fast construction of extension velocities in level set methods*, J. Comput. Phys., 148 (1999), pp. 2–22.
[31] S. F. Davis, *An interface tracking method for hyperbolic systems of conservation laws*, Appl. Numer. Math., 10 (1992), pp. 447–472.
[32] T. G. Liu, B. C. Khoo, and K. S. Yeo, *The numerical simulations of explosion and implosion in air: Use of a modified Harten's TVD scheme*, Internat. J. Numer. Methods Fluids, 31 (1999), pp. 661–680.
[33] R. Grandey, *Application of Finite Difference Methods to Problems in Two-Dimensional Hydrodynamics*, Aeronautic Publication U-1118, California, 1961.
[34] T. G. Liu, B. C. Khoo, and K. S. Yeo, *The simulation of compressible multi-medium flow. Part* II: *Applications to* 2*D underwater shock refraction*, Comput. & Fluids, 30 (2001), pp. 315–337.

SIAM J. SCI. COMPUT.
Vol. 28, No. 1, pp. 303–320
© 2006 Society for Industrial and Applied Mathematics

THE LINEAR PROCESS DEFERMENT ALGORITHM: A NEW TECHNIQUE FOR SOLVING POPULATION BALANCE EQUATIONS*

R. I. A. PATTERSON†, J. SINGH†, M. BALTHASAR†, M. KRAFT‡, AND J. R. NORRIS§

Key words. coagulation, population balances, Monte Carlo, acceleration

AMS subject classifications. 65C05, 65C35, 68U20, 82C22

DOI. 10.1137/040618953

Abstract. In this paper a new stochastic algorithm for the solution of population balance equations is presented. The population balance equations have the form of extended Smoluchowski equations which include linear and source terms. The new algorithm, called the linear process deferment algorithm (LPDA), is used for solving a detailed model describing the formation of soot in premixed laminar flames. A measure theoretic formulation of a stochastic jump process is developed and the corresponding generator presented. The numerical properties of the algorithm are studied in detail and compared to the direct simulation algorithm and various splitting techniques. LPDA is designed for all kinds of population balance problems where nonlinear processes cannot be neglected but are dominated in rate by linear ones. In those cases the LPDA is seen to reduce run times for a population balance simulation by a factor of up to 1000 with a negligible loss of accuracy.

1. Introduction.

The study of population balance problems for collections of particles that interact with each other via collisions has a long history—Bird [3] used this approach for gas dynamics in the 1970s. More recent applications include atmospheric aerosols (see, e.g., [4] and the references given there) and particle synthesis in flames [17].

In these later cases interactions between the particles and the surrounding gas which operate linearly on the particle population have to be considered in addition to particle-particle interactions which have nonlinear effects. Work by some of the authors of this present paper on the formation of soot in laminar premixed flames [2, 14] showed that a large amount of computer time was being devoted to the linear processes of particle-gas interactions.

The purpose of this paper is to present a new algorithm that we call the linear process deferment algorithm (LPDA) and which reduces the time required for simulations. We also compare this new method to the general numerical analysis technique of splitting. LPDA is applicable to population models of systems other than soot where nonlinear processes are important but dominated in rate by linear ones.

In this paper we first set out the population balance problem for soot formation and growth that we seek to solve. Then we give the structure of our original simulation procedures, the direct simulation algorithm (DSA). Detailed measurements of the

*Received by the editors November 15, 2004; accepted for publication (in revised form) September 20, 2005; published electronically March 10, 2006.
http://www.siam.org/journals/sisc/28-1/61895.html

†Department of Chemical Engineering, University of Cambridge, UK. The third author is now with Volvo, Göteborg The second author was supported by Churchill College. The third author was supported by the Marie Curie program under FP6-2002-Mobility-5.

‡Corresponding author. Department of Chemical Engineering, University of Cambridge, Pembroke Street, Cambridge, CB2 3RA, UK (mk306@cam.ac.uk). This author received support from EPSRC grant GR/R85662/01.

§Statistical Laboratory, University of Cambridge, Centre for Mathematical Sciences, Wilberforce Road, Cambridge, CB3 0WB, UK (j.r.norris@atatslab.cam.ac.uk). This author received support from EPSRC grant GR/R85662/01.

computer time used in different parts of this method motivate the introduction of operator splitting. The splitting provided accelerations and a framework to investigate methods for simulating linear processes which are then reused in our new algorithm. We present the LPDA and its stochastic basis and conclude with some comparisons between the algorithms considered in the paper.

2. Outline of general soot model. We assume

1. Individual soot particles may be completely described by elements of some type space E on which addition corresponding to coagulation is defined.

2. The soot population is described at time t by the number $n(t,x)$ per cm^3 of particles of type $x \in E$.

3. The function n evolves according to the discrete Smoluchowski coagulation equation:

$$\frac{\mathrm{d}}{\mathrm{d}t} n(t,x) = \left(\mathcal{K}_t(x) + \sum_{l \in \mathcal{U}} \mathcal{S}_t^{(l)}(x) \right) (n(t,\cdot)) + I(t,x). \tag{2.1}$$

In these definitions we make the implicit assumption that E is countable so that summations are meaningful. This assumption is common in the literature but not essential and later in this paper it will be removed by replacing the sums with integrals. The (time-dependent and nonlinear) coagulation operator $\mathcal{K}$ is then defined by

$$\begin{aligned} \mathcal{K}_t(x)(n(t,\cdot)) = {} & \frac{1}{2} \sum_{y,z \in E : y+z=x} K_t(y,z)\, n(t,y)\, n(t,z) \\ & - \sum_{y \in E} K_t(x,y)\, n(t,x)\, n(t,y). \end{aligned} \tag{2.2}$$

The first sum represents coagulation to form particles of type x and the second loss of particles of type x due to coagulation. $K_t(x,y)$ defines a map from the concentrations of particles of types x and y to their coagulation rate at time t given by $K_t(x,y)\, n(t,x)\, n(t,y)$. K is known as the coagulation kernel and we use the transition kernel given in [14].

Surface reactions which only involve one physical particle at a time are described by the linear operators $\mathcal{S}^{(l)}$ defined by

$$\mathcal{S}_t^{(l)}(x)(n(t,\cdot)) = \left(\sum_{y \in E} \beta_t^{(l)}(y)\, \mathbb{P}\left(g^{(l)}(y) = x \right) n(t,y) \right) - \beta_t^{(l)}(x)\, n(t,x) \tag{2.3}$$

for $l \in \mathcal{U}$, an index set for the surface reactions, according to our flame model we use, (see [1, 14]),

$$\mathcal{U} = \{ \mathrm{C_2H_2} \text{ addition}, \mathrm{OH} \text{ oxidation}, \mathrm{O_2} \text{ oxidation}, \text{pyrene condensation} \}.$$

$\beta_t^{(l)}(x)$ is the rate at which a particle of type x undergoes surface reaction l at time t.

$g^{(l)}(x)$ is the type of the result of a particle of type x undergoing surface reaction l or if the reaction removed the particle from the soot population the value will be $0 \in E$. We allow for $g^{(l)}$ to be a random function; in the deterministic case the probability in (2.3) is to be understood as an indicator function.

$I(t,x)$ is the rate at which particles of type x enter the system at time t; throughout our work we have assumed this is zero except when x describes a spherical particle of 32 carbon atoms (C atoms).

The gas phase chemistry (see [1]) needed to calculate I and the $\beta^{(l)}$ etc. is simulated separately with a method of moments approximation [7].

No boundary conditions are specified at this stage although in general we will use $n(0,x) = 0$ for all $x \in E$ since a small element of flow has no soot in it when it enters the flame.

We assume all particles to be spheres, all of the same density, so we take $E \subseteq \mathbb{R}_0^+$ and let x be the mass or the volume of the particle it describes.[1] It is then convenient to take $E = \mathbb{N}$, where $x \in E$ is the number of C atoms in a particle it describes. Addition is defined just as for the natural numbers: the result of the coagulation of two spheres is a new sphere with volume equal to the sum of the volumes of the initial spheres, i.e., coagulation is completely coalescent.

Different coagulation kernels are appropriate in different conditions; these are also used to model the pyrene condensation and inception processes. For details see [14]. All the work in this paper was carried out using the transition regime kernel. We have not found the use of more specific kernels to affect the results.

3. Direct simulation. The DSA [5] is the basic stochastic approach to solving equations like (2.1) and forms the starting point of our work. Our implementation is described in [8, 14]; however, we use a binary tree method for particle selection as this enables the use of more complex soot models. Various similar methods are discussed in [11], along with a detailed performance analysis. An essential assumption underlying much of the work in later sections of this paper is that DSA can be used to generate solutions of arbitrary precision for (2.1). Excellent reasons for accepting this assumption are provided by the mathematical proofs of a simplified version of it presented in [5, 6] and in the numerical tests reported in [13, 14].

3.1. Variation in number of particles. In a direct simulation on a computer of a chemical system where the number of particles may increase and decrease one has to limit the number of particles being tracked in the simulation. We do this by setting a maximum number of particles and when we wish to add a particle that would cause us to exceed this limit we rescale the volume for which soot formation is being simulated to reduce the number of particles by 1, the particle to exclude from the simulation is chosen uniformly at random from the entire ensemble. As coagulation or oxidation reduces the number of particles being simulated the accuracy of the simulation is also reduced. Therefore when the number of computational particles in the simulation drops below one half the maximum number we double the sample volume and duplicate each particle that is being simulated.

An introduction to this kind of issue is given in [15]. We tried a form of constant number algorithm by extending the method described above for use every time particles were removed from the simulation volume but found that for simulations of some artificial test cases this led to a much greater statistical uncertainty (around 1 order of magnitude in the first few moments of the distribution) in the results compared to using particle doubling. This is consistent with the results in [12]; therefore we use only particle doubling in our simulations.

3.2. Complexity of our implementation. For a binary tree with l levels (capacity 2^l particles) each event requires $O(l)$ operations. Selecting a particle involves reading one node on each level of the tree. Updating the tree when a particle has changed requires updating one node on each level of the tree (following the path to the root of the tree).

[1] The preceding formulation assumes the countability of E; this requirement is removed later.

TABLE 1
Illustrative results.

flame	tree depth	repetitions	m2 uncertainty	time / min
JW1.69	13	10	2.3%	50
JW1.69	10	30	6.8%	16.5
JW10.68	11	20	1.5%	4020
JW10.68	10	20	2.9%	1534

TABLE 2
Frequency of stochastic events.

	JW1.69	JW10.68
time steps	1.25×10^9	2.06×10^9
surface events (not pyrene)	1.20×10^9	2.05×10^9
pyrene condensation	4×10^7	1×10^7
coagulation	5×10^6	4×10^5
inception	4×10^6	2×10^5

The number of events scales linearly with the sample volume, V, as does the number of particles. Hence for reasonable choices of V and l (those for which the full capacity of the tree is used) a simulation will need a tree with $O(\log V)$ levels and will have a run time that is $O(V \log V)$.

3.3. DSA applied to flames. Most of the work in this paper focuses on simulating two premixed laminar flames studied in [10]—JW1.69 and JW10.68. The first is a 1-bar flame with equivalence ratio 0.69 and a peak soot volume fraction of approximately 2×10^{-8}, the second a 10-bar flame with equivalence ratio 0.68 and a peak soot volume fraction of approximately 2×10^{-6}. These soot volume fractions are at the extremes of the range studied in [10].

We performed simulations using different numbers of computational particles and varying numbers of repetitions getting some idea of systematic error in moments 0 to 3 of the final size distribution due to using a low number of computational particles (this error was generally small) and the number of repetitions needed to control the statistical noise. Some initial results follow in Table 1. "m2" refers to the second moment of the mass distribution, and the uncertainty is the half width, using a central limit theorem estimate, of the 95% confidence interval for the mean of the distribution being sampled (that of m2).

The run times for JW10.68 indicate that DSA as we have implemented it is not a very practical tool for investigating sooty flames on a desktop computer. In order to inform our attempts to reduce the run times we used the standard technique of profiling to find out what parts of the program consumed most of the execution time.

3.3.1. Profiling of DSA simulation. Table 2 shows the number of each type of event performed during 10 runs of a simulation of the premixed flame JW1.69 and 20 runs of JW10.68. JW1.69 was simulated with 16 tree levels and an initial sample volume of 2.8×10^{-7} cm^3, JW10.68 with a tree depth of 10 and an initial sample volume of 1.25×10^{-9} cm^3. Note that twice as many events are shown for the JW10.68 case as for the JW1.69 case although the sample volume is 200 times smaller. Pyrene condensation is reported separately from the other surface events because the physical model for the process is significantly different from that for the other surface reactions in $\mathcal{U}$ and the computational implementation is therefore different as well.

The main conclusion to be drawn from Table 2 is that essentially all the events are surface events and that there are not many pyrene events compared to the other

events. One may also note that coagulation events are even more rare than pyrene events.

To check exactly which parts of our program took longest to execute we used a profiler to measure the time spent in each function. The data were obtained over five runs simulating JW1.69 with a tree depth of 12 and an initial sample volume of 1.75×10^{-8} cm^3. We found that 64% of the execution time was consumed in functions that calculated the random time steps. These parts of the program are key to the simulation of a nonlinear coagulation process but are not fundamental for the surface processes since the surface growth for a single particle is simply a nonhomogeneous Poisson process until the time that particle is involved in a coagulation and which, conditional on the coagulation time, does not depend on the other particles. However, most of these time steps are simply concerned with the surface processes and not coagulation. A considerable amount of the 31% of time spent in functions that updated our computational particle ensemble to reflect the simulated surface events was also attributable to the binary tree structure necessary for simulating coagulation.

We profiled simulations of JW1.69 rather than JW10.68 because profiling leads to a very large increase in run times and so profiling JW10.68 was not feasible given the large run times for the basic simulations reported in Table 1.

3.3.2. Conclusions from DSA profiling. We note that

1. Almost all the computational time is spent simulating surface processes.
2. These surface processes are simulated in a framework designed for a different purpose.
3. In JW10.68 the preponderance of surface events is even greater (see Table 2) than for JW1.69, so an even greater proportion of time will be spent simulating surface processes.

We therefore looked at alternative ways of incorporating the surface processes into our simulations in the hope of reducing run times.

4. Operator splitting. In (2.1) we see the evolution of the particle distribution is driven by three kinds of process:

1. Coagulation—two particle events;
2. Single particle events,
   (a) Surface area processes,
   (b) Pyrene condensation;
3. Particle inception—involves no preexisting particles.

We know from the analysis above that most of the time taken in a simulation is spent updating the binary tree after events under 2(a). Operator splitting ([9] gives far more material than we use here) is an approach to numerically solving differential equations that allows us to introduce different methods to solve for the effects of single particle events.

4.1. Mathematical outline of splitting. Consider, for some operators $\mathcal{A}$ and $\mathcal{B}$, the ODE which we assume to have a unique, well-behaved solution

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t} f &= (\mathcal{A} + \mathcal{B})(f), \\ f(0) &= f_0. \end{aligned} \tag{4.1}$$

The easiest way to numerically compute the solution over time is based on the idea that for small h

$$f(t+h) - f(t) \approx h(\mathcal{A} + \mathcal{B})(f)(t).$$

This is the basis of DSA where we simulate all the different processes simultaneously—the jump rate of every process depends on the current state of the particle ensemble.

However, at the cost of introducing a further approximation and by assuming that the resulting ODEs have well-behaved solutions, one can define auxiliary functions f_1, f_2 and advance the solution over a small time interval $[t_0, t_0 + \delta t]$ by solving with whatever numerical schemes one considers appropriate first,

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t} f_1 =& \mathcal{A}(f_1), \\ f_1(t_0) =& f(t_0), \end{aligned} \tag{4.2}$$

and then

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t} f_2 =& \mathcal{B}(f_2), \\ f_2(t_0) =& f_1(t_0 + \delta t). \end{aligned} \tag{4.3}$$

The key point is that completely different numerical methods may be used to solve (4.2) and (4.3).

4.2. Implementation of operator splitting. We split (2.1) into two parts by handling OH oxidation and C_2H_2 addition (sometimes also the insignificant process of O_2 oxidation) separately from all the other processes which are simulated in exactly the same way as in the DSA. We typically performed this splitting over time intervals of a length such that each particle would, on average, undergo r surface reactions during a splitting step where r was a positive but not necessarily integer valued parameter. The choice of r was the way we determined the length of the splitting step and so the accuracy and speed of the approximation.

The steps in advancing a simulation over times t from t_0 to t_1 are (t_2 and t_3 are auxiliary times):

1. $t \leftarrow t_0,\ t_2 \leftarrow t_0$
2. WHILE($t < t_1$)
   (a) Choose a step length $\triangle t$ and set $t_2 = t + \triangle t$. We choose $\triangle t$ as the time in which $n \times r$ events due to the split processes would be expected under DSA. n is the number of computational particles. Various conditions may lead to large values of $\triangle t$ being calculated—these must be reduced.
   (b) Perform the unsplit processes including coagulation from t to t_2 exactly as for DSA using the binary tree.
   (c) Extract a list of particles from the binary tree.
   (d) WHILE($t < t_2$)
      i. Choose a secondary time step length δt such that the chemical conditions vary only slightly over the time interval $[t, t + \delta t]$ and $t + \delta t \leqslant t_2$.
      ii. $t_3 \leftarrow t + \delta t$
      iii. Simulate the split processes up to t_3 for each particle. Methods for doing this are discussed below.
      iv. $t \leftarrow t_3$
      v. Return to top of the loop—2(d)
   (e) Put the updated computational particles back into the binary tree. (We should now have $t = t_2$)
   (f) Return to the top of the loop—2
3. Stop

4.2.1. Simulating split events for one particle. Exactly the same method as is used in DSA was applied to individual computational particles. We generated the inhomogeneous Poisson process referred to in section 3.3.1 by repeatedly calculating the exponentially distributed waiting times and performing events at their end points. We refer to this method as PP in later sections.

In order to further reduce the computational effort we introduced a second level of splitting. By dealing with the split processes one by one and making a further approximation we only have to generate one random variable per process per particle as follows:

1. Calculate the rate for the selected process at the start of the interval.
2. Assume this rate applies throughout the interval (e.g., ignoring the effects of the particle size changing) and so calculate the expected number of events λ.
3. Generate a random variable with the Poisson (λ) distribution as the number of events which actually occur and update the computational particle accordingly.

We refer to this method as PV (Poisson variable) in later sections.

4.2.2. Complexity. We perform the split surface processes without using the binary tree; this avoids the $O(\log n)$ update operations (n is the number of stochastic particles which should be roughly the same as the capacity of the binary tree) after each event that were slowing the simulations down. Instead, after performing surface reactions, an $O(1)$ operation, on each of the $n \sim O(V)$ particles individually for the full length of the splitting time step, we then completely rebuild the binary tree, which takes $O(n) \sim O(V)$ time.

Therefore the surface processes which previously had overall complexity $O(V \log V)$ in the sample volume V now have only complexity $O(V)$; contrast section 3.2. Since the surface processes dominate the simulation (see Table 2), this should reduce the apparent complexity of the simulation.

4.3. Numerical results.

4.3.1. Comparison of JW1.69 moments. We present some results for JW1.69 (one of the laminar premixed flames studied in [10]) generated using our implementations of operator splitting and compare to DSA results generated as described above. We used JW1.69 because it provides a realistic test case that can be simulated quickly and also for historic reasons.

Table 3 gives a summary of the simulations performed and the computer time they took. All timed simulations were performed using a binary tree of depth 13 and with an initial sample volume of $3.5 \times 10^{-8}\text{cm}^{-3}$. The simulations reported in this paper were generated using PCs with either 2.4 GHz Pentium 4 or 2 GHz Athlon CPUs. Both configurations were found to give almost identical performance on test cases.

TABLE 3
Detail of operator splitting runs for JW1.69.

algorithm	r	time for 10 repetitions / min
DSA	0	50
PV	1	30
PV	2	16
PV	5	8
PV	10	5.5
PP	5	12.5
PP	10	10

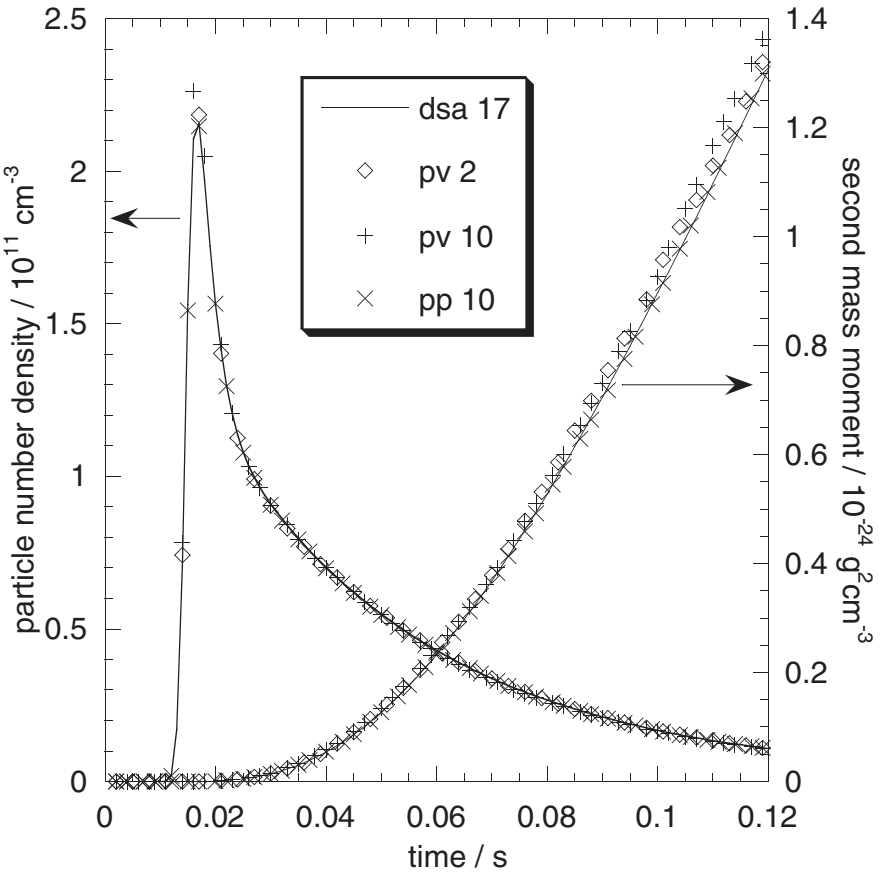


FIG. 1. *Operator splitting for* JW1.69.

With the PV algorithm we see a considerable speed increase as we move from $r = 0$ (DSA) to $r = 5$ with speed being almost linear in r but changing r from 5 to 10 makes very less difference to the execution time. We expect the bottleneck to be the inner loop of the algorithm set out in section 4.2. The PP splitting leads to longer run times than the PV because of the way each surface event is handled individually.

However, speed is not everything—one also has to consider the error introduced by the splitting approximation, so in Figure 1 we show the time evolution of the zeroth and second moments of the particle size distribution in a small volume element passing through the flame. To show the error introduced by the splitting we also plot results obtained using DSA with 30 repetitions, a binary tree depth of 17, and an initial sample volume of $5.6 \times 10^{-7}\text{cm}^{-3}$. We believe these results are indistinguishable from the true solution on the scale shown.

We see that the splitting only introduces a very modest error in the second moment of the distribution, at least for $r \leqslant 5$, compared to the high precision DSA. For the first moment the error will be smaller.

4.3.2. Particle distribution accuracy. One of the advantages of stochastic simulations is that one gets a complete description of the particle size distribution. We present details of this for the flame JW10.68 (see [10]). JW10.68 is a much sootier flame than JW1.69 and so provides a test in a situation where soot is significant. As

TABLE 4
Detail of operator splitting runs for JW10.68.

algorithm	r	time for 10 repetitions / min
DSA	0	2010
PV	5	134
PV	10	86
PP	20	250
PP	50	235

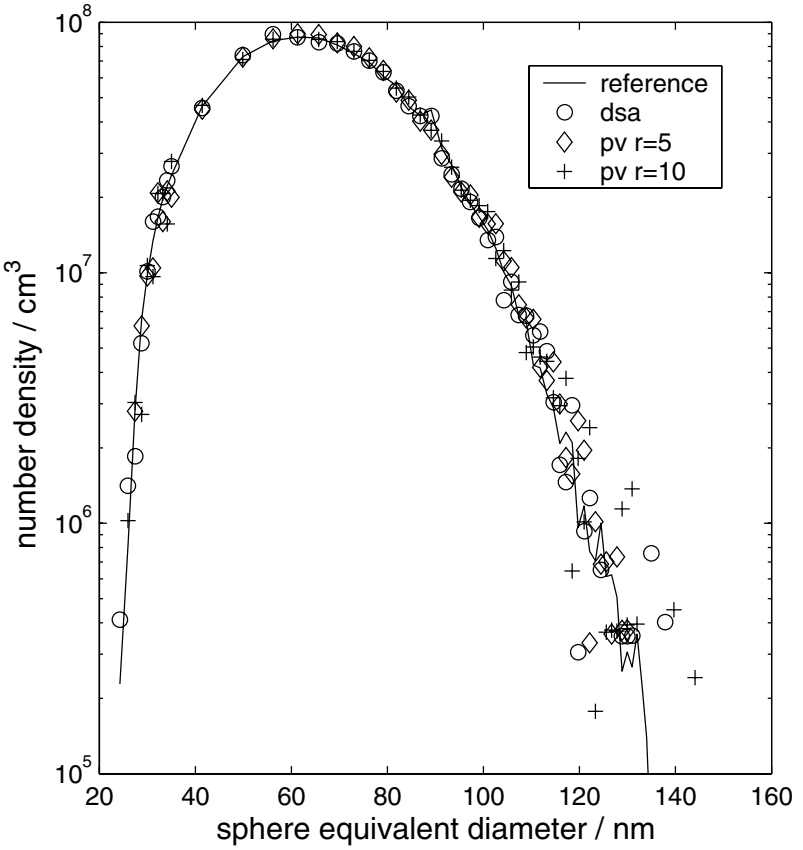


FIG. 2. *Final particle size distribution for* JW10.68.

can be seen, it is also far more computationally demanding. Table 4 gives details of the simulations we performed using a binary tree depth of 11 and an initial sample volume of $2.5 \times 10^{-9}\text{cm}^{-3}$ on the PCs described above.

The acceleration due to splitting is much larger for this flame than for JW1.69 (see Table 3)—a factor of 30 compared to one of 6 for DSA to PV $r = 5$. Further increases in r were found to offer much smaller speed gains with the PV algorithm while the PP algorithm times appear to have only a weak dependence on r, suggesting the time limiting step is the event-by-event simulation of the split processes.

Figure 2 shows the particle distributions obtained from the first three simulations described in Table 4. The distribution calculated from a set of 10 DSA runs using a binary tree depth of 14 is also shown as a reference curve; the simulations to generate this took 13 days of PC time! It can be seen that the factor of 8 ($= 2^{14-11}$) increase

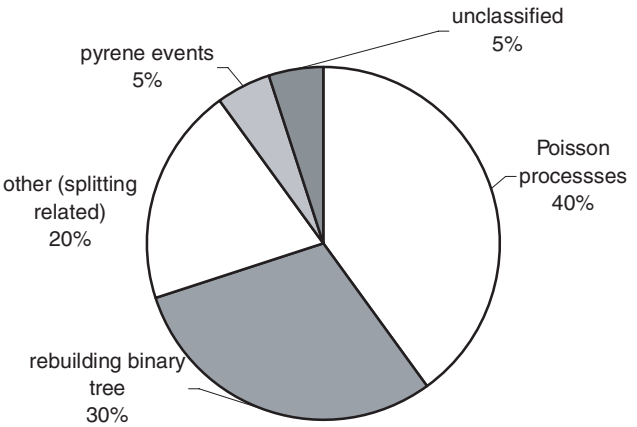


FIG. 3. *% of execution time spent on different tasks—op split.*

in the number of computational particles and hence the resolution of the reference solution simulations compared to those for the other results leads to less scatter in the distribution at large particle sizes in Figure 2. This is because the number of stochastic particles representing particles in a particular physical size range is proportional to the number density multiplied by the number of computational particles. If Figure 2 were extended in the bottom right-hand corner into the size range where no particles are predicted by the simulations using only 11 tree levels, greater scatter would be seen in the reference solution until it too dropped to 0.

4.4. Profiling of operator splitting. Operator splitting yields a considerable acceleration from DSA—see Table 4. The computational times are still substantial and would still be noticeable even with 1000 stochastic particles (tree depth of 10). Therefore it makes sense to repeat the analysis of section 3.3.1 and see if further improvements can be made. Figure 3 shows how the computational time is used during a PV algorithm run with the parameters used for the DSA profiling in section 3.3.1. "Poisson processes" refers to the approximation of the surface event Poisson processes by Poisson random variables for each particle during the splitting steps.

From this data we see that simulating the surface reactions is still by far the most computationally intensive task in the simulation—for JW1.69 a flame with relatively little surface growth compared to a sooty flame such as JW10.68. We also note that a substantial amount of time is spent approximating the Poisson processes that represent the surface reactions during splitting. We therefore try a further approximation in the next section.

4.5. Conclusions on operator splitting. Operator splitting using the PP and PV methods provides a substantial performance gain from DSA, in the most striking case reducing a run time by a factor of almost 10. The approximations made to achieve this acceleration have a small effect on the results; this error does not depend very strongly on the size of the time splitting steps. Compared to the uncertainties in the simulation of the gas phase chemistry (see, for example, [16]), the systematic error should not be a cause for concern.

5. Deferment of surface reactions. A new approach is to tag each stochastic particle with the time when it is added to the ensemble so particles are described by elements of the space $\tilde{E} = E \times [0, \infty)$. Some surface reactions (possibly now including

pyrene condensation) can be selected for "deferment"; such reactions are completely neglected as the simulation calculates time steps. Once a particle has been selected for an event the deferred events are simulated from the time recorded in the tag up to the current time. Only then does the nondeferred event take place. The tag is then reset to the current time so that the procedure can be repeated.

5.1. Measure theoretic formulation. To work toward the derivation of the generator for a stochastic jump process implementing this idea it is helpful to work in terms of measures, so we replace

$$n : [0, \infty) \times \tilde{E} \to [0, \infty)$$

by a family of measures giving the particle distribution on $\tilde{E}$ at each time.

To do this we endow $\tilde{E}$ with a σ-algebra $\mathcal{E}$ which makes the $g^{(l)}$, the $\beta_t^{(l)}$, K_t, $+$ and $I(t, \cdot)$ as defined below measurable. The $\mathbb{R}$ valued functions K_t and $\beta_t^{(l)}$ are extended to $\tilde{E}$ by simply ignoring the time stamp component, $g^{(l)}$ and $+$ (strictly $g_t^{(l)}$ and $+_t$) by defining the time stamp of the result to be the current time.

Let M be the space of measures on $\tilde{E}$ which we give the weak topology and the associated Borel σ-algebra $\mathcal{M}$. We describe the particle distribution by a function from time to M

$$\lambda : [0, \infty) \to M; \quad t \mapsto \lambda_t = \lambda(t, \cdot).$$

Thus for $A \in \mathcal{E}$ the number of particles per cm^3 described by elements of A at time t will be $\lambda_t(A)$.

The particle inflow rate will be given by a similar function,

$$I : [0, \infty) \to M; \quad t \mapsto I_t = I(t, \cdot).$$

The rate of inflow of particles with descriptions in A at time t is $I_t(A)$.

(2.1) can now be written for $A \in \tilde{E}$ as

(5.1)
$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\lambda_t(A) = &\int_{x\in A} I_t(dx)\\ &+ \sum_{l\in\mathcal{U}}\left[\int_{x\in\tilde{E}:g^{(l)}(x)\in A}\beta_t^{(l)}(x)\,\lambda_t(dx) - \int_{x\in A}\beta_t^{(l)}(x)\,\lambda_t(dx)\right]\\ &+ \frac{1}{2}\int_{y,z\in\tilde{E}:y+z\in A} K_t(y,z)\,\lambda_t(dy)\,\lambda_t(dz) - \int_{x\in A,y\in\tilde{E}} K_t(x,y)\,\lambda_t(dx)\,\lambda_t(dy).\end{aligned}$$

Hence for ϕ in a suitable class of test functions $\mathscr{C}$ we have a weak form (see, e.g., [6]),

(5.2)
$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\int_{x\in\tilde{E}}\phi(x)\,\lambda_t(dx) = &\int_{x\in\tilde{E}}\phi(x)\,I_t(dx)\\ &+ \sum_{l\in\mathcal{U}}\int_{x\in\tilde{E}}\beta_t^{(l)}(x)\left[\phi\left(g^{(l)}(x)\right) - \phi(x)\right]\lambda_t(dx)\\ &+ \frac{1}{2}\int_{x,y\in\tilde{E}}\left[\phi(x+y) - \phi(x) - \phi(y)\right]K_t(x,y)\,\lambda_t(dx)\,\lambda_t(dy).\end{aligned}$$

This includes the time stamp without really making use of it in λ, our representation of the particle population.

5.1.1. Deferred surface process operator. We now define a family of mappings $R_t : t \in [0, \infty)$ which map $x \in \tilde{E}$ to a random variable $R_t x \in E \times \{t\} \subseteq \tilde{E}$ with the distribution that a particle with description x would have at time t in the absence of deferrment. This only makes sense if t is later than the time stamp in x.

We also define an update operator

$$\mathcal{P}_t : M \to M$$

by

$$\mathcal{P}_t(\lambda)(A) = \int_{x \in \tilde{E}} \mathbb{P}(R_t x \in A)\, \lambda(dx). \tag{5.3}$$

Let $\mathcal{U}' \subseteq \mathcal{U}$ be the possibly empty set of surface reactions that are not deferred. We now want to define a "deferred solution" $\bar{\lambda}$ via an equation like (5.2) such that for any $\phi \in \mathscr{C}$

$$\Phi(\lambda_t) = \Phi\left(\mathcal{P}_t\left(\bar{\lambda}_t\right)\right), \tag{5.4}$$

where

$$\Phi(\nu) := \int_{x \in \tilde{E}} \phi(x)\, \nu(dx) \tag{5.5}$$

and λ solves (5.2).

A fairly obvious approach is

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t} \int_{x \in \tilde{E}} \phi(x)\, \bar{\lambda}_t(dx) &= \int_{x \in \tilde{E}} \phi(x)\, I_t(dx) \\
&+ \sum_{l \in \mathcal{U}'} \int_{x, \xi \in \tilde{E}} \left[\phi\left(g^{(l)}(\xi)\right) - \phi(x)\right] \beta_t^{(l)}(\xi)\, \mathbb{P}(R_t x \in d\xi)\, \bar{\lambda}_t(dx) \\
&+ \frac{1}{2} \int_{x,y,\xi,\zeta \in \tilde{E}} \left[\phi(\xi + \zeta) - \phi(x) - \phi(y)\right] K_t(\xi, \zeta) \\
&\qquad \mathbb{P}(R_t x \in d\xi)\, \mathbb{P}(R_t y \in d\zeta)\, \bar{\lambda}_t(dx)\, \bar{\lambda}_t(dy).
\end{aligned} \tag{5.6}$$

This defines a (or possibly some, since uniqueness is hard to establish) deterministic map from time to the soot population. However, while the resulting $\bar{\lambda}$ may satisfy (5.4) there are real problems using it as a basis for a simulation. First, we have the usual need to incorporate a majorant coagulation kernel (see, e.g., [8]). This then raises questions about the deferred processes for particles involved in fictitious jumps. Second, even when we have a majorant we still cannot handle individually the rates of the infinity of possible jumps given by R_t.

We developed an equation which allows for the use of deferred events generated for computational particles involved in events that prove to be fictitious and avoids considering individual values of $R_t x$. The basic idea is that if we can find $\hat{\beta}$ and $\hat{K}$ such that

$$\hat{\beta}_t^{(l)}\left(x; \bar{\lambda}_t\right) \geqslant \beta_t^{(l)}(R_t x) \tag{5.7a}$$

and

$$\hat{K}_t\left(x, y; \bar{\lambda}_t\right) \geqslant K_t(R_t x, R_t y) \tag{5.7b}$$

hold with probability 1, then it is sufficient to calculate rates for a particle using a majorant without knowing anything about deferred events. The deferred events can then be incorporated when another event is actually performed. In addition, if a particle is selected for a potential event which eventually turns out to be fictitious, postponed processes can be simulated on it anyway and the updated particle returned to the computational ensemble, thus reducing the deferment error.

Given the random nature of $R_t x$ it is likely to be hard to find $\hat{\beta}$ and $\hat{K}$ to satisfy (5.7) with probability 1 so we actually formulate our equation[2]

$$
\begin{aligned}
(5.8)\quad & \frac{\mathrm{d}}{\mathrm{d}t}\int_{x\in\tilde{E}}\phi(x)\,\bar{\lambda}_t(dx)=\int_{x\in\tilde{E}}\phi(x)\,I_t(dx)\\
&+\sum_{l\in U'}\int_{x,\xi\in\tilde{E}}\left[\phi\left(g^{(l)}(\xi)\right)-\phi(x)\right]\left[\beta_t^{(l)}(\xi)\wedge\hat{\beta}_t^{(l)}\left(x;\bar{\lambda}_t\right)\right]\mathbb{P}(R_t x\in d\xi)\,\bar{\lambda}_t(dx)\\
&\quad+\sum_{l\in U'}\int_{x,\xi\in\tilde{E}}[\phi(\xi)-\phi(x)]\left[\hat{\beta}_t^{(l)}\left(x;\bar{\lambda}_t\right)-\beta_t^{(l)}(\xi)\right]_+\mathbb{P}(R_t x\in d\xi)\,\bar{\lambda}_t(dx)\\
&\quad+\frac{1}{2}\int_{x,y,\xi,\zeta\in\tilde{E}}[\phi(\xi+\zeta)-\phi(x)-\phi(y)]\left[K_t(\xi,\zeta)\wedge\hat{K}_t\left(x,y;\bar{\lambda}_t\right)\right]\\
&\qquad\qquad\mathbb{P}(R_t x\in d\xi)\,\mathbb{P}(R_t y\in d\zeta)\,\bar{\lambda}_t(dx)\,\bar{\lambda}_t(dy)\\
&\quad+\frac{1}{2}\int_{x,y,\xi,\zeta\in\tilde{E}}[\phi(\xi)+\phi(\zeta)-\phi(x)-\phi(y)]\left[\hat{K}_t\left(x,y;\bar{\lambda}_t\right)-K_t(\xi,\zeta)\right]_+\\
&\qquad\qquad\mathbb{P}(R_t x\in d\xi)\,\mathbb{P}(R_t y\in d\zeta)\,\bar{\lambda}_t(dx)\,\bar{\lambda}_t(dy)
\end{aligned}
$$

to make sense even if (5.7) are sometimes violated. Clearly (5.4) will not be satisfied if (5.7) do not hold, but provided this violation happens only with very small probability we can hope to recover a close approximation to (5.4).

5.1.2. Rate kernel. Define jump operators on the space of signed Borel measures on $\tilde{E}$ by

$$J^V_{x,y,z,w};\ \mu\mapsto\mu-V^{-1}(\delta_x+\delta_y-\delta_z-\delta_w)\ \ x,y,z,w\in\tilde{E}$$

and

$$J^V_{x,y,z};\ \mu\mapsto\mu-V^{-1}(\delta_x+\delta_y-\delta_z)\ \ x,y,z\in\tilde{E}$$

and

$$J^V_{x,y};\ \mu\mapsto\mu-V^{-1}(\delta_x-\delta_y)\ \ x,y\in\tilde{E}$$

and

$$J^V_x;\ \mu\mapsto\mu+V^{-1}\delta_x\ x\in\tilde{E}.$$

For $t\in[0,\infty)$ and $V\in(0,\infty)$ we define kernels

$$\alpha^V_t: M\times\mathcal{M}\to[0,\infty)$$

[2]Note that for $a,b\in\mathbb{R}$ we have $a_+\equiv\max(a,0)$ and $a\wedge b\equiv\min(a,b)$.

by specifying the value for $\mu \in M,\ C \in \mathcal{M}$

$$
\begin{aligned}
(5.9)\quad \alpha_t^V(\mu, C) &= \int_{x\in\tilde{E}} 1\left\{J_x^V \mu \in C\right\} V I_t(dx) \\
&+ \sum_{l\in\mathcal{U}'} \int_{x,\xi\in\tilde{E}} 1\left\{J_{x,g^{(l)}(\xi)}^V \mu \in C\right\} V \left[\beta_t^{(l)}(\xi) \wedge \hat{\beta}_t^{(l)}(x;\mu)\right] \mathbb{P}(R_t x \in d\xi)\, \mu(dx) \\
&+ \sum_{l\in\mathcal{U}'} \int_{x,\xi\in\tilde{E}} 1\left\{J_{x,\xi}^V \mu \in C\right\} V \left[\hat{\beta}_t^{(l)}(x;\mu) - \beta_t^{(l)}(\xi)\right]_+ \mathbb{P}(R_t x \in d\xi)\, \mu(dx) \\
&+ \frac{1}{2} \int_{x,y,\xi,\zeta\in\tilde{E}} 1\left\{J_{x,y,\xi+\zeta}^V \mu \in C\right\} V \left[K_t(\xi,\zeta) \wedge \hat{K}_t(x,y;\mu)\right] \\
&\qquad \mathbb{P}(R_t x \in d\xi)\, \mathbb{P}(R_t y \in d\zeta)\, \mu^{(2)}(dx, dy) \\
&+ \frac{1}{2} \int_{x,y,\xi,\zeta\in\tilde{E}} 1\left\{J_{x,y,\xi,\zeta}^V \mu \in C\right\} V \left[\hat{K}_t(x,y;\mu) - K_t(\xi,\zeta)\right]_+ \\
&\qquad \mathbb{P}(R_t x \in d\xi)\, \mathbb{P}(R_t y \in d\zeta)\, \mu^{(2)}(dx, dy),
\end{aligned}
$$

where for $A, B \in \tilde{E}$ we have $\mu^{(2)}(A \times B) = \mu(A)\,\mu(B) - \mu(A \cap B)$.

Using the kernels we define generators for stochastic processes on M by

$$
(5.10)\qquad \mathcal{G}_t^V(\Psi)(\mu) = \int_{\nu\in M} [\Psi(\nu) - \Psi(\mu)]\, \alpha_t^V(\mu, d\nu)
$$

for Ψ in a suitable class of test functions (not necessarily of the form given in (5.5)) and assuming (5.7) we can approximate (5.8) by

$$
(5.11)\qquad \frac{\mathrm{d}}{\mathrm{d}t}\Phi\left(\bar{\lambda}_t\right) = \mathcal{G}_t^V(\Phi)\left(\bar{\lambda}_t\right),
$$

which has an error that is $O\left(V^{-1}\right)$ as $V \to \infty$. By analogy with the work of Eibeck and Wagner (see [5] for a basic introduction and [6] for their latest generalization), we claim that the stochastic jump processes μ_t^V given by the generators $\mathcal{G}_t^V$ converge in distribution as $V \to \infty$ to a deterministic solution of (5.8) provided the initial conditions also converge. We then expect $\mathcal{P}_t\left(\bar{\lambda}_t\right)$ to converge in distribution to a deterministic solution, λ_t, of (5.2) in the same limit, i.e., to be a weak solution of our extended form of Smoluchowski's coagulation equation.

5.2. Implementation of deferment. Our implementation of this algorithm (LPDA) is essentially an application of DSA to the processes given in (5.10). We always used $\mathcal{U}' = \{\text{pyrene}\}$.

Some quantities (e.g., the moments plotted in Figure 4) are recorded frequently, in our case every millisecond. At these times we simulate all deferred growth due to all particles so that the size distribution is fully up to date before storing any data. This has the convenient effect of ensuring no stochastic particle is left for too long before its deferred surface growth is performed. One can therefore think of LPDA as a form of splitting but with a "just in time" feature to remove the main source of approximation error. However, with LPDA the number of times the entire ensemble has to be updated is orders of magnitude less than for ordinary operator splitting and so for practical purposes it is a completely different algorithm.

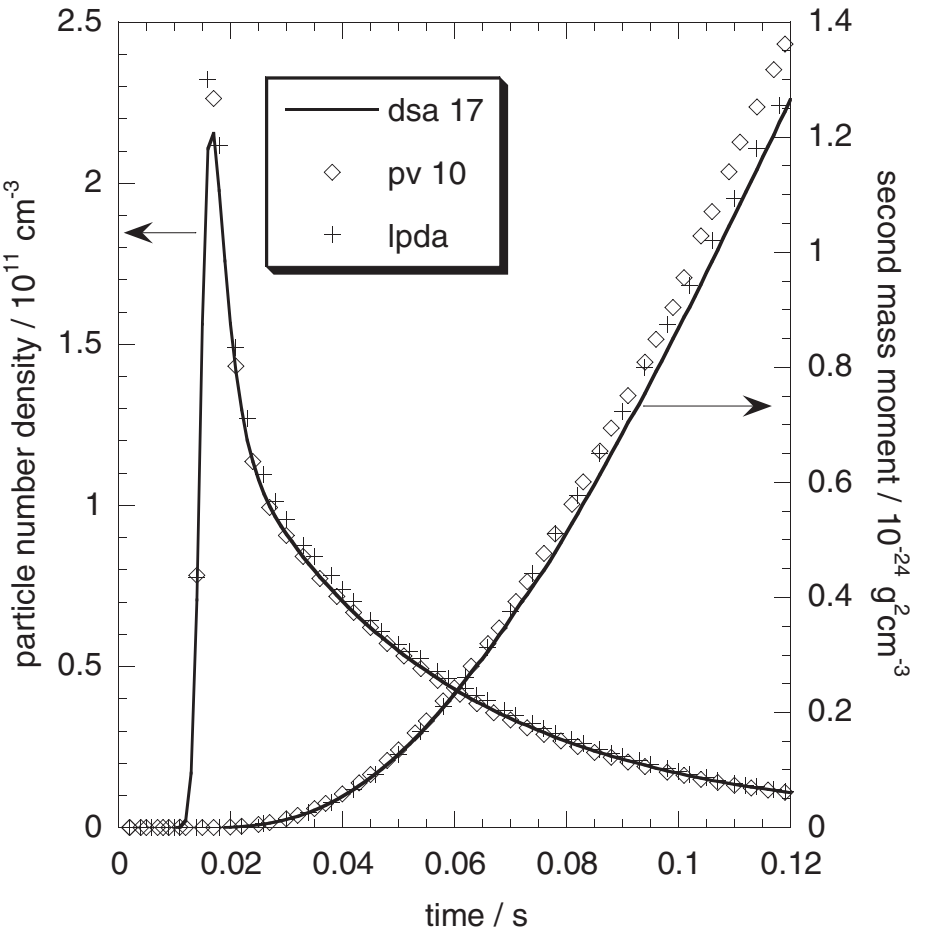


FIG. 4. *LPDA applied to* JW1.69.

5.2.1. Complexity. The (random) number of deferred events due to any particular particle is $O(1)$ in the sample volume V. There are $O(V)$ nondeferred events in a run and each of these requires use of the binary tree which takes $O(\log V)$ per event (see section 3.2). Hence the overall complexity is still $O(V \log V)$ in the (initial) sample volume for reasonable choices of tree depth. However, in section 4.2.2 we noted that most computational effort seemed to go on surface reactions and so if all these were deferred the complexity might not be observed in practice.

5.3. Numerical results.

5.3.1. Accuracy of moments. We used the same JW1.69 test case using a tree depth of 13 as in section 4.3.1. Ten runs took 250 seconds, an appreciable acceleration compared to the operator splitting cases. Figure 4 compares the zeroth and second moments of the mass distribution obtained with LPDA to the DSA precision run and to the PV splitting algorithm with $r = 10$ results. One can see that LPDA slightly overstates the number of particles compared to the other two methods at least until 0.08 s. Interestingly, with the second moment LPDA tracks the operator splitting solution until about 0.1 s and then moves much closer to the DSA solution. The first

TABLE 5
Detail of deferred suface growth runs for JW10.68.

	number of tree levels	time for 10 repetitions / min
DSA	13	8497
LPDA 15	15	22
LPDA 14	14	11
LPDA 13	13	5.25
LPDA 12	12	2.5

moment is not shown but here LPDA is virtually indistiguishable from DSA on the scale of the figures and noticeably better than the operator splitting results.

5.3.2. Particle distribution accuracy. As with operator splitting we investigate this topic for JW10.68; a summary of our runs is given in Table 5. The deferred surface growth runs were performed on the same machines used to generate results for section 4.3.1. We observe very considerable speed gains from using LPDA.

The LPDA run time with 13 tree levels is more than 3 orders of magnitude less than that for DSA, which is a very positive result. Figure 5 shows that the particle distribution is well reproduced using LPDA.

5.4. Conclusions on LPDA. This method is extremely attractive for further use in the simulation of laminar premixed flames because of the large speed gains possible without introducing large errors. The comments on the errors made in section 4.5 apply here as well.

The initial sample volume was scaled in direct proportion to the tree capacity for all the runs; for 13 tree levels we used an initial sample volume of 1×10^{-8} cm^{-3}. It is therefore interesting to see the run times showing $O(V)$ behavior, not $O(V \log V)$, as discussed in section 5.2.1.

6. Comparison of simulation methods.

6.1. Equal numbers of computational particles. A clear ordering in terms of speed can be seen in Table 6. In our view the differences in the particle distributions produced using the different methods are not material.

The ratios of run times between methods are very different for the two flames considered, as are the total run times. Unsurprisingly, the greatest accelerations were achieved for the problem that takes the most CPU time. We believe the reason JW10.68 takes so much longer to simulate than JW1.69 is that there are far more surface reactions.

6.2. Equal precision for JW1.69. In this section we consider the time required to get results of comparable quality using the three main algorithms discussed in this paper. We measure result quality using the second moment of the particle size distribution on exit from the flame. We take as our reference value the mean obtained from 30 runs using DSA, 17 tree levels, and an initial sample volume of 5.6×10^{-7} cm^3. Table 7 gives details of settings which lead to 99.9% confidence intervals for the second moment being entirely contained in an interval of $\pm 10\%$ around the reference value.

The parameters shown were selected as the least multiples of five runs that led to adequate results for the second moment. More runs or more tree levels were required for LPDA and operator splitting because these methods bias the mean value of the second moment and so tighter confidence intervals are needed to stay within the $\pm 10\%$ range.

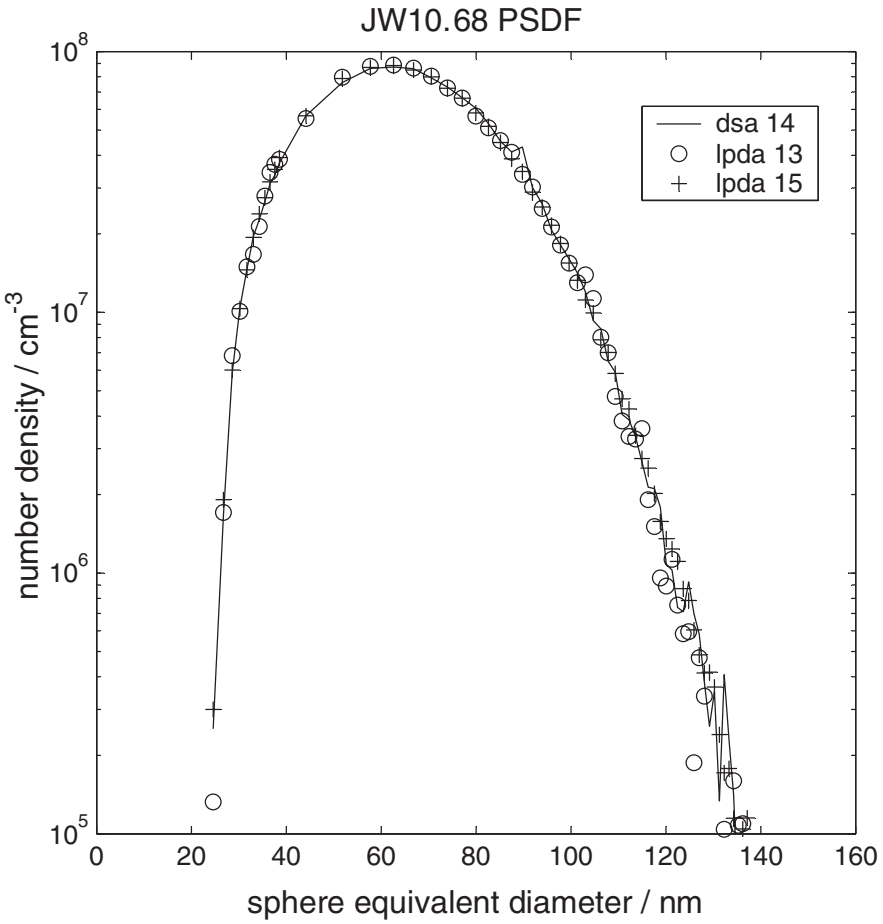


FIG. 5. *Final particle distribution for* JW10.68.

TABLE 6
Comparison of simulation methods.

algorithm	flame	number of tree levels	time for 10 repetitions / min
DSA	JW10.68	13	8497
PV $r = 5$	JW10.68	13	844
LPDA	JW10.68	13	6.33
DSA	JW 1-69	13	50
PV $r = 5$	JW 1-69	13	8
LPDA	JW 1-69	13	4.2

TABLE 7
Time to achieve tolerances for JW1.69.

algorithm	number of tree levels	repetitions	total time / s
DSA	10	30	990
DSA	11	15	645
LPDA	11	15	98
LPDA	12	10	133
PV $r = 5$	12	20	500
PV $r = 5$	13	10	480

7. Conclusion. We have explored the computational challenges of simulating the soot particle size distribution in laminar premixed flames using stochastic methods. We have tested a splitting technique that is well established in many fields of computation and give some indication of the optimum parameter values for this method. Finally we set out for the first time, in detail, a new stochastic algorithm for solving the population balance equations for soot particles in a laminar premixed flame. This algorithm is found to offer significant reductions in computational time and for one sooty, 10 bar flame, LPDA is 1000 times faster.

REFERENCES

[1] J. Appel, H. Bockhorn, and M. Frenklach, *Kinetic modeling of soot formation with detailed chemistry and physics: Laminar premixed flames of* C_2 *hydrocarbons*, Combust. Flame, 121 (2000), pp. 122–136.

[2] M. Balthasar and M. Kraft, *A stochastic approach to solve the particle size distribution function of soot particles in laminar premixed flames*, Combust. Flame, 133 (2003), pp. 289–298.

[3] G. A. Bird, *Molecular Gas Dynamics*, Clarendon Press, Oxford, UK, 1976.

[4] E. Debry, B. Sportisse, and B. Jourdain, *A stochastic approach for the numerical simulation of the general dynamics equation for aerosols*, J. Comput. Phys., 184 (2003), pp. 649–669.

[5] A. Eibeck and W. Wagner, *Approximative solution of the coagulation-fragmentation equation by stochastic particle systems*, Stochastic Anal. Appl., 18 (2000), pp. 921–948.

[6] A. Eibeck and W. Wagner, *Stochastic interacting particle systems and nonlinear kinetic equations*, Ann. Appl. Probab., 13 (2003), pp. 845–889.

[7] M. Frenklach, *Method of moments with interpolative closure*, Chem. Eng. Sci., 57 (2002), pp. 2229–2239.

[8] M. Goodson and M. Kraft, *An efficient stochastic algorithm for simulating nano-particle dynamics*, J. Comput. Phys., 183 (2002), pp. 210–232.

[9] A. R. Gourlay, *Splitting methods for time dependent partial differential equations*, in The State of the Art in Numerical Analysis, D. Jacobs, ed., Academic Press, London, 1977, pp. 757–796.

[10] A. Kazakov, H. Wang, and M. Frenklach, *Detailed modeling of soot formation in laminar premixed ethylene flames at a pressure of* 10 *bar*, Combust. Flame, 100 (1995), pp. 111–120.

[11] A. Kolodko and K. Sabelfeld, *Stochastic particle methods for Smoluchowski coagulation equation: Variance reduction and error estimations*, Monte Carlo Methods Appl., 9 (2003), pp. 315–339.

[12] A. Maisels, F. E. Kruis, and H. Fissan, *Direct simulation Monte Carlo for simultaneous nucleation, coagulation and surface growth in dispersed systems*, Chem. Eng. Sci., 59 (2004), pp. 2231–2239.

[13] K. K. Sabelfeld, S. V. Rogasinsky, A. A. Kolodko, and A. I. Levykin, *Stochastic algorithms for solving Smoluchovsky coagulation equation and applications to aerosol growth simulation*, Monte Carlo Methods Appl., 2 (1996), pp. 41–87.

[14] J. Singh, R. Patterson, M. Balthasar, M. Kraft, and W. Wagner, *Modelling soot particle size distribution: Dynamics of pressure regimes*, Tech. Report 25, c4e Preprint-Series, Cambridge, UK, 2004.

[15] M. Smith and T. Matsoukas, *Constant-number Monte Carlo simulation of population balances*, Chem. Eng. Sci., 53 (1998), pp. 1777–1786.

[16] M. D. Smooke, C. S. McEnally, L. Pfefferle, R. J. Hall, and M. B. Colket, *Computational and experimental study of soot formation in a coflow, laminar diffusion flame*, Combust. Flame, 117 (1999), pp. 117–139.

[17] C. G. Wells, N. M. Morgan, M. Kraft, and W. Wagner, *Bivariate stochastic modelling of nanoparticles*, Tech. Report 23, c4e Preprint-Series, University of Cambridge, Cambridge, UK, 2004.

SIAM J. SCI. COMPUT.
Vol. 28, No. 1, pp. 321–338
© 2006 Society for Industrial and Applied Mathematics

A DEFECT CORRECTION SCHEME FOR FINITE ELEMENT EIGENVALUES WITH APPLICATIONS TO QUANTUM CHEMISTRY*

LIHUA SHEN† AND AIHUI ZHOU†

Abstract. To produce highly accurate approximations for self-adjoint elliptic eigenvalues, we propose a finite element defect correction scheme. Its efficiency is proven by theoretical and numerical evidence. Very satisfying applications in computational quantum chemistry are also presented.

Key words. defect correction, eigenvalue, finite element, Hartree–Fock equation, highly accurate, quantum chemistry

AMS subject classifications. 65N15, 65N25, 65N30, 65N50, 81Q05

DOI. 10.1137/040614013

1. Introduction.

It is known that the time independent Schrödinger equation models the behavior of a quantum molecular system. This equation deals with state functions $\psi(x)$, where x denotes the position of the particles (nuclei and electrons) and hence is a variable that lives in high dimensions. In quantum chemistry, it is significant to compute the total energy of the ground state of the system. However, this system is far too large to be directly tractable by numerical simulation for molecules larger than the hydrogen atom. Quantum chemists have thus introduced a series of simplified models. One of them is the so-called Hartree–Fock model [27, 28], which is a nonlinear eigenvalue system in three dimensions. In computation, approximations to the total energy of the ground state of the system require solving self-consistently a large number of linear eigenvalue problems. Namely, an iteration procedure for solving a large number of linear eigenvalue problems must be involved. Moreover, in order to obtain the numerical solution with satisfactory accuracy, the number of iterations is usually very large. Hence it is important to reduce the computational cost or improve the approximation accuracy in solving such linear eigenvalue problems (in three dimensions) at each iteration step. The purpose of this paper is to present a simple postprocessing approach to getting highly accurate approximations for a class of linear eigenvalue problems.

As the finite element method is one of the most effective numerical methods, we shall consider the use of finite element schemes to discretize eigenvalue problems. We first investigate a finite element postprocessing approach for the following model problem theoretically and numerically: Find $(u, \lambda) \in H_0^1(\Omega) \times R$ such that

$$\begin{cases} -\nabla(A\nabla u) + \beta u = \lambda u & \text{in } \Omega, \\ \|u\|_{0,\Omega} = 1, \end{cases} \tag{1.1}$$

*Received by the editors August 28, 2004; accepted for publication (in revised form) September 14, 2005; published electronically March 17, 2006. This work was supported in part by the National Natural Science Foundation of China under grant 10425105 and by the National Basic Research Program under grant 2005CB321704.

http://www.siam.org/journals/sisc/28-1/61401.html

†State Key Laboratory of Scientific and Engineering Computing (LSEC), Institute of Computational Mathematics and Scientific/Engineering Computing, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100080, P.O. Box 2719, China (shenlh@lsec.cc.ac.cn, azhou@lsec.cc.ac.cn).

where $\Omega \subset R^d (d = 2, 3)$ is a polygonal domain with boundary $\partial\Omega, A$ is a smooth positive definite symmetric matrix, and $0 \leq \beta \in L^\infty(\Omega)$. Then we turn to studying two Hartree–Fock problems and apply a similar numerical approach to achieve highly accurate approximations. One Hartree–Fock problem we will study is: Find $(u, E) \in H^1(R^3) \times R$, such that $\int_{R^3} |u|^2 dx = 1$ and

$$-\frac{1}{2}\Delta u - \frac{1}{|x|}u = Eu \quad \text{in } R^3, \tag{1.2}$$

while the other Hartree–Fock problem is: Find $(u, E) \in H^1(R^3) \times R$ such that $\int_{R^3} |u|^2 dx = 1$ and

$$-\frac{1}{2}\Delta u - \frac{2}{|x|}u + \int_{R^3} \frac{|u(y)|^2 dy}{|x - y|}u = Eu \quad \text{in } R^3. \tag{1.3}$$

Obviously, the Hartree–Fock equations are singular. For such singular problems, it is not so easy to construct highly accurate finite element approximations.

Highly accurate finite element approximations have been studied for decades [14, 17, 21, 23, 24, 32, 44]. In the context of solving boundary value problems, it is noted that most previous work focused on searching some superconvergent points when the associated data are smooth. For improving the accuracy of finite element eigenvalues, however, we need to study highly accurate approximations for eigenfunctions at all points of the underlying domain. It is now found that it is possible to get finite element approximations for eigenfunctions with higher accuracy by using some averaging techniques. As for highly accurate finite element approximations for eigenvalues, it is shown from our investigation that the averaging techniques need to be combined with some defect correction procedures.

It is observed from finite element discretizations that finite element solutions are usually oscillating even though the exact solutions are smooth. It is also seen that the finite element solutions can be reconstructed to be smooth if some averaging approaches are used and also that the smoothed finite element approximations would have better approximation accuracy, provided that the exact solutions are smooth. Historically, averaging approaches have indeed been applied in finite element computations. To our knowledge, the most popular averaging technique is the Zienkiewicz–Zhu (ZZ) gradient patch recovery [45, 46], which is based on a local discrete least-squares fitting. This technique has been widely used in engineering practices for its robustness in a posteriori error estimates and its efficiency in computer implementation. Superconvergence properties of the ZZ recovery for boundary value problems are studied by Zhang [39], Li and Zhang [25], and Zhang and Victory [41]. We refer to [2, 5, 6, 7, 12, 19, 20, 33, 36, 38, 40] for other investigations on superconvergence and a posteriori error control for boundary value problems by using various averaging techniques.

Based on the ZZ-averaging techniques, in this paper we will propose a defect correction scheme to produce highly accurate finite element eigenvalue approximations for (1.1), (1.2), and (1.3). Our defect correction scheme is nothing but a simple post-processing of the computed eigenvalue and eigenfunction. Further, it is shown that our defect correction scheme is efficient not only for smooth data problems but also for nonsmooth data problems. In particular, our defect scheme has been successfully applied to computational quantum chemistry and has obtained very satisfying results.

Note that it should be interesting to apply the approaches proposed in this paper to more complex quantum molecular systems. Indeed, it is our ongoing research to

solve the so-called Hartree–Fock–Slater equations (or X_α equations) and Kohn–Sham equations by the finite element method and to use the defect correction scheme to get highly accurate approximations for the eigenvalues and hence for the total energy for complex molecular systems.

The rest of this paper is organized as follows. In the next section, some preliminaries, including the finite element discretization and averaging operators, are presented. In section 3, a finite element defect correction scheme is proposed and analyzed. Examples of highly accurate finite element approximations for both eigenfunctions and eigenvalues are then constructed for (1.1) by using some averaging techniques, together with the defect correction procedure. Several numerical results that support our theory are also reported. In section 4, the finite element defect correction scheme, together with the averaging approaches, is applied to solving (1.2) and (1.3), which are two typical models in quantum chemistry. It is shown that our defect correction scheme is very efficient in such computations, too.

2. Preliminaries. We shall use the standard notation for Sobolev spaces $W^{s,p}(\Omega)$ and their associated norms and seminorms; see, e.g., [1, 15]. For $p = 2$, we denote $H^s(\Omega) = W^{s,2}(\Omega)$ and $H_0^1(\Omega) = \{v \in H^1(\Omega) : \ v|_{\partial\Omega} = 0\}$, where $v|_{\partial\Omega}$ is in the sense of trace, $\|\cdot\|_{s,\Omega} = \|\cdot\|_{s,2,\Omega}$, and $(\cdot,\cdot)$ is the standard L^2-inner product. Throughout this paper, we shall use C to denote a generic positive constant, which may stand for different values at its different occurrences and is independent of the mesh parameters.

Assume that $T^h = \{\tau\}$ is a triangulation of Ω with mesh size h, which consists of shape-regular simplices. Denote by ∂T^h the set of all edges of triangles for $d = 2$ or all surfaces of simplices for $d = 3$, by $\partial^2 T^h$ the set of all vertices of T^h, and let $\Lambda = \partial^2 T^h \setminus \partial\Omega$. We define finite element spaces

$$S^{h,r}(\Omega) = \{\ v \in H^1(\Omega) : \ v|_\tau \in P^r(\tau),\ \forall \tau \in T^h\}, \qquad S_0^{h,r}(\Omega) = S^{h,r}(\Omega) \cap H_0^1(\Omega),$$

where $P^r(\tau)$ is the space of polynomials of degree not greater than a positive integer r. In most of this paper we use $S^h(\Omega)$ and $S_0^h(\Omega)$ to denote $S^{h,1}(\Omega)$ and $S_0^{h,1}(\Omega)$, respectively. For a given $z \in \partial^2 T^h$, $l \in \partial T^h$, and $\tau \in T^h$, let ϕ_z be the nodal basis function of $S^h(\Omega)$ corresponding to z (i.e., the continuous piecewise linear function with respect to T^h, which takes the value 1 at z and which vanishes at all other vertices) and define

$$\omega_z = \bigcup_{z \in \bar{\tau}} \tau, \qquad \omega_l = \bigcup_{l \subset \bar{\tau}} \tau, \qquad \omega_\tau = \bigcup_{\bar{\tau}' \cap \tau \neq \emptyset} \tau'.$$

2.1. ρ-regular meshes. In discussing superconvergence of finite element approximations to boundary value problems, the finite element mesh T^h is usually required to be regular enough; for instance, it satisfies

$$\left| \int_\Omega A\nabla(u - u_I) \cdot \nabla v dx \right| \le Ch^{1+\rho} |u|_{2+\alpha,\Omega} |v|_{1,\Omega} \quad \forall v \in S_0^h(\Omega), \tag{2.1}$$

$$\left| \int_\Omega (u - u_I) \mathrm{div}(\varphi v) dx \right| \le Ch^{1+\rho} |u|_{2+\alpha,\Omega} |v|_{0,\Omega} \quad \forall v \in S^h(\Omega) \tag{2.2}$$

for $\varphi \in (W^{1,\infty}(\Omega))^d$ and $u \in H_0^1(\Omega) \cap H^{2+\alpha}(\Omega)$ with $\alpha > 0$. Here $\rho = \min(\alpha, 1/2)$, and u_I is the standard linear Lagrange interpolant of $u \in C(\bar{\Omega})$, namely

$$u_I \in S^h \ \text{ and } \ u_I = u \qquad \text{on } \partial^2 T^h.$$

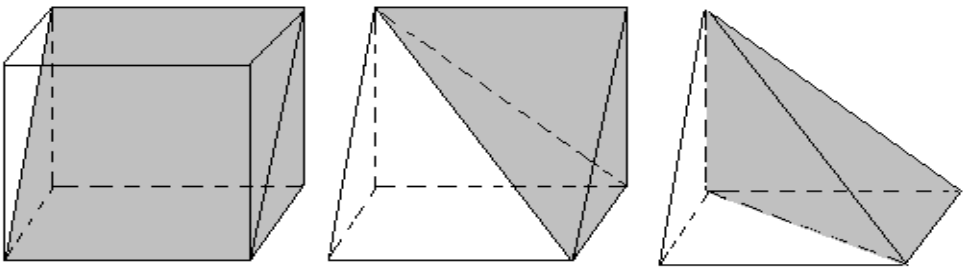


FIG. 1. *Kuhn's triangulation.*

It is known that (2.1) and (2.2) are satisfied for $\alpha = 1$ if T^h is piecewise uniform; see, e.g., [9, 14, 17, 21, 22, 23, 36, 42, 44] for (2.1) and [42, 43] for (2.2).

For a triangulation T^h in two dimensions, if the union of any two triangles (in T^h) that share an entire edge forms a parallelogram, then T^h is called uniform. A uniform triangulation T^h in three dimensions is induced from Kuhn's triangulation [9, 31]. Kuhn's triangulation T_0 (see Figure 1) is a partition of the cube $[0,1]^3$ consisting of six tetrahedra $\{\tau_\sigma : \sigma = (\sigma_1, \sigma_2, \sigma_3) \in \Sigma\}$ with Σ the group of permutations of the numbers $1, 2$, and 3, and each τ_σ is given by

$$\tau_\sigma = \{(x_1, x_2, x_3) \in [0,1]^3 : 0 \le x_{\sigma_1} \le x_{\sigma_2} \le x_{\sigma_3} \le 1\}.$$

Let $T = \{T_0 + z : z \in Z^3\}$ be the partition of R^3. Then a triangulation T^h in three dimensions is called uniform if there exist a nonsingular 3×3 matrix $\mathcal{B}$ and a point $w \in R^3$ such that $\tau \subset \bar{\Omega}$ or $\tau \subset \overline{R^3 \setminus \Omega}$ for all $\tau \in T_{\mathcal{B},w}$ and

$$T^h = \{\tau \in T_{\mathcal{B},w} : \tau \subset \bar{\Omega}\},$$

where $\mathcal{B}/h$ is independent of h, and

$$T_{\mathcal{B},w} = \{\mathcal{B}\tau + w : \tau \in T\}.$$

The partition $T_{\mathcal{B},w}$ is also called a uniform partition of R^3 for any given nonsingular 3×3 matrix $\mathcal{B}$ and $w \in R^3$. A shape-regular triangulation T^h is called piecewise uniform if there exist subdomains $\Omega_1, \Omega_2, \ldots, \Omega_m$ such that $\bar{\Omega} = \cup_{j=1}^m \bar{\Omega}_j$ and $T^h|_{\Omega_j}$ is a uniform triangulation of the subdomain Ω_j $(j = 1, 2, \ldots, m)$. One sees that any polygonal domain in two or three dimensions can be covered by a piecewise uniform triangulation.

We refer to [5, 36, 42] for other meshes that produce (2.1) and (2.2). For simplicity, we shall call a shape-regular mesh T^h ρ-regular if (2.1) and (2.2) are satisfied, and a shape-regular mesh T^h without any additional requirement irregular.

2.2. Averaging operators. In this subsection, we introduce global and local averaging operators: the L^2-projection operator and a Clément-type operator.

Associated with the finite element space $S^h(\Omega)$, the L^2-projection operator $Q_h : L^2(\Omega) \mapsto S^h(\Omega)$ is defined by

$$(Q_h w, v) = (w, v) \quad \forall v \in S^h(\Omega).$$

The following standard estimates for Q_h,

$$\|w - Q_h w\|_{l,\Omega} \le Ch^{s-l}|w|_{s,\Omega} \qquad \forall w \in H^s(\Omega),\ l \le s \le 2, l = 0, 1,$$

will be used in our analysis. Here and hereafter, we use the notation $Q_h(v_1, v_2) = (Q_h v_1, Q_h v_2)$ for $(v_1, v_2) \in L^2(\Omega) \times L^2(\Omega)$.

If T^h is ρ-regular, then the inequality

$$\|A\nabla u - Q_h A\nabla u_I\|_{0,\Omega} \le Ch^{1+\rho} \tag{2.3}$$

can be derived directly from (2.2) (cf. [5]). Note that a slightly different estimate from (2.3) can be found in [5], where a variant estimate of (2.2) was used.

Define a local averaging operator

$$C_h : A\nabla S_0^h(\Omega) \mapsto \underbrace{S^h(\Omega) \times \cdots \times S^h(\Omega)}_{d \text{ times}}$$

by (cf., e.g., [2, 26, 38, 45])

$$\mathcal{C}_h A\nabla v = \sum_{z \in \partial^2 T^h} (A\nabla v)_z \phi_z, \quad (A\nabla v)_z = \sum_{j=1}^{J_z} \alpha_z^j \big(A(z)\nabla v\big)_{\tau_z^j} \quad \forall v \in S_0^h(\Omega), \tag{2.4}$$

where $\bigcup_{j=1}^{J_z} \tau_z^j = \omega_z$, $\sum_{j=1}^{J_z} \alpha_z^j = 1$, and $\alpha_z^j \ge 0$. For instance, $\alpha_z^j = \frac{1}{J_z}$ or $\alpha_z^j = \frac{|\tau_z^j|}{|\omega_z|}$. One sees that $\mathcal{C}_h$ is a Clément-type operator. By the local averaging operator $\mathcal{C}_h$, a smoothened fluxfield $\mathcal{C}_h A\nabla v$ is then obtained from a fluxfield $A\nabla v$. Hence, the operator $\mathcal{C}_h$ may be called a flux recovery operator. Indeed, the operator $\mathcal{C}_h$ is nothing but a discretized L^2-projection. The so-called gradient recovery operator is discussed in [2, 11, 38, 45].

In contrast to [2, 11, 38, 45], here we are required to apply flux recovery operators to construct superconvergent approximations for $A^{1/2}\nabla u$, where u is some eigenfunction. We may also employ other types of local averaging operators, for instance, a local L^2-projection operator, a local discrete least-squares fitting operator, or Zhang's fitting operator (see, e.g., [12, 25, 40]). In this paper, we shall apply the local averaging operator C_h defined by (2.4). It is easy to see that

$$\|\mathcal{C}_h A\nabla v\|_{0,\Omega} \le \|v\|_{a,\Omega} \quad \forall v \in S_0^h(\Omega). \tag{2.5}$$

A direct calculation leads to (cf. [9, 36, 38])

$$\|(A\nabla u)_I - C_h A\nabla u_I\|_{0,\Omega} \le Ch^{1+\rho}\|u\|_{2+\alpha,\Omega} \tag{2.6}$$

if T^h is ρ-regular.

2.3. A model problem. Define a bilinear form $a(\cdot,\cdot) : H^1(\Omega) \times H^1(\Omega) \mapsto R$ by

$$a(w, v) = \int_\Omega (A\nabla w \cdot \nabla v + \beta w v)dx,$$

which obviously satisfies

$$a(w, v) \le C\|w\|_{1,\Omega}\|v\|_{1,\Omega} \quad \forall w, v \in H_0^1(\Omega).$$

It is noted that $a(\cdot,\cdot)$ can induce a norm $\|\cdot\|_{a,\Omega}$ on $H_0^1(\Omega)$ defined by

$$\|w\|_{a,\Omega}^2 = a(w,w) \quad \forall w \in H_0^1(\Omega).$$

The weak form of (1.1) is stated as follows: Find $(u,\lambda) \in H_0^1(\Omega) \times R$ such that

$$\begin{cases} a(u,v) &= \lambda(u,v) \quad \forall v \in H_0^1(\Omega), \\ \|u\|_{0,\Omega} &= 1. \end{cases} \tag{2.7}$$

It is known that (2.7) has a sequence of real eigenvalues $0 < \lambda_1 \le \lambda_2 \le \cdots$ and the corresponding eigenfunctions $u_1, u_2, \ldots$, which can be assumed to satisfy $(u_i, u_j) = \delta_{ij}$ $(i,j \ge 1)$.

The finite element discretization for (2.7) reads: Find $(u_h, \lambda_h) \in S_0^h(\Omega) \times R$ such that

$$\begin{cases} a(u_h, v_h) &= \lambda_h(u_h, v_h) \quad \forall v_h \in S_0^h(\Omega), \\ \|u_h\|_{0,\Omega} &= 1. \end{cases} \tag{2.8}$$

One sees that (2.8) has a finite sequence of eigenvalues $0 < \lambda_{1,h} \le \lambda_{2,h} \le \cdots \le \lambda_{n_h,h}$ and the corresponding eigenfunctions $u_{1,h}, u_{2,h}, \ldots, u_{n_h,h}$ with $(u_{i,h}, u_{j,h}) = \delta_{ij}$ $(1 \le i,j \le n_h)$, where $n_h = \dim S_0^h(\Omega)$. It is well known that (see, e.g., [3, 4, 13])

$$\lambda_{i,h} \ge \lambda_i, \quad i = 1,2,\ldots,n_h. \tag{2.9}$$

It is shown in [3, 4] (see also [37]) that

$$\|u - u_h\|_{0,\Omega} \le C\eta(h)\|u - u_h\|_{a,\Omega} \tag{2.10}$$

holds with $\eta(h) \to 0$ as $h \to 0$.

In our analysis, we will use the following property of eigenvalue and eigenfunction approximations (see, e.g., [3, 4, 37]).

LEMMA 2.1. [illegible] $(u,\lambda) \in H_0^1(\Omega) \times R$ [illegible] (2.7) [illegible] $w \in H_0^1(\Omega)$ [illegible]

$$\frac{a(w,w)}{\|w\|_{0,\Omega}^2} - \lambda = \frac{a(w-u, w-u)}{\|w\|_{0,\Omega}^2} - \lambda\frac{\|w-u\|_{0,\Omega}^2}{\|w\|_{0,\Omega}^2}. \tag{2.11}$$

The equality (2.11) is straightforward and useful. For instance, if $(u_h, \lambda_h) \in S_0^h(\Omega) \times R$ satisfies (2.8), then it is easy to obtain from (2.11) that

$$\lambda_h - \lambda = \frac{(A\nabla(u_h - u), \nabla(u_h - u)) + (\beta(u_h - u), u_h - u)}{\|u_h\|_{0,\Omega}^2} - \lambda\frac{(u_h - u, u_h - u)}{\|u_h\|_{0,\Omega}^2}$$

and hence

$$\lambda_h - \lambda = \|u_h - u\|_{a,\Omega}^2 + O(\|u_h - u\|_{0,\Omega}^2) = O(\|u_h - u\|_{a,\Omega}^2). \tag{2.12}$$

In this section, we assume that Ω is regular enough (for instance, Ω is a convex domain or a domain with a smooth boundary) so that for any $f \in L^2(\Omega)$ there exists a unique solution $u \in H_0^1(\Omega) \cap H^2(\Omega)$ satisfying

$$a(u,v) = (f,v) \quad \forall v \in H_0^1(\Omega).$$

As a result, the exact solution of (1.1) or (2.7) has the $H^2(\Omega)$-regularity, namely $M(\lambda_i) \subset H_0^1(\Omega) \cap H^2(\Omega)$, where

$$M(\lambda_i) = \{w \in H_0^1(\Omega) : w \text{ is an eigenfunction of } \lambda_i,\ \|w\|_{0,\Omega} = 1\}, \quad i = 1, 2, \ldots.$$

Thus, $\lambda_{i,h} - \lambda_i = O(h^2)\,(i = 1, 2, \ldots, n_h)$, and for $u_{i,h}$, there exists $\hat{u}_i \in M(\lambda_i)$ satisfying (see, e.g., [3, 4, 13])

$$h\|\hat{u}_i - u_{i,h}\|_{a,\Omega} + \|\hat{u}_i - u_{i,h}\|_{0,\Omega} \le Ch^2 \tag{2.13}$$

when $h \ll 1$.

In this paper we assume that (u, λ) and (u_h, λ_h) are solutions of (2.7) and (2.8), respectively, and satisfy

$$\lambda_h - \lambda + \|u - u_h\|_{0,\Omega} + h\|u - u_h\|_{a,\Omega} \le Ch^2. \tag{2.14}$$

Let $R_h : H_0^1(\Omega) \mapsto S_0^h(\Omega)$ be the Ritz–Galerkin projection operator, namely

$$a(R_h w, v) = a(w, v) \quad \forall v \in S_0^h(\Omega),\ \forall w \in H^1(\Omega).$$

Between the Ritz–Galerkin projection $R_h u$ of the eigenfunction and the finite element approximation u_h to the eigenfunction u, there is a super-close relationship (see [37]), as the next lemma shows.

LEMMA 2.2.

$$\|u_h - R_h u\|_{a,\Omega} \le C(\lambda_h - \lambda + \lambda\|u_h - u\|_{0,\Omega}). \tag{2.15}$$

To get approximations with higher accuracy, we require that the eigenfunction $u \in H_0^1(\Omega) \cap H^{2+\alpha}(\Omega)$ for some $\alpha \in (0, 1]$, which is reasonable (cf., e.g., [9, 17, 20, 23, 25, 32, 34, 44, 45, 46]). Consequently,

$$\|u_h - R_h u\|_{a,\Omega} \le Ch^2. \tag{2.16}$$

Note that (2.1) implies

$$\|R_h u - u_I\|_{a,\Omega} \le Ch^{1+\rho}, \tag{2.17}$$

which is well known in the superconvergence literature. Combining (2.16) and (2.17), we immediately obtain the following result.

LEMMA 2.3. [illegible] $u \in H_0^1(\Omega) \cap H^{2+\alpha}(\Omega)$ [illegible] T^h [illegible]

$$\|u_h - u_I\|_{a,\Omega} = O(h^{1+\rho}). \tag{2.18}$$

3. A defect correction scheme. Based on a recovery operator

$$\mathcal{F} : S_0^h(\Omega) \longrightarrow \underbrace{H^1(\Omega) \times \cdots \times H^1(\Omega)}_{d \text{ times}},$$

we construct a defect correction scheme for approximating eigenvalues as follows:

$$\lambda_h^{\mathcal{F}} = \lambda_h - \|\mathcal{F}u_h - A^{1/2}\nabla u_h\|_{0,\Omega}^2. \tag{3.1}$$

For the error analysis, we have the following estimation for $\lambda_h^{\mathcal{F}}$.

PROPOSITION 3.1. [illegible]

$$\mathcal{F} : S_0^h(\Omega) \longrightarrow \underbrace{H^1(\Omega) \times \cdots \times H^1(\Omega)}_{d \text{ times}},$$

[illegible]

$$\left|\lambda_h^{\mathcal{F}} - \lambda\right| \le C\|u - u_h\|_{a,\Omega}\|A^{1/2}\nabla u - \mathcal{F}u_h\|_{0,\Omega}$$

$$+ C(\|A^{1/2}\nabla u - \mathcal{F}u_h\|_{0,\Omega}^2 + \|u - u_h\|_{0,\Omega}^2). \tag{3.2}$$

[illegible] $\mathcal{F}$ [illegible]

$$\|\mathcal{F}v\|_{0,\Omega} \le C\|v\|_{a,\Omega} \quad \forall v \in S_0^h(\Omega), \tag{3.3}$$

[illegible]

$$\left|\lambda_h^{\mathcal{F}} - \lambda\right| \le C\|u - u_h\|_{a,\Omega} \inf_{v \in S_0^h(\Omega)} \left(\|v - u_h\|_{a,\Omega} + \|A^{1/2}\nabla u - \mathcal{F}v\|_{0,\Omega}\right)$$

$$+ C \inf_{v \in S_0^h(\Omega)} \left(\|v - u_h\|_{a,\Omega} + \|A^{1/2}\nabla u - \mathcal{F}v\|_{0,\Omega}\right)^2 + C\|u - u_h\|_{0,\Omega}^2. \tag{3.4}$$

[illegible] From the equality

$$A^{1/2}\nabla(u - u_h) = \mathcal{F}u_h - A^{1/2}\nabla u_h + A^{1/2}\nabla u - \mathcal{F}u_h, \tag{3.5}$$

we have

$$\left|\|\mathcal{F}u_h - A^{1/2}\nabla u_h\|_{0,\Omega} - \|u - u_h\|_{a,\Omega}\right| \le \|A^{1/2}\nabla u - \mathcal{F}u_h\|_{0,\Omega},$$

which implies

$$\left|\|\mathcal{F}u_h - A^{1/2}\nabla u_h\|_{0,\Omega}^2 - \|u - u_h\|_{a,\Omega}^2\right|$$

$$\le 2\|u - u_h\|_{a,\Omega}\|A^{1/2}\nabla u - \mathcal{F}u_h\|_{0,\Omega} + \|A^{1/2}\nabla u - \mathcal{F}u_h\|_{0,\Omega}^2. \tag{3.6}$$

Hence we obtain (3.2) from (2.12).

Using the boundedness of $\mathcal{F}$ and the estimation

$$\|A^{1/2}\nabla u - \mathcal{F}u_h\|_{0,\Omega} \le \|\mathcal{F}(v - u_h)\|_{0,\Omega} + \|A^{1/2}\nabla u - \mathcal{F}v\|_{0,\Omega} \quad \forall v \in S_0^h(\Omega),$$

we get

$$\|A^{1/2}\nabla u - \mathcal{F}u_h\|_{0,\Omega} \le C \inf_{v \in S_0^h(\Omega)} \left(\|v - u_h\|_{a,\Omega} + \|A^{1/2}\nabla u - \mathcal{F}v\|_{0,\Omega}\right), \tag{3.7}$$

which together with (3.2) produces (3.4). □

It is seen from (2.10) that $\lambda_h^{\mathcal{F}}$ is a highly accurate approximation to λ if

$$\inf_{v \in S_0^h(\Omega)} \left(\|v - u_h\|_{a,\Omega} + \|A^{1/2}\nabla u - \mathcal{F}v\|_{0,\Omega}\right) \ll \|u - u_h\|_{a,\Omega}, \tag{3.8}$$

as compared with λ_h. Hence, it is significant to construct a recovery operator $\mathcal{F}$ such that (3.8) is satisfied. Indeed, it is shown from our investigation that the general defect correction scheme (3.1) can produce highly accurate finite element eigenvalue approximations whether the data are smooth or not when some suitable recovery operators $\mathcal{F}$ are employed.

3.1. Averaging recovery on regular meshes. Applying the L^2-projection averaging, approximations for eigenvalues and eigenfunctions with higher accuracy are then constructed on ρ-regular meshes.

THEOREM 3.2. [illegible] $u \in H_0^1(\Omega) \cap H^{2+\alpha}(\Omega)$ [illegible] $T^h(\Omega)$, [illegible] $\mathcal{F} = A^{-1/2} Q_h A \nabla$, [illegible]

$$\|A^{1/2}\nabla u - \mathcal{F} u_h\|_{0,\Omega} \le C h^{1+\rho}, \tag{3.9}$$

$$|\lambda_h^{\mathcal{F}} - \lambda| \le C h^{2+\rho}. \tag{3.10}$$

[illegible] The estimation (3.10) can be directly derived from (3.2), (2.14), and (3.9). Hence we need only prove (3.9). From (2.3) and the equality

$$A^{1/2}\nabla u - A^{-1/2} Q_h A \nabla u_I = A^{-1/2}(A\nabla u - Q_h A \nabla u_I)$$

we get

$$\|A^{1/2}\nabla u - \mathcal{F} u_I\|_{0,\Omega} \le C h^{1+\rho}. \tag{3.11}$$

Since A is a smooth positive definite symmetric matrix and

$$\|Q_h v\|_{0,\Omega} \le \|v\|_{0,\Omega} \quad \forall v \in L^2(\Omega),$$

we see that (3.3) is satisfied, and hence (3.7) leads to

$$\|A^{1/2}\nabla u - \mathcal{F} u_h\|_{0,\Omega} \le C \left(\|u_I - u_h\|_{a,\Omega} + \|A^{1/2}\nabla u - \mathcal{F} u_I\|_{0,\Omega} \right), \tag{3.12}$$

which together with (2.18) and (3.11) yields (3.9). □

Note that (3.9) is an L^2-projection averaging–based superconvergence result. We should mention here that by using the L^2-projection averaging, a similar superconvergence estimate for a finite element solution to elliptic boundary value problems has been provided in [5].

We may also obtain highly accurate approximations on ρ-regular meshes by using the local averaging operator $\mathcal{C}_h$, which is a discretized L^2-projection (cf. also [26]). In fact, from (2.6) and

$$A^{1/2}\nabla u - A^{-1/2}\mathcal{C}_h A \nabla u_I = A^{-1/2}(A\nabla u - (A\nabla u)_I) + A^{-1/2}((A\nabla u)_I - \mathcal{C}_h A \nabla u_I)$$

we have

$$\|A^{1/2}\nabla u - A^{-1/2}\mathcal{C}_h A \nabla u_I\|_{0,\Omega} \le C h^{1+\rho} \tag{3.13}$$

when T^h is ρ-regular and $u \in H_0^1(\Omega) \cap H^{2+\alpha}(\Omega)$. Note that (3.3) is valid if $\mathcal{F} = A^{-1/2}\mathcal{C}_h A \nabla$; we then get

$$\|A^{1/2}\nabla u - \mathcal{F} u_h\|_{0,\Omega} \le C h^{1+\rho} \tag{3.14}$$

from (2.18), (3.13), and (3.12), and

$$|\lambda_h^{\mathcal{F}} - \lambda| \le C h^{2+\rho} \tag{3.15}$$

from (2.14), (3.14), and (3.2).

[illegible] It should be pointed out that if T^h is a piecewise uniform mesh obtained from T^{2h} by some refinements, it may be possible to produce highly accurate approximations to eigenvalues and eigenfunctions by applying some quadratic interpolation operator. Unfortunately, it is not so easy to construct quadratic interpolation operators on unstructured meshes.

3.2. Averaging recovery on irregular meshes. By using the averaging technique, we may also obtain highly accurate approximations on irregular meshes if we use other coarser meshes. More precisely, we may construct highly accurate approximations on shape-regular meshes if some L^2-projections on coarser meshes are applied.

For $H \gg h$, let $Q_H : L^2(\Omega) \longrightarrow S^H(\Omega)$ and $\tilde{Q}_H : L^2(\Omega) \longrightarrow S^{H,2}(\Omega)$ be the L^2-projection operators; namely,

$$(Q_H w - w, v) = 0 \qquad \forall v \in S^H(\Omega), \forall w \in L^2(\Omega),$$

$$(\tilde{Q}_H w - w, v) = 0 \qquad \forall v \in S_0^{H,2}(\Omega), \forall w \in L^2(\Omega).$$

By using the L^2-projections on the coarser mesh T^H, superconvergence for the gradient of finite element approximations to boundary value problems has been studied in [19, 33]. In the following, we shall apply this technique together with the defect correction to finite element approximations to eigenvalue problems.

THEOREM 3.3. Let $u \in H_0^1(\Omega) \cap H^{2+\alpha}(\Omega)$ and $\mathcal{F} = A^{-1/2} Q_H A \nabla$, then

$$\|A^{1/2}\nabla u - \mathcal{F} u_h\|_{0,\Omega} \le C(H^{1+\alpha} + h^2/H), \tag{3.16}$$

$$|\lambda_h^{\mathcal{F}} - \lambda| \le C(H^{1+\alpha} + h)(H^{1+\alpha} + h^2/H). \tag{3.17}$$

PROOF. For any $v \in (L^2(\Omega))^d$ we have

$$\begin{aligned} |(Q_H A\nabla(u - u_h), v)| &= |(A\nabla(u - u_h), Q_H v)| \\ &= |(u - u_h, \operatorname{div}(AQ_H v))| \le \|u - u_h\|_{0,\Omega} |AQ_H v|_{1,\Omega} \\ &\le C\|u - u_h\|_{0,\Omega} |Q_H v|_{1,\Omega}, \end{aligned}$$

which together with the inverse inequality and the estimate $\|Q_H v\|_{0,\Omega} \le C\|v\|_{0,\Omega}$ yields

$$|(Q_H A\nabla(u - u_h), v)| \le Ch^2 H^{-1} \|Q_H v\|_{0,\Omega} \le Ch^2 H^{-1} \|v\|_{0,\Omega} \tag{3.18}$$

and

$$\|Q_H A\nabla(u - u_h)\|_{0,\Omega} \le Ch^2/H. \tag{3.19}$$

Using the triangular inequality, (3.19), and

$$\|A\nabla u - Q_H A\nabla u\|_{0,\Omega} \le CH^{1+\alpha},$$

we then arrive at

$$\begin{aligned} &\|A^{1/2}\nabla u - \mathcal{F} u_h\|_{0,\Omega} \\ &= \|A^{1/2}\nabla u - A^{-1/2} Q_H A\nabla u_h\|_{0,\Omega} \le \|A^{-1/2}(A\nabla u - Q_H A\nabla u_h)\|_{0,\Omega} \\ &\le C\left(\|A^{-1/2}(A\nabla u - Q_H A\nabla u)\|_{0,\Omega} + \|A^{-1/2} Q_H A\nabla(u - u_h)\|_{0,\Omega}\right) \\ &\le C(H^{1+\alpha} + h^2/H), \end{aligned}$$

which is nothing but (3.16). Thus we obtain (3.17) from (3.2) and (3.16). □

REMARK 3.4. One sees that the optimal choice of size of the coarser mesh should be

$$H = O(h^{2/(2+\alpha)}).$$

If we take this choice, then we have approximations with higher accuracy, which are estimated as follows:

$$\|A\nabla u - Q_H A\nabla u_h\|_{0,\Omega} + \|A^{1/2}\nabla u - A^{-1/2}Q_H A\nabla u_h\|_{0,\Omega} \le Ch^{1+\alpha/(2+\alpha)}$$

and

$$|\lambda_h^{\mathcal{F}} - \lambda| \le Ch^{2+\alpha/(2+\alpha)}.$$

We may also get some highly accurate approximations from using the L^2-projection operator $\tilde{Q}_H$.

THEOREM 3.4. [illegible] $u \in H_0^1(\Omega) \cap H^{2+\alpha}(\Omega)$ [illegible] $\mathcal{F} = A^{1/2}\nabla\tilde{Q}_H$, [illegible]

$$\|A^{1/2}\nabla u - \mathcal{F}u_h\|_{0,\Omega} \le C(H^{1+\alpha} + h^2/H), \tag{3.20}$$

$$|\lambda_h^{\mathcal{F}} - \lambda| \le C(H^{1+\alpha} + h)(H^{1+\alpha} + h^2/H). \tag{3.21}$$

[illegible] Using the inverse inequality and the estimates

$$\|u - \tilde{Q}_H u\|_{a,\Omega} \le CH^{1+\alpha},$$
$$\|\tilde{Q}_H w\|_{0,\Omega} \le C\|w\|_{0,\Omega} \quad \forall w \in L^2(\Omega),$$

we obtain the following estimation:

$$\begin{aligned}\|u - \tilde{Q}_H u_h\|_{a,\Omega} &\le \|u - \tilde{Q}_H u\|_{a,\Omega} + \|\tilde{Q}_H(u - u_h)\|_{a,\Omega}\\ &\le CH^{1+\alpha} + CH^{-1}\|\tilde{Q}_H(u - u_h)\|_{0,\Omega}\\ &\le CH^{1+\alpha} + CH^{-1}\|u - u_h\|_{0,\Omega} \le C(H^{1+\alpha} + h^2/H),\end{aligned}$$

which is nothing but (3.20). Hence we obtain (3.21) from (3.2) and (3.20). □

[illegible] We may use another postprocess to construct highly accurate approximations for eigenvalues. If, for instance,

$$\tilde{\lambda}_h = \frac{\|\tilde{Q}_H u_h\|_{a,\Omega}^2}{\|\tilde{Q}_H u_h\|_{0,\Omega}^2},$$

then

$$0 < \tilde{\lambda}_h - \lambda \le C(H^{2(1+\alpha)} + h^4/H^2). \tag{3.22}$$

It should be pointed out that our defect correction schemes based on L^2-projection averaging techniques are different from that proposed in [37]. It is noted that the defect correction scheme on a shape-regular mesh proposed in this paper is first to solve the eigenvalue problem on the fine mesh and then to do some postprocessing through an L^2-projection averaging on some coarser mesh, while the scheme provided in [37] is first to solve the eigenvalue problem on some coarser mesh and then to correct the residual by solving some boundary value problem on the fine mesh.

3.3. Numerical experiments. In this subsection we will report several numerical results of highly accurate approximations of finite element eigenvalues constructed by using our averaging techniques and the defect correction procedures. Examples in two and three dimensions are provided. These results support our theory established above.

TABLE 1
Example 3.1*: Global averaging on uniform meshes.*

h	$\eta(Q_h,h)$	order($\eta(Q_h,h)$)	$\tilde{\lambda}_{h,Q_h}$	order($e(Q_h,h)$)
2^{-2}	7.7993e-1		19.911672	
2^{-3}	2.48623e-1	1.6494	19.771539	2.4153
2^{-4}	7.95558e-2	1.6439	19.743630	2.8704
2^{-5}	2.6198e-2	1.5312	19.739774	2.9676
2^{-6}	8.8756e-3	1.5615	19.739280	2.9889
2^{-7}	3.06537e-3	1.5338	19.739218	2.9525
2^{-8}	1.07054e-3	1.5177	19.739210	2.9409

TABLE 2
Example 3.1*: Local averaging on uniform meshes.*

h	$\eta(G_h,h)$	order($\eta(G_h,h)$)	$\tilde{\lambda}_{h,G_h}$	order($e(G_h,h)$)
2^{-2}	1.19859		19.268480	
2^{-3}	4.41226e-1	1.4417	19.680152	2.9949
2^{-4}	1.5337e-1	1.5245	19.732977	3.2462
2^{-5}	5.30632e-2	1.5312	19.738572	3.3088
2^{-6}	1.84894e-2	1.5210	19.739141	3.4120
2^{-7}	6.48318e-3	1.5119	19.739201	3.1194
2^{-8}	2.28212e-3	1.5063	19.739208	3.2819

Let (u_h,λ_h) be an approximation to (u,λ) and define

$$\begin{aligned}
&\eta(Q_h,h) = \|A^{1/2}\nabla u_h - A^{-1/2}Q_hA\nabla u_h\|_{0,\Omega},\\
&\tilde{\lambda}_{h,Q_h} = \lambda_h - \|A^{1/2}\nabla u_h - A^{-1/2}Q_hA\nabla u_h\|^2_{0,\Omega},\\
&e(Q_h,h) = |\tilde{\lambda}_{h,Q_h} - \lambda|,\\
&\eta(G_h,h) = \|A^{1/2}\nabla u_h - A^{-1/2}\mathcal{C}_hAu_h\|_{0,\Omega},\\
&\tilde{\lambda}_{h,G_h} = \lambda_h - \|A^{1/2}\nabla u_h - A^{-1/2}\mathcal{C}_hAu_h\|^2_{0,\Omega},\\
&e(G_h,h) = |\tilde{\lambda}_{h,G} - \lambda|,\\
&\tilde{\lambda}_{h,Q_H} = \lambda_h - \|A^{1/2}\nabla u_h - A^{-1/2}Q_HA\nabla u_h\|^2_{0,\Omega},\\
&\tilde{\lambda}_{h,\tilde{Q}_H} = \frac{\|\tilde{Q}_Hu_h\|^2_{a,\Omega}}{\|\tilde{Q}_Hu_h\|^2_{0,\Omega}}.
\end{aligned}$$

EXAMPLE 3.1. Consider a model problem in two dimensions:

$$\begin{cases} -\Delta u &= \lambda u \quad \text{in } \Omega,\\ u &= 0 \quad \text{on } \partial\Omega, \end{cases} \tag{3.23}$$

where $\Omega = (0,1)\times(0,1)$. The eigenvalues of (3.23) are given by $\lambda_{k,l} = (k^2+l^2)\pi^2$, and the associated eigenfunctions are $u_{k,l} = \sin(k\pi(x-1))\sin(l\pi(y-1))$, $k,l = 1,2,\ldots$.

Here we address the first eigenvalue $\lambda \simeq 19.7392$ and its associated eigenfunction $u = \sin(\pi(x-1))\sin(\pi(y-1))$. Note that α in (2.13) or (2.14) is 1; the numerical results in Tables 1 and 2 coincide with our theory, while the numerical results in Tables 3 and 4 are better than those predicted by our theory.

Note that in Tables 3 and 4, the optimal choices of the mesh sizes are $H = O(h^{0.6667})$ (i.e., $\ln H/\ln h \simeq 0.6667$), and the theoretical convergence order in Tables 3 and 4 is 2.3333 and 2.6667, respectively.

TABLE 3
Example 3.1*: Global average on irregular meshes (Q_H projection).*

H	h	$\ln H/\ln h$	$\tilde{\lambda}_{h,Q_H}-\lambda$	order$(\tilde{\lambda}_{h,Q_H}-\lambda)$
3.535534e-01	1.767767e-01	6.000000e-01	-8.096618e-02	
1.767767e-01	8.838835e-02	7.142857e-01	-4.528332e-03	4.117053e+00
8.838835e-02	3.125000e-02	7.000000e-01	-2.192194e-04	2.912355e+00
4.419417e-02	1.5622643e-02	7.6255081e-01	-1.622172e-05	3.755559e+00

TABLE 4
Example 3.1*: Global average on irregular meshes ($\tilde{Q}_H$ projection.)*

H	h	$\ln H/\ln h$	$\tilde{\lambda}_{h,\tilde{Q}_H}-\lambda$	order$(\tilde{\lambda}_{h,\tilde{Q}_H}-\lambda)$
3.535534e-01	1.767767e-01	6.000000e-01	2.159374e-01	
2.357023e-01	8.333333e-02	5.815856e-01	1.928435e-02	3.212194e+00
1.767767e-01	6.250000e-02	6.250000e-01	9.470608e-03	3.007341e+00
7.071068e-02	5.000000e-02	8.843109e-01	5.125633e-03	2.751319e+00

Example 3.2. Consider a model problem in three dimensions:

$$\begin{cases} -\sum_{i=1}^{3}\dfrac{\partial}{\partial x_i}\left(x_i^2\dfrac{\partial u}{\partial x_i}\right) = \lambda u & \text{in } \Omega, \\ u = 0 & \text{on } \partial\Omega, \end{cases} \tag{3.24}$$

where $\Omega=(1,3)\times(1,2)\times(1,2)$. The first eigenvalue is $\lambda=\frac{3}{4}+(\frac{2}{\ln^2 2}+\frac{1}{\ln^2 3})\pi^2\simeq 50.01212422$, and the associated eigenfunction is $u=\prod_{i=1}^{3}(x_i^{-\frac{1}{2}}\sin(\frac{\pi\ln x_i}{\ln\beta_i}))$, where $\beta_1=3$, $\beta_2=\beta_3=2$.

Some numerical results on uniform meshes are presented in Tables 5 and 6. In this example, the value α in (2.13) or (2.14) is 1. Hence our numerical results show a better order of convergence for eigenfunctions than those predicted by our theory. The numerical results on irregular meshes are presented in Tables 7 and 8. These results are comparable to those predicted by our theory. Note that the optimal choices of mesh sizes in Tables 7 and 8 are $H=O(h^{0.6667})$ (i.e., $\ln H/\ln h\simeq 0.6667$), and the theoretical convergence order in Tables 7 and 8 is 2.3333 and 2.6667, respectively.

4. Applications to quantum chemistry. For highly accurate finite element approximations to elliptic eigenvalue problems, theoretically we need to require that the exact solution and the coefficients are smooth. In practice, however, we have finite element approximations with higer accuracy even if the data are not so smooth. In this section, as illustrations, we shall apply the finite element defect correction scheme together with the averaging techniques to construct highly accurate approximations for two simplest Hartree–Fock equations in computational quantum chemistry: one Hartree–Fock equation is for hydrogen atoms and the other is for helium atoms. The data in such Hartree–Fock equations are not smooth. More precisely, the coefficients and the exact solutions of the Hartree–Fock equations are singular. However, it is shown by our numerical experiments that higher accuracy approximations for Hartree–Fock equations may be achieved. Particularly, approximations with much higher accuracy can be obtained if some adaptive finite element grids are employed. (As for the generalization of adaptive finite element grids for eigenvalue problems, we refer to [26] for details.) In our computation, the adaptive finite element grids that

TABLE 5
Example 3.2*: Global averaging on uniform meshes.*

h	$\eta(Q_h,h)$	order($\eta(Q_h,h)$)	$\tilde{\lambda}_{h,Q_h}$	order($e(Q_h,h)$)
2^{-2}	1.3652	1.6068	50.7875	
2^{-3}	4.48244e-1	1.6068	50.1327	2.6837
2^{-4}	1.43955e-1	1.6387	50.0273	2.9798
2^{-5}	4.71814e-2	1.6093	50.0138	3.0875

TABLE 6
Example 3.2*: Local averaging on uniform meshes.*

h	$\eta(G_h,h)$	order($\eta(G_h,h)$)	$\tilde{\lambda}_{h,G_h}$	order($e(G_h,h)$)
2^{-2}	1.6819		49.1804	
2^{-3}	6.25215e-1	1.4277	49.9034	2.9369
2^{-4}	1.43955e-1	1.5549	49.9985	3.0080
2^{-5}	7.20579e-2	1.5622	50.0102	2.9069

TABLE 7
Example 3.2*: Global averaging on irregular meshes (Q_H projection).*

H	h	$\ln H/\ln h$	$\tilde{\lambda}_{h,Q_H}-\lambda$	order($\tilde{\lambda}_{h,Q_H}-\lambda$)
7.250000e-01	5.250000e-01	4.990768e-01	-2.319297e+00	
4.403479e-01	2.625000e-01	6.132244e-01	-2.683262e-01	2.907006e+00
2.201739e-01	1.312500e-01	7.452473e-01	-1.255236e-02	4.417957e+00
1.732051e-01	7.071068e-02	5.059121e-01	-2.086580e-03	2.901150e+00

TABLE 8
Example 3.2*: Global averaging on irregular meshes ($\tilde{Q}_H$ projection).*

H	h	$\ln H/\ln h$	$\tilde{\lambda}_{h,\tilde{Q}_H}-\lambda$	order($\tilde{\lambda}_{h,\tilde{Q}_H}-\lambda$)
7.250000e-01	5.250000e-01	4.990768e-01	1.526152e+01	
4.403479e-01	2.625000e-01	6.132244e-01	1.712164e+00	3.156006e+00
2.201739e-01	1.119780e-01	6.911942e-01	2.308108e-01	2.352155e+00
1.237179e-01	5.050763e-02	6.999362e-01	3.315419e-02	2.437180e+00

will be used are constructed from the error indicator

$$\sum_{\tau\in T^h} \|A^{1/2}\nabla u_h - \mathcal{F}u_h\|_{0,\tau}^2$$

and the bisection approaches, where $\mathcal{F}$ is a recovery operator.

4.1. Hydrogen atoms. The Hartree–Fock equation for hydrogen atoms is written as

$$-\frac{1}{2}\Delta u - \frac{1}{|x|}u = Eu \qquad \text{in } R^3 \tag{4.1}$$

with $\int_{R^3}|u|^2dx = 1$. The total energy of the ground state for a hydrogen atom is the first eigenvalue E of (4.1). In the simulation, we solve (4.1) by using the finite element discretization (2.8) when

$$A = \begin{pmatrix} 1/2 & 0 & 0 \\ 0 & 1/2 & 0 \\ 0 & 0 & 1/2 \end{pmatrix}, \tag{4.2}$$

TABLE 9
H atoms, uniform meshes, and defect correction.

Degree of freedom	Energy	energy(Q_h)	energy(G_h)
3375	-0.431732	-4.779223e-01	-4.867663e-01
29791	-0.472074	-4.939226e-01	-4.972154e-01
250047	-0.498744	-4.987437e-01	-4.995358e-01

TABLE 10
H atoms, adaptive meshes, and defect correction.

Degree of freedom	Energy	energy(Q_h)	energy(G_h)
5483	-0.483779	-4.988560e-01	-4.999287e-01
13019	-0.492387	-4.995934e-01	-4.998503e-01

TABLE 11
H atoms.

Method	Reference	Degree of freedom	Energy
Standard FEM	[35]	538,312	-0.499967
Standard FEM	[18]	$3,125^*$	-0.499997

** Here the 5-order polynomial basis functions are used.*

$\beta = -1/|x|$, $\Omega = (-6.4, 6.4) \times (-6.4, 6.4) \times (-6.4, 6.4)$, and $\lambda = E$. Further, we apply $\mathcal{F} = \sqrt{1/2}Q_h\nabla$ and $\mathcal{F} = \sqrt{1/2}\mathcal{C}_h\nabla$ in our defect correction scheme, respectively. The numerical results are shown in Tables 9 and 10, where

$$\text{energy}(Q_h) = E_h - \|\sqrt{1/2}\nabla u_h - \sqrt{1/2}Q_h\nabla u_h\|^2_{0,\Omega}, \tag{4.3}$$

$$\text{energy}(G_h) = E_h - \|\sqrt{1/2}\nabla u_h - \sqrt{1/2}\mathcal{C}_h\nabla u_h\|^2_{0,\Omega}. \tag{4.4}$$

As compared with the results in the literature (see Table 11), one sees that our approaches are powerful.

4.2. Helium atoms. For the helium atom, the associated Hartree–Fock equation is

$$-\frac{1}{2}\Delta u - \frac{2}{|x|}u + \int_{R^3}\frac{|u(y)|^2dy}{|x-y|}u = Eu \ \text{ in } R^3 \tag{4.5}$$

with $\int_{R^3}|u|^2dx = 1$. In our computation for the total energy of the ground state, we solve the following nonlinear eigenvalue problem: Find $(u, E) \in H_0^1(\Omega) \times R$ such that $\int_\Omega |u|^2dx = 1$ and

$$\left\{\begin{array}{rcl} -\frac{1}{2}\Delta u - \frac{2}{|x|}u + \int_\Omega \frac{|u(y)|^2dy}{|x-y|}u &=& Eu \ \text{ in } \Omega, \\ u &=& 0 \ \text{ on } \partial\Omega, \end{array}\right. \tag{4.6}$$

where $\Omega = (-7.0, 7.0) \times (-7.0, 7.0) \times (-7.0, 7.0)$. The total energy of the ground state for the helium atom is

$$\mathcal{E} = 2E - \int_\Omega\int_\Omega \frac{|u(x)|^2|u(y)|^2dxdy}{|x-y|},$$

where E is the first eigenvalue of (4.5). Since it is nonlinear, we solve (4.6) self-consistently:

TABLE 12
He atoms.

Equation	Method	Reference	Degree of freedom	Energy
Schrödinger	Standard FEM	[29]	$6,720^*$	-2.90289
Schrödinger	Standard FEM	[10]	unknown	-2.9036118
Hartree–Fock	Standard FEM	[35]	$500,000^{**}$	-2.8522

* *Here special basis functions and special grids are used.*
** *Here the products of orthogonal or nonorthogonal 1D finite element shape functions are used to form 3D basis functions.*

TABLE 13
He atoms, uniform meshes, and defect correction.

Degree of freedom	Energy	energy(Q_h)	energy(G_h)
343	-1.967724	-2.493541e+00	-2.587672e+00
1331	-2.166529	-2.608688e+00	-2.694204e+00
3375	-2.318735	-2.686435e+00	-2.758896e+00
29791	-2.613051	-2.810009e+00	-2.845223e+00

TABLE 14
He atoms, adaptive meshes, and defect correction.

Degree of freedom	Energy	energy(Q_h)	energy(G_h)
1341	-2.680944	-2.827951e+00	-2.847466e+00
2975	-2.752170	-2.841256e+00	-2.848725e+00
16691	-2.830592	-2.861307e+00	-2.862088e+00
49573	-2.847419	-2.864203e+00	-2.862788e+00

- Step 1. Given an initial data (u_0, E_0).
- Step 2. For any $n = 0, 1, 2, \ldots$, find $(u_{n+1}, E_{n+1}) \in H_0^1(\Omega) \times R$ such that $\int_\Omega |u_{n+1}|^2 dx = 1$ and

$$\text{(4.7)} \quad \begin{cases} -\dfrac{1}{2}\Delta u_{n+1} - \dfrac{2}{|x|}u_{n+1} + \displaystyle\int_\Omega \frac{|u_n(y)|^2 dy}{|x-y|}u_{n+1} = E_{n+1}u_{n+1} \ \text{ in } \Omega, \\ \qquad\qquad\qquad\qquad\qquad\qquad u_{n+1} = 0 \ \text{ on } \partial\Omega. \end{cases}$$

For $n = 1, 2, \ldots$, we solve (4.7) by using the finite element discretization (2.8) when A is given by (4.2),

$$\beta = \frac{-2}{|x|} + \int_\Omega \frac{|u_n(y)|^2 dy}{|x-y|},$$

$\Omega = (-7.0, 7.0) \times (-7.0, 7.0) \times (-7.0, 7.0)$, and $\lambda = E_{n+1}$. Similarly, we apply $\mathcal{F} = \sqrt{1/2}Q_h\nabla$ and $\mathcal{F} = \sqrt{1/2}\mathcal{C}_h\nabla$ in our defect correction scheme, respectively.

The relevant numerical results in the literature are provided in Table 12. The exact total energy may be viewed as -2.90 (cf. [10, 29]). Our numerical results are reported in Tables 13 and 14, where energy(Q_h) and energy(G_h) are defined by (4.3) and (4.4), respectively. It is shown again that our approaches are efficient. For instance, by using our defect correction scheme and the adaptive mesh, we may get an approximate total energy -2.849 with only 2975 degrees of freedom, which can be compared with results using the standard finite element method with special basis functions, whose number of degrees of freedom is 5×10^5.

Acknowledgments. The authors would like to thank Professor X. Gang and Mr. D. Zhang for their discussions of electronic structure computations that moti-

vated the application in section 4 above, and to thank the referees for their valuable comments and suggestions that improved the presentation of this paper.

REFERENCES

[1] R. A. Adams, *Sobolev Spaces*, Academic Press, New York, 1975.

[2] M. Ainsworth and J. T. Oden, *A Posteriori Error Estimation in Finite Element Analysis*, Wiley, New York, 2000.

[3] I. Babuska and J. E. Osborn, *Finite element-Galerkin approximation for the eigenvalues and eigenvectors of selfadjoint problems*, Math. Comput., 52 (1989), pp. 275–297.

[4] I. Babuska and J. E. Osborn, *Eigenvalue problems*, in Handbook of Numerical Analysis, Vol. II, Finite Element Methods Part I, P. G. Ciarlet and J. L. Lions, eds., Elsevier, New York, 1991, pp. 641–792.

[5] R. E. Bank and J. Xu, *Asymptotically exact a posteriori error estimators, Part* I: *Grids with superconvergence*, SIAM J. Numer. Anal., 41 (2003), pp. 2294–2312.

[6] R. E. Bank and J. Xu, *Asymptotically exact a posteriori error estimators, Part* II: *General unstructured grids*, SIAM J. Numer. Anal., 41 (2003), pp. 2313–2332.

[7] T. Blacker and T. Belychko, *Superconvergent path recovery with equilibrium and conjoint interpolant enhancements*, Int. J. Numer. Methods Engrg., 37 (1994), pp. 517–536.

[8] J. H. Bramble and J. E. Osborn, *Rate of convergence estimates for nonselfadjoint eigenvalue approximations*, Math. Comput., 27 (1973), pp. 525–549.

[9] J. Brandts and M. Křížek, *Gradient superconvergence on uniform simplicial partitions of polytopes*, IMA J. Numer. Anal., 33 (2003), pp. 1–17.

[10] M. Braun, W. Schweizer, and H. Herold, *Finite-element calculations for the S states of helium*, Phys. Rev. A, 48 (1993), pp. 1916–1920.

[11] C. Carstensen, *Quasi-interpolation and a posteriori analysis in finite element methods*, M^2AN Math. Model. Numer. Anal., 33 (1999), pp. 1187–1202.

[12] C. Carstensen and S. Bartels, *Each averaging technique yields reliable a posteriori error control in FEM on unstructured grids, Part* I: *Low order conforming, nonconforming, and mixed FEM*, Math. Comput., 71 (2002), pp. 945–969.

[13] F. Chatelin, *Spectral Approximations of Linear Operators*, Academic Press, New York, 1983.

[14] C. M. Chen, *Optimal points of stresses for tetrahedron linear element,* Natur. Sci. J. Xiangtan Univ., 3 (1980), pp. 16–24 (in Chinese).

[15] P. G. Ciarlet, *The Finite Element Method for Elliptic Problems*, North–Holland, Amsterdam, New York, Oxford, 1978.

[16] P. Clément, *Approximation by finite element functions using local regularization*, RAIRO Anal. Numér., 9 (1975), pp. 77–84.

[17] G. Goodsell, *Pointwise superconvergence of the gradient for the linear tetrahedral element,* Numer. Methods Partial Differential Equations, 10 (1994), pp. 651–666.

[18] S. Hakel, D. Heinemann, D. Kolb, and B. Fricke, *Calculations of the polycentric linear molecule H_3^{2+} with the finite element method*, Chem. Phys. Lett., 206 (1993), pp. 91–95.

[19] B.-O. Heimsund, X.-C. Tai, and J. Wang, *Superconvergence for the gradient of finite element approximations by L^2 projections*, SIAM J. Numer. Anal., 40 (2002), pp. 1263–1280.

[20] W. Hoffmann, A. H. Schatz, L. B. Wahlbin, and G. Wittum, *Asymptotically exact a posteriori estimators for the pointwise gradient error on each element in irregular meshes* I: *A smooth problem and globally quasi-uniform meshes,* Math. Comp., 70 (2001), pp. 897–909.

[21] V. K. Kantchev and R. D. Lazarov, *Superconvergence of the gradient of linear finite elements for* 3*-D Poisson equation*, in Proceedings of the Conference on Optimal Algorithms, B. Sendov, ed., Publishing House BAS, Sofia, Bulgaria, 1986, pp. 172–182.

[22] M. Křížek and P. Neittaanmäki, *Superconvergence phenomenon in the finite element method arising from averaging gradients*, Numer. Math., 45 (1984), pp. 105–116.

[23] M. Křížek, P. Neittaanmäki, and R. Stenberg, *Finite Element Methods: Superconvergence, Post-processing, and A Posteriori Estimates*, Lectures Notes in Pure and Appl. Math. 196, Marcel Dekker, New York, 1998.

[24] N. Levine, *Superconvergent recovery of the gradient from piecewise linear finite element approximations,* IMA J. Numer. Anal., 5 (1985), pp. 407–427.

[25] B. Li and Z. Zhang, *Analysis of a class of superconvergence patch recovery techniques for linear and bilinear finite elements*, Numer. Methods Partial Differential Equations, 15 (1999), pp. 151–167.

[26] D. Mao, L. Shen, and A. Zhou, *Adaptive finite element algorithms for eigenvalue problems based on local averaging type a posteriori error estimates*, Adv. Comput. Math., in press.

[27] R. McWeeny, *Methods of Molecular Quantum Mechanics*, Academic Press, New York, 1992.
[28] R. G. Parr and W. Yang, *Density-Functional Theory of Atoms and Molecules*, Oxford University Press, Oxford, UK, 1989.
[29] A. Scrinzi, *A 3-dimensional finite elements procedure for quantum mechanical applications*, Comput. Phys. Comm., 86 (1995), pp. 67–80.
[30] B. Y. Tong and L. J. Sham, *Application of a self-consistent schemes including exchange and correlation effects to atoms*, Phys. Rev., 144 (1966), pp. 1–4.
[31] M. J. Todd, *The Computation of Fixed Points and Applications*, Springer-Verlag, Berlin, 1976.
[32] L. B. Wahlbin, *Superconvergence in Galerkin Finite Element Methods,* Lecture Notes in Math. 1605, Springer, Berlin, 1995.
[33] J. Wang, *A superconvergence analysis for finite element solutions by the least-squares surface fitting on irregular meshes for smooth problems*, J. Math. Study, 33 (2000), pp. 229–243.
[34] M. F. Wheeler and J. R. Whiteman, *Superconvergence recovery of gradients on subdomains from piecewise linear finite element approximations*, Numer. Methods Partial Differential Equations, 3 (1987), pp. 65–82.
[35] S. R. White, J. W. Wilkins, and M. P. Teter, *Finite-element method for electronic structure*, Phys. Rev. B, 39 (1989), pp. 5819–5833.
[36] J. Xu and Z. Zhang, *Analysis of recovery type a posteriori error estimators for mildly structured grids*, Math. Comput., 73 (2004), pp. 1139–1152.
[37] J. Xu and A. Zhou, *A two-grid discretization scheme for eigenvalue problems*, Math. Comput., 70 (2001), pp. 17–25.
[38] N. Yan and A. Zhou, *Gradient recovery type a posteriori error estimates for finite element approximations on irregular meshes*, Comput. Methods Appl. Mech. Engrg., 190 (2001), pp. 4289–4299.
[39] Z. Zhang, *Ultraconvergence of the patch recovery technique* II, Math. Comp., 69 (2000), pp. 141–158.
[40] Z. Zhang and A. Naga, *A new finite element gradient recovery method: Superconvergence property*, SIAM J. Sci. Comput., 26 (2005), pp. 1192–1213.
[41] Z. Zhang and H. D. Victory, Jr., *Mathematical analysis of Zienkiewicz-Zhu's derivative patch recovery technique for quadrilateral finite elements*, Numer. Methods Partial Differential Equations, 12 (1996), pp. 507–524.
[42] A. Zhou, *Numerical Analysis of Finite Element Methods for Hyperbolic Systems*, Ph.D. Thesis, Institute of Systems Science, Chinese Academy of Sciences, Beijing, R.O.C., 1990.
[43] A. Zhou and Q. Lin,*Optimal and superconvergence estimates of the finite element method for a scale hyperbolic equation*, Acta. Math. Sci., 14 (1994), pp. 90–94.
[44] Q. Zhu and Q. Lin, *Superconvergence Theory of Finite Element Methods*, Hunan Science Press, Changsha, R.O.C., 1989 (in Chinese).
[45] O. C. Zienkiewicz and J. Z. Zhu, *The superconvergence patch recovery and a posteriori error estimates*, Int. J. Numer. Methods Engrg., 33 (1992), pp. 1331–1382.
[46] O. C. Zienkiewicz and J. Z. Zhu, *The superconvergence patch recovery (SPR) and adaptive finite element refinement*, Comput. Methods Appl. Mech. Engrg., 101 (1992), pp. 207–224.

SIAM J. SCI. COMPUT.
Vol. 28, No. 1, pp. 339–358
© 2006 Society for Industrial and Applied Mathematics

ON EFFECTIVE METHODS FOR IMPLICIT PIECEWISE SMOOTH SURFACE RECOVERY*

U. M. ASCHER†, E. HABER‡, AND H. HUANG§

Abstract. This paper considers the problem of reconstructing a piecewise smooth model function from given, measured data. The data are compared to a field which is given as a possibly nonlinear function of the model. A regularization functional is added which incorporates the a priori knowledge that the model function is piecewise smooth and may contain jump discontinuities. Regularization operators related to total variation (TV) are therefore employed.

Two popular methods are modified TV and Huber's function. Both contain a parameter which must be selected. The Huber variant provides a more natural approach for selecting its parameter, and we use this to propose a scheme for both methods. Our selected parameter depends both on the resolution and on the model average roughness; thus, it is determined adaptively. Its variation from one iteration to the next yields additional information about the progress of the regularization process.

The modified TV operator has a smoother generating function; nonetheless we obtain a Huber variant with comparable, and occasionally better, performance.

For large problems (e.g., high resolution) the resulting reconstruction algorithms can be tediously slow. We propose two mechanisms to improve efficiency. The first is a multilevel continuation approach aimed mainly at obtaining a cheap yet good estimate for the regularization parameter and the solution. The second is a special multigrid preconditioner for the conjugate gradient algorithm used to solve the linearized systems encountered in the procedures for recovering the model function.

Key words. inverse problem, total variation, Huber regularization, lagged diffusivity, Gauss–Newton, multilevel continuation, multigrid preconditioner

AMS subject classification. 65N21

DOI. 10.1137/040617261

1. Introduction.

We consider the problem of reconstructing a piecewise smooth model function in several dimensions from given, measured data. In general, the data are viewed as a nonlinear function of the model:

$$b = F(m) + \epsilon,$$

where F is the forward modeling operator, $m = m(\mathbf{x})$ is the model to be recovered, b is the data, and ϵ is measurement noise. We assume that the model is defined on a two-dimensional (2D) domain with unit volume, Ω, although extensions to 3D are possible and in principle straightforward.

The operators F under consideration vary considerably from an identity in simple denoising problems [34, 38, 35] through a linear, constant coefficient form in deblurring and tomography problems [38, 24, 23, 16] to the solution of a diffusive partial differential equation (PDE) system involving Poisson or Maxwell's differential equations

*Received by the editors October 20, 2004; accepted for publication (in revised form) October 11, 2005; published electronically March 17, 2006.
http://www.siam.org/journals/sisc/28-1/61726.html

†Department of Computer Science, University of British Columbia, Vancouver, BC, V6T 1Z4, Canada (ascher@cs.ubc.ca). This author's work was supported in part under NSERC research grant 84306.

‡Department of Mathematics and Computer Science, Emory University, Atlanta, GA 30322 (haber@mathcs.emory.edu).

§Institute of Applied Mathematics, University of British Columbia, Vancouver, BC, V6T 1Z2, Canada (hhzhiyan@math.ubc.ca). This author's work was supported in part under NSERC research grant 84306.

[10, 28, 5, 22]. Due to either ill-posedness of the problem or the presence of noise in the measured data (or both) the problem requires regularization. In a Tikhonov-type approach this leads to the optimization problem

$$\min_m T(m) \equiv \frac{1}{2}\|F(m) - b\|^2 + \beta R(m), \tag{1}$$

where $R(m)$ is the regularization operator which is the focus of attention in the present work, and $\beta \geq 0$ is the regularization parameter. We use the least squares norm for the data fitting term, although other norms are possible and may be more suitable depending on the noise statistics.

For simplicity of exposition we employ the following notational convention. The domain Ω is discretized by a uniform grid with N^2 square cells of length $h = 1/N$ each. With a slight abuse of notation the function $m(\mathbf{x})$ becomes an $(N+1)\times(N+1)$ grid function m which is occasionally further reshaped into a vector, $m \in \Re^K$, $K = (N+1)^2$. Likewise, the data b, given on a subset of the grid's nodes or cells, is occasionally rearranged as a vector, $b \in \Re^M$. Then $F\colon \Re^K \to \Re^M$ is a vector function, assumed continuously differentiable. And yet, we shall also denote the discretized gradient of m, say, simply as ∇m. Finally, we define the

$$J(m) = \frac{\partial F}{\partial m}.$$

Then the necessary conditions for optimum in (1) are written as a system of generally nonlinear algebraic equations,

$$J^T(F - b) + \beta R_m = 0. \tag{2}$$

For the regularization term, one typically considers a same-grid discretization of

$$R(m) = \int_\Omega \rho(|\nabla(m - m_{ref})|) + \hat{\alpha}(m - m_{ref})^2, \tag{3}$$

where $\hat{\alpha} \geq 0$ is a small parameter and m_{ref} is a given reference function. The selection of m_{ref} can be crucial, for instance in geophysical exploration, but here we set $\nabla m_{ref} = 0$ and concentrate on the choice of the function ρ. The latter relates directly to the a priori information we have about the smoothness of the model. Note that

$$R_m(m) = -\nabla \cdot (\sigma(|\nabla m|)\nabla m), \quad \text{where} \quad \sigma(\tau) = \frac{\rho'(\tau)}{\tau}.$$

For recovering smooth surfaces it is common to choose least squares, $\rho(\tau) = \frac{1}{2}\tau^2$, yielding $\sigma \equiv 1$. However, this choice is inadequate in the presence of discontinuities. The famous total variation (TV) choice (e.g. [11, 32, 38, 18]) uses the l_1-norm instead, $\rho(\tau) = \tau$, implying $\sigma(\tau) = 1/\tau$. Unfortunately, $\sigma(\tau) = 1/\tau$ is unbounded when τ shrinks, i.e., when $|\nabla m| \to 0$. Three remedies have been proposed in the literature to alleviate this. The first, and most recent, is to view $|\nabla m|$ as dual variables in a primal-dual approach. See [6] and references therein. This leads to some beautiful, more complex mathematical analysis. However, the other two possibilities to be explored below are so simple, [illegible], popular and effective in practice that the case for their replacement is necessarily not crucial. Moreover, the adaptive variants proposed in this article often yield better results than the pure TV would.

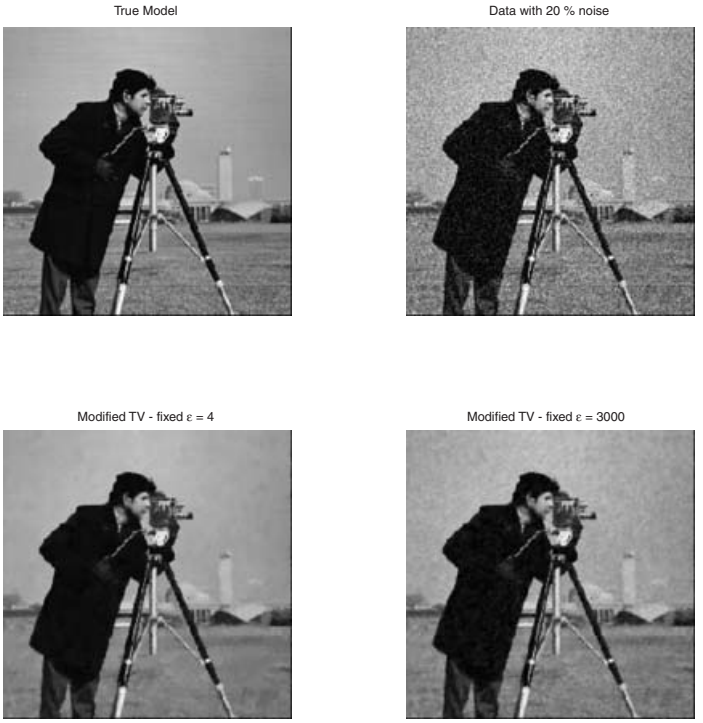


FIG. 1. *Denoising: The "true image" (upper left) is corrupted by adding* 20% *white noise (upper right) and the latter is fed as data b to a denoising program using the usual Tikhonov regularization with modified TV. The smaller value of switching parameter* ε *(lower left) produces a more pleasing result than the larger value (lower right). Taking* $\varepsilon = .01$ *or* $\varepsilon = 10^{-6}$ *yields results similar to those displayed in the lower left panel but at a higher cost.*

The [illegible] approach replaces $|\nabla m|$ in ρ and therefore in σ by

$$|\nabla m|_\varepsilon = \sqrt{m_x^2 + m_y^2 + \varepsilon^2}. \tag{4}$$

See, e.g., [9, 7, 38]. The parameter $\varepsilon \geq 0$ must be selected to be small enough so as to sharply preserve discontinuities and large enough so as not to cause undue numerical hardship, especially when iterative methods such as preconditioned conjugate gradients are used. We have found surprisingly little in the literature regarding the practical selection of this parameter.

Experiment 1. To see that there is an issue here, consider the popular example of denoising the picture given in the upper right panel of Figure 1 using a suitably modified TV regularization [32, 9]. For our grid function $m = \{m_{i,j}\}_{i,j=0}^{N}$ ($N = 256$, $h = 1/N$), denoting

$$D_{+,x} m_{i,j} = \frac{m_{i+1,j} - m_{i,j}}{h}$$

and similarly for $D_{+,y} m_{i,j}$, we define $|\nabla m_{i,j}|_h$, etc., by

$$|\nabla m|_\varepsilon = |\nabla m|_{h,\varepsilon} = \sqrt{(D_{+,x} m)^2 + (D_{+,y} m)^2 + \varepsilon^2}.$$

TABLE 1
Work units and CPU time vs the modification parameter ε.

ε	10^{-6}	10^{-2}	4	3000
Work units	37	30	12	10
CPU sec	127	115	50	42

This yields a symmetric term in the sum replacing the integral for $R(m)$ in (3) which appears in (1). We next vary ε and record work units and CPU times: the actual method used is described later, in sections 3 and 4; here we merely note that an increase in computing time relates directly to an increase in the total number of preconditioner calls in our procedure for solving the resulting linear systems.

Table 1 clearly indicates that more computational effort is required for smaller values of ε. Next, the results displayed in Figure 1 indicate that the choice $\varepsilon = 3000$ is not very good, especially for segmentation purposes; the other three values yield very similar, and satisfactory, results.

The combination of quality and cost therefore indicates not only that (i) as is well-known, values of ε which are too large or too small should be avoided, but also that (ii) the best value among those displayed, $\varepsilon = 4$, does not automatically leap to mind! It should be noted that the performance indicators in Table 1 do not change significantly in the range $4 \le \varepsilon \le 100$. Thus, the quest is not to find an optimal value of ε, but rather, to select it from "the right range." This parameter range, however, need not necessarily include values $\varepsilon \ll 1$ which are commonly used by practitioners. Unless one is content with a highly expensive trial and error approach, a procedure is required that would yield an effective, automatic selection of ε.

An alternative to modified TV uses the Huber switching function [25, 18, 34]

$$\rho(\tau) = \begin{cases} \tau, & \tau \ge \gamma, \\ \tau^2/(2\gamma) + \gamma/2, & \tau < \gamma \end{cases} \tag{5}$$

$$R_m(m) \leftarrow \nabla \cdot \left(\min\left\{ \frac{1}{\gamma}, \frac{1}{|\nabla m|} \right\} \nabla m \right).$$

Unlike for the smooth modified TV function, here ρ is not C^2, although its second derivative is bounded. Nonetheless we show in section 2 and Appendix A that the lagged diffusivity algorithm converges globally under the same assumptions as for the modified TV, extending the proof of [9]. The advantage of the choice (5) is that the selection of the parameter γ is much more transparent, and we capitalize on this in the present article.

Indeed, γ in (5) is viewed as answering the question, "what can be interpreted as a jump?" Thus, it is the minimal size of a local change in $|\nabla m|$ which is still interpreted by a TV functional and hence jumps will not be necessarily smeared. The TV interpretation is desirable, particularly if edges are to be respected, but it must be avoided in an $O(h)$ band, where the resolution limitation does not allow a true distinction between what may be zero and what is not.

With this in mind, we propose the following [illegible]:

$$\gamma = h \int_\Omega |\nabla m|. \tag{6}$$

The integral is of course replaced by a sum upon discretization. If the volume of Ω differs from unity then we divide by it to obtain the average of $|\nabla m|$. In (6), $\gamma = \gamma(m)$

depends on the solution, and indeed we adjust its value through the iteration in an obvious fashion. What this choice basically says is that we avoid the possibility of a jump [illegible] $|\nabla m|$ [illegible] h [illegible]; otherwise, a jump [illegible] happen.

Note that the average value of $|\nabla m|$ may well depend on the application, its scaling, and the amount of noise still present in a given iteration. Moreover, the rate of change in γ from one iteration to the next indicates to what extent the discontinuities have already been distinguished from the smooth parts of the surface m to be recovered.

Our choice of γ also leads to a selection of the parameter ε in (4), as elaborated upon in section 2.

Another, earlier choice uses so-called "robust statistics theory" where γ is interpreted as answering the question of how large the model's gradient can be before it is considered to be an outlier. One arrives at the formula (see, e.g., [34, p. 231] and references therein)

$$\gamma = c \cdot \text{median}[|\nabla m - \text{median}(|\nabla m|)|], \tag{7}$$

with $c \approx 1.5$. However, it is not entirely clear under what conditions the statistical arguments leading to (7) are valid, nor is it clear how this formula scales with h. In fact, the rationale leading to (7), unlike ours in deriving (6), does not favor TV whenever possible, in recognition of the tendency of TV regularization to flatten natural scenes.

While the selection of γ and ε is settled, that of the regularization parameter β, which is notoriously challenging in practical problems even for least squares regularization [38, 17], does not become easier here. Several systems must be solved in a search for a good β, each involving difficult nonlinear optimization and the solution of often ill-conditioned linear systems. For large problems (corresponding to high resolution) the resulting reconstruction algorithms can become tediously slow.

In section 3 we propose two mechanisms to improve efficiency. The first is a multilevel continuation approach [1]. While applying continuation also in a procedure to grow the model solution gradually this technique is aimed mainly at obtaining a cheap yet good estimate for the regularization parameter β. The key observation is that β should not vary much between different grids with our choice of γ.

The second item in section 3 is a special multigrid preconditioner for the conjugate gradient algorithm used to solve the linearized systems encountered in the procedures employed for recovering the model function. Because of jumps in the coefficient function σ a usual multigrid cycle can be ineffective [37].

We close this paper with numerical experiments demonstrating the proposed techniques on several inverse problems. These techniques work very well for problems such as denoising, deblurring and emission tomography. But for a problem such as electrical impedance tomography (EIT) [5] or electromagnetic geophysical prospecting [22] the reconstruction of a piecewise continuous model is fundamentally difficult. The conclusion section seals the paper.

2. Modified TV and Huber regularization. Let us consider first the Huber regularization (5) and the choice of the parameter γ given by (6). It is not difficult to see that, despite the appearance of the term $1/\gamma$ in (5), the regularization term $R(m)$ remains bounded as $h \to 0$. Indeed, on the continuous level fix γ and let

$$\Phi = \{\mathbf{x} : |\nabla m(\mathbf{x})| < \gamma\}. \tag{8a}$$

Then (1) is written as

$$\min_m \frac{1}{2}\|F(m)-b\|^2 + \beta\left[\frac{1}{2\gamma}\int_\Phi |\nabla m|^2 + \frac{\gamma}{2}\int_\Phi 1 + \int_{\Omega\setminus\Phi}|\nabla m|\right]. \tag{8b}$$

But by (8a),

$$\frac{1}{2\gamma}\int_\Phi |\nabla m|^2 + \frac{\gamma}{2}\int_\Phi 1 < \gamma\int_\Phi 1,$$

and also $\int_\Phi |\nabla m| < \gamma\int_\Phi 1$. Hence, letting $\gamma\to 0$ we get $R(m)\to\int_\Omega|\nabla m|$.

Now, the regularization term on a grid with mesh width h approximates the continuous $R(m)$ up to $O(h)$ by straightforward consistency. We summarize this in Lemma 1.

LEMMA 1. [illegible] (6) [illegible]

$$R(m) = \int_\Omega |\nabla m| \quad + O(h). \tag{9}$$

Next, consider the modified TV regularization (4). When $\varepsilon \gg |\nabla m|$ we obviously get approximately the least squares norm on $|\nabla m|$. Thus, the considerations for selecting ε should follow similar lines to those for γ. We choose

$$\varepsilon = 0.5\gamma, \tag{10}$$

which yields agreement with (5) at $\tau = 0$, where these formulae are most useful if considered as modifications of TV. It is important to note that while both forms of stabilizing the TV seminorm are roughly equivalent the choice of the stabilization parameter is natural in the Huber case but is less obvious for the modified TV. If edges are not to be emphasized then the choice of a larger γ again has a more obvious meaning in the Huber case.

Note that as $h\to 0$, $|\nabla m|_{h,\varepsilon}\to|\nabla m|$ in a well-defined manner.

2.1. Lagged diffusivity. Upon differentiating (1) with respect to m we obtain the following system of equations:

$$\beta[L(m)m]_{i,j} + [J^T(m)(F(m)-b)]_{i,j} = 0, \quad \text{where} \tag{11a}$$

$$(L(m)m)_{i,j} = \frac{1}{h^2}[\sigma_{i+1/2,j}(m_{i,j}-m_{i+1,j}) + \sigma_{i,j+1/2}(m_{i,j}-m_{i,j+1})$$

$$+ \sigma_{i-1/2,j}(m_{i,j}-m_{i-1,j}) + \sigma_{i,j-1/2}(m_{i,j}-m_{i,j-1})], \tag{11b}$$

$$\sigma_{i+1/2,j} = \min\left\{\frac{1}{\gamma}, \frac{h}{\sqrt{(m_{i,j}-m_{i+1,j})^2 + (m_{i,j}-m_{i,j+1})^2}}\right\}. \tag{11c}$$

Here, h is the step size of a uniform grid and $i,j = 1,\dots,N-1$. The five point discretization of the regularization term above can be thought of as a consistent finite volume discretization of the differential operator $-\beta\nabla\cdot(\min\{\frac{1}{\gamma},\frac{1}{|\nabla m|}\}\nabla m)$ with natural boundary conditions.

In order to solve the nonlinear system (11c) numerically, a fixed point iteration called [illegible] has been proposed and studied extensively (see, e.g., [38] and references therein). Given an initial iterate m^0, one repeatedly and approximately solves

$$\beta L(m^{s-1})m^s + J^T(m^s)(F(m^s)-b) = 0, \quad s = 1,2,\dots. \tag{12}$$

Thus, if F is linear, then for each s we solve a linear, anisotropic diffusion equation. This method works very well for low accuracy requirements (i.e., it converges rapidly at first but then slows down as its asymptotic convergence rate is only linear), especially if J^TJ is nonsingular.

Global convergence, i.e., from any starting point m^0, for the modified TV norm was shown in several references in the case that J^TJ is constant and bounded away from singularity; see [13] and references in Chapter 8 of [38], where it is also claimed that some proofs easily extend to Huber's function with γ fixed.

Here, for the sake of completeness, we extend the proof by Chan and Mulet [9] to the case where $F(m) = Jm$, J constant with a full column rank, using Huber's function (5) with γ adaptively defined by (6).

THEOREM 2. [illegible] $F(m) = Jm$ [illegible] J [illegible] (1) (2) [illegible] $\{m^s\}$ [illegible] (12) [illegible]

1. $T(m^s) \le T(m^{s-1})$ [illegible] $s \ge 1$
2. $\lim_{s\to\infty} |m^s - m^{s-1}| = 0$
3. [illegible] $\{m^s\}$ [illegible]

The proof is given in Appendix A.

Another possibility proposed by Chan, Golub and Mulet [7] and advocated in [38] is a primal-dual Newton method: dual variables corresponding to the flux (in (11c) these are the discretized $\mathbf{v} = \min\{\frac{1}{\gamma}, \frac{1}{|\nabla m|}\}\,\nabla m$) are considered as independent variables along with m and a Newton method is defined, observing maximum size constraints for $\mathbf{v}$. However, the typical number of iterations required using lagged diffusivity to reach the rough error tolerances which makes sense in the present context is so small that the improved convergence rate of the primal-dual method may kick in too late to be of practical interest.

Lagged diffusivity and damped Gauss–Newton. Returning to lagged diffusivity, in case that $F(m)$ is nonlinear it makes sense to also linearize the second term in (12) in a comparable manner. In particular, it makes sense to use a single Gauss–Newton iteration, linearizing $F(m^s)$ around m^{s-1}, which leads to the following linear system to be solved at each iteration:

$$\left(J^TJ \;+\; \beta L\right)\; \delta m \;= J^T(b - F(m^{s-1})) \;-\; \beta L m^{s-1}, \tag{13}$$

with $J = J(m^{s-1})$, and $L = L(m^{s-1})$ defined in (11c). In general global convergence is no longer guaranteed by simply setting $m^s = m^{s-1} + \delta m$, and a line search (i.e., setting $m^s = m^{s-1} + \eta\,\delta m$, $0 < \eta \le 1$; see, for instance, [30]) or a trust region method may be necessary to guarantee sufficient reduction of the objective function $T(m)$ of (1).

2.2. On choosing the regularization parameter. In general, the optimization problem (1) must be solved many times by the process described above for different regularization parameter values β. This contributes in a major way to the total cost of the solution process. A solution is accepted when some stopping criterion is satisfied. This is not our focus in the present article; however, leading to the next section we mention the simplest such criterion, namely, the discrepancy principle. Thus, one requires

$$\phi_d(m(\beta)) = \|F(m) - b\|^2 \le \text{tol} \tag{14}$$

for some specified tolerance level tol which depends on the noise level and is assumed given. A procedure of continuation in β is then developed, where previous values of ϕ_d are used to estimate the next value of β, e.g., by a secant iteration.

In practice the tolerance is derived from statistical arguments, such as χ^2 distribution, and it must be noted that one rarely has more than a rough estimate of what tol should be. This is true even for the case of a smooth model recovery.

3. Numerical techniques for large problems.

Multilevel continuation. Lemma 1 assures us that $R(m)$ does not vary much as the resolution is increased on the same domain. The same holds true, of course, for the data fitting term in (1). We therefore conclude that [illegible] (6) [illegible] β [illegible].

As in [1, 20] we develop a multilevel continuation methodology, aimed chiefly at exporting the value of β constructed on a coarse grid to finer grids.

Starting on the coarse grid and subsampling the data as needed, the algorithm calculates an approximate solution and an approximate regularization parameter. Next, we interpolate the solution to a finer grid using a piecewise linear interpolation (cf. [1]) and solve the optimization problem using the interpolated solution as a starting guess, with the regularization parameter obtained on the coarse grid possibly slightly adjusted. The process continues until we reach the finest grid.

The reason that we may need to adjust β as the grid is refined is that the discretization error may play a more major role on coarser grids, particularly when the forward problem also involves the inversion of a discretized differential equation. (That "the coarsest grid must be fine enough" is a well-known requirement in the multigrid literature; see [1] in our particular context.) Thus, a search for β commences as in section 2.2, so that a criterion such as (14) with a "soft tolerance" is respected before work starts on the next finer grid.

A multigrid preconditioner. Although multilevel continuation decreases the number of iterations needed to compute a solution on the finer grids, each iteration is still expensive because (13) is a large system of linear equations which corresponds to a differential or integro-differential equation with nonsmooth coefficients. Therefore, an essential component in any fast algorithm for the solution of the problem is a fast algorithm for the solution of the linear system.

To simplify notation we rewrite the system as

$$H\ \delta m \equiv (J^T J + \beta L)\ \delta m\ =\ \text{rhs}. \tag{15}$$

There are two cases to consider here. First, if $J^T J$ is sparse with a sparsity pattern similar to that of L (e.g. denoising, superresolution), then a preconditioned conjugate gradient (PCG) method with a multigrid cycle as a preconditioner can be directly applied to the system. If on the other hand $J^T J$ is dense, then one can approximate $J^T J$ either by a sparse matrix, for example, $\text{diag}\{J^T J\}$, or by a low rank matrix (see [20]) and obtain an approximate $\hat{J}^T \hat{J} + \beta L$ which is used as a preconditioner. We now briefly discuss how to obtain such an approximation. Obtaining an efficient approximation to J is a nontrivial matter see, for example, [19, 38, 3, 4], which is beyond the scope of the present paper.

We therefore apply a multigrid preconditioner based on the approximate inversion of L plus a positive definite sparse matrix. Note that since L is a discretization of a differential operator with nonsmooth coefficients standard multigrid methods are ineffective. One can resort to algebraic multigrid (AMG) (see, e.g., [36]); however, this may not be as efficient as geometric multigrid which utilizes the underlining grid that exists for our problem. We therefore turn to geometric multigrid methods which are developed for problems with nonsmooth coefficients [12, 26].

Let m_F be a grid function on the finest grid of $(N+1)^2$ points, or nodes. We set the next coarser grid function, m_C, to be the grid function on $(N/2+1)^2$ points chosen in the obvious manner. The difference from standard multigrid is in the use of [illegible]. This approach combines algebraic multigrid with geometric multigrid methods to obtain the interpolation operator. As common to algebraic multigrid methods we divide the fine grid points to coarse points (C) and fine points (F). The approximate Gauss–Newton step is reordered as

$$\begin{pmatrix} H_{FF} & H_{FC} \\ H_{FC}^T & H_{CC} \end{pmatrix} \begin{pmatrix} \delta m_F \\ \delta m_C \end{pmatrix} = \text{rhs}.$$

We next define the [illegible]

$$P = \begin{pmatrix} H_{FF}^{-1} H_{FC} \\ -I \end{pmatrix}$$

and, eliminating the fine grid points, obtain the [illegible]

$$P^T \begin{pmatrix} H_{FF} & H_{FC} \\ H_{FC}^T & H_{CC} \end{pmatrix} P \; \delta m_C = \text{rhs}_C.$$

The obvious problem with the exact interpolation and coarse grid operators is that the matrix H_{FF} does not have a sparse inverse. Different approximations for its inverse define different methods. For a survey and comparison of methods see [26]. Here we have chosen a similar approach to the one suggested by Dendy [12] to obtain sparse interpolation and coarse grid operators. The restriction operator is chosen to be the transpose of the interpolation and symmetric Gauss–Seidel is used as a smoother. Combining the above components into a V(3,1) cycle we have found that in most cases a single cycle is sufficient to reduce the residual norm below 10^{-2}.

4. Numerical experiments.

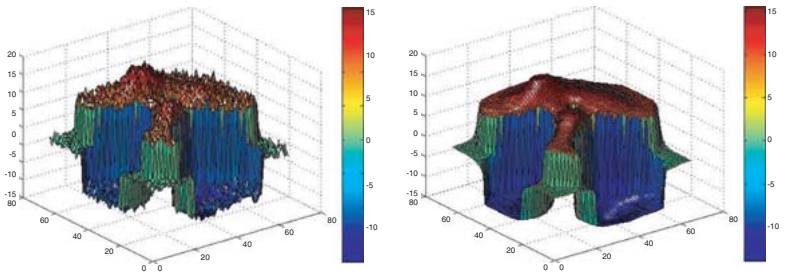
Experiment 1 (cont.). For simple denoising, $F(m) = m$ and hence $J = I$, the identity. We apply PCG iterations using our multigrid V-cycle preconditioner. The coarsest grid in all cases has $h_{coarsest} = 2^{-4}$.

For the example considered in section 1, $h = 2^{-8}$. Let us complete its details. The quality of the results strongly depends on β and some search was required before settling on $\beta = .07$ for a 20% noise. Using the Huber function with γ determined adaptively according to the formulas (6) and (7) we obtain after 10 lagged diffusivity iterations results which are very similar, indeed indistinguishable to the naked eye, to those displayed in the lower left and right panels, respectively, of Figure 1. The choice of 10 lagged diffusivity iterations has been made because that is when γ in (6) changes by less than 10% from one iteration to the next, but further iterations have not revealed significantly different results.

The respective final values of γ are 8.34 and 6.56×10^3, and this gives rise to the values of ε used in the experiment reported in section 1. There has been no notable difference in the performance of the Huber vs the modified TV variants in this case, and the results reported earlier therefore demonstrate both the general utility of the formula (6) and its superiority over (7). The (nonoptimized) CPU times reported in Table 1 correspond directly to the total number of PCG iterations required, and the latter count, for 10 nonlinear iterations in each case, is reported as "work units" in Table 1. The V-cycle preconditioner is therefore seen to perform rather effectively

TABLE 2
γ by (6) vs mesh width h.

h	2^{-4}	2^{-5}	2^{-6}	2^{-7}	2^{-8}
γ	4.05	2.03	1.04	0.518	0.255



(a) noisy data (b) denoised data

FIG. 2. *Denoising a surface with discontinuity.*

as well in our algorithm. The additional cost of carrying out the evaluation of the adaptive γ, by either formula (6) or (7), is negligible. In summary, then, our proposed algorithm performs very well indeed for this denoising problem.

Here is another denoising experiment. The "clean" data are synthesized from MATLAB's supplied function "peaks" modified as follows: $\forall\, |m_{i,j}| > 0.01$ $m_{i,j} = m_{i,j} + d\ sign(m_{i,j})$ We set $d = 10$. Adding 10% ($nlevel = 0.1$) Gaussian noise and setting $\beta = .06$ we run for different values of h and record in Table 2 the values of γ for the final m in each run. The value of γ is determined by (6). The linear dependence of γ on h clearly emerges. For the relative tolerance value of 10^{-2} on both the lagged diffusivity iteration and the PCG iteration, three lagged diffusivity iterations are required in all cases for both Huber and modified TV. A total of three PCG iterations (i.e., one for each lagged diffusivity iteration) are needed for modified TV while an additional PCG iteration is required for the Huber function. A typical result is depicted in Figure 2.

Note also that the same value of β has yielded quality results on all five grids, satisfying $\|F(m) - b\| < 2\|m\|nlevel$. On the other hand, when the coarsest grid was set coarser than $h = 2^{-4}$ it proved useless for the multilevel process.

A multilevel continuation process as described in section 3 has been set up as well. Bilinear interpolation from each grid to the next finer one proves more efficient than a similar piecewise constant interpolation. It cuts the iteration count on the finest grid to just one iteration and the overall elapsed time by a factor of more than two.

Repeating the above experiments with the lower noise level of 1% yields a faster convergence on the finest grid starting from the data (and requiring one or at most two iterations for convergence), so the multilevel continuation process is not useful. However, exporting the value of $\beta = .006$ determined on a coarse grid with the same $h_{coarsest}$ as before to finer grids does prove to work well here, too.

We have also made comparative runs choosing γ by the formula (7). The resulting value is significantly larger than γ of (6), especially on fine grids. Thus, the least squares norm is used more often in the regularization term R, typically resulting in

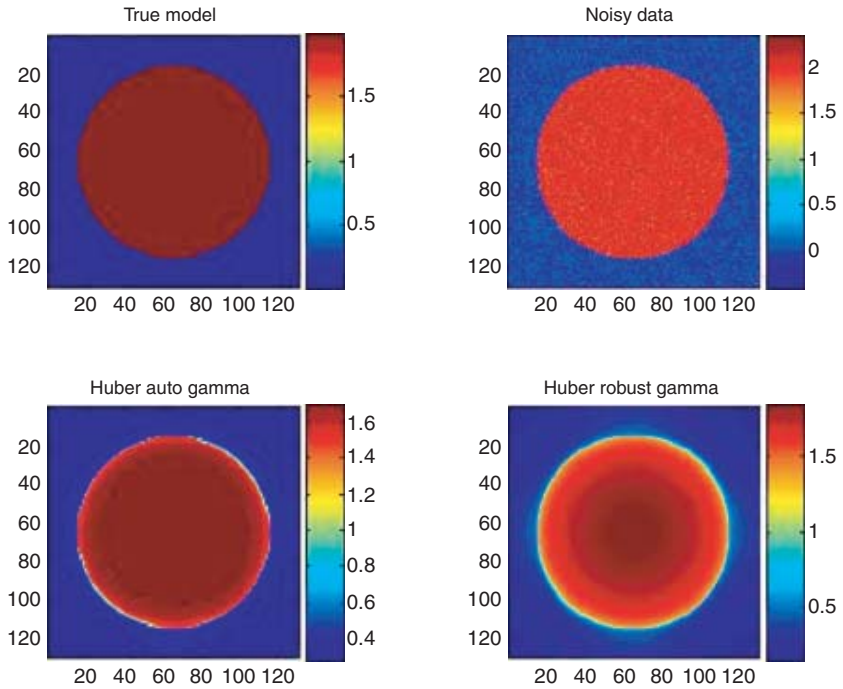


FIG. 3. *Denoising a piecewise constant surface: The true model takes the value* 2 *inside a circular pie subdomain (brown) and* 0 *in the background (blue). The run parameters are nlevel* $= .1$, $h = 2^{-7}$, $\beta = .06$. *The robust* $\gamma = 2.83$ *determined by* (7) *is much larger than the auto* $\gamma = .0276$ *determined by* (6). *Consequently, the discontinuity is more smeared in the lower right plot than in the lower left one.*

a possibly better approximation in smooth regions but in less sharp edges as well. An example of this sort is obtained here upon setting $d = 2$ in the above model. To make the point about the discontinuity smearing clearer to visualize, though, we plot in Figure 3 the results of a reconstruction of a piecewise constant surface.

Experiment 2. Next we consider the limited angle single photon emission tomography problem (SPECT). A brief description of the problem follows; see [23] for more detail. A chemical tracer is injected into a patient. This chemical emits photons in all directions. The chemical is targeted to bond to a specific functional area in the body (for example, it attaches itself to active muscles). As a result, a few minutes after the injection, some active parts of the body "glow" while others do not. In order to find these parts, the patient is put under a collimated camera. The camera can detect the glow only from a single direction. Mathematically, the number of photons that are detected at each collimator are

$$b_i = \int_{\text{ray}_i} m(x, y)\, dx\, dy. \tag{16}$$

The set of all the rays at the same direction forms a projection of the object onto this direction. The goal is to find m based on the projection data. In order to discretize the line integral (16) the model is divided into cells. We then trace each of the rays and obtain a matrix J. The entries of the matrix J_{ij} are the length of the ith ray in the jth cell. Since each ray usually passes through only a few cells, J is typically sparse.

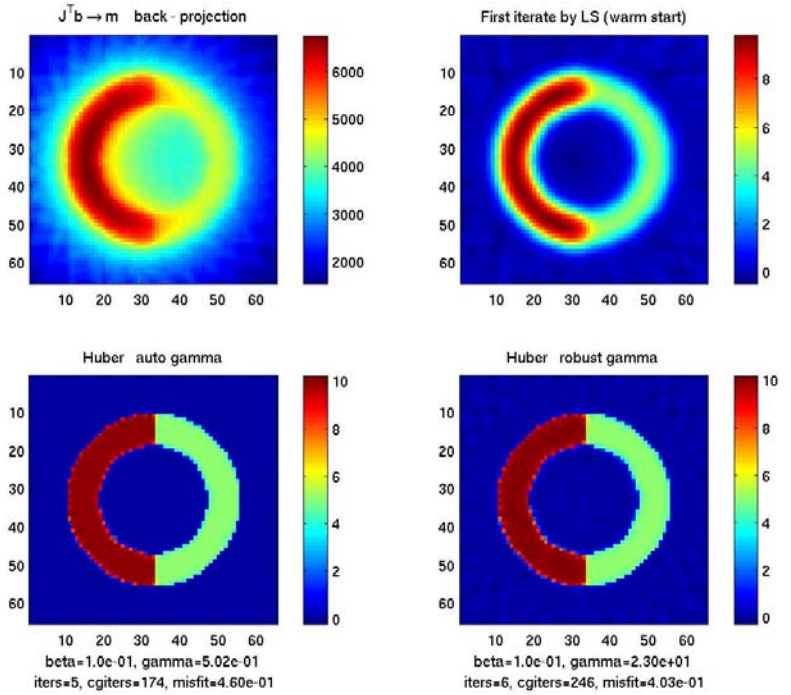


FIG. 4. *SPECT: noiseless data synthesized from two halves of a circular ring. Here* $\beta = .1$. *The back-projection image appears unfocused and the solution obtained by least squares regularization is useful as a starting iterate. The image obtained using Huber's function with* (6) *is excellent (lower left). The robust* γ (7) *gives somewhat poorer results.*

Here, the forward problem is linear,

$$F(m) = Jm.$$

The sensitivity matrix J is constant and has full row rank, but the problem is typically underdetermined and thus J^TJ is singular. For instance, one has to solve the problem (2) given by

$$J^TJm + \beta R_m = J^Tb,$$

where J is 2048×4225, for a 65×65 grid function m. So, we must have $\beta > 0$. Indeed, β may neither be too large nor too small. We have used a piecewise constant "true model" m_{true} which is well approximated in the lower left panel of Figure 4 to synthesize data b. At first we add no noise, so $b = Jm_{true}$. Results are pictured in Figure 4. Using the simple back-projection method, $m = J^Tb$, which is common in practice, leads to a blurred, out of scale image. We then set $\beta = 0.1$ and, keeping the same iteration tolerance values as in the previous experiment, apply our regularization techniques. (A significantly smaller β has led to an increased number of iterations due to the increased ill-conditioning of the linear system.) Using a least squares regularization ($\gamma = \infty$) yields blurred results which nonetheless prove useful as a starting iterate for a TV regularization. The Huber and modified TV regularizations produce excellent results with γ determined by (6). The TV regularization yields

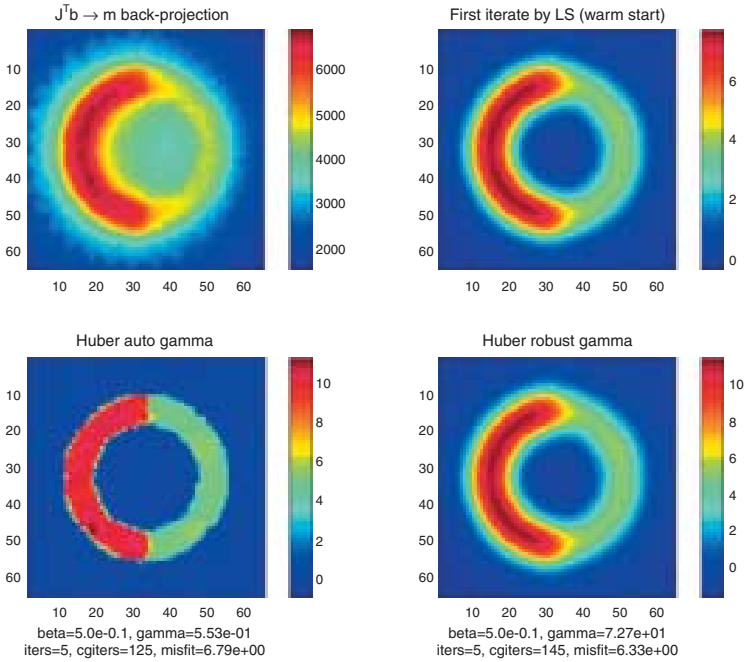


FIG. 5. *SPECT: data synthesized from two halves of a ring with* 10% *noise added on top. Here* $\beta = .5$. *The back-projection image appears unfocused and the solution obtained by least squares regularization is useful as a starting iterate. The image obtained using Huber's function with* (6) *is excellent (lower left). The robust* γ *gives inferior results.*

identical looking results and costs one additional work unit. The formula (7) for γ produces somewhat poorer results here; the difference can be more easily observed when reading an electronic copy of this article.

Next we add 10% normally distributed random noise to the data, set $\beta = .5$ and repeat the calculations. The results are gathered in Figure 5: the observations are similar to those in the noiseless case, except that now the performance of the "robust γ" is more clearly worse.

In Figure 6 we record the history of a continuation process with the same β. The total cost is only a fraction less than the one-grid process described above. The relative efficiency of multilevel continuation improves for larger values of β, though. Note how the dependence of γ of (6), not only on h but also on the smoothness of the surface, is expressed here.

We have also experimented with standard deblurring problems (e.g., [38, 24]) and found rather satisfactory results—and very close performance indicators—using either the modified TV or Huber's function with the switching parameter of (6). We do wish to repeat a general note of caution, though, that here too the TV regularization shines especially when recovering piecewise constant surfaces, although it does well also for a combination of rolling hills and steep canyons.

Experiment 3. The third experiment is conducted on problem (1) with $F(m) = Qu$, where u is the solution of the forward problem

$$A(m)u = q \tag{17}$$

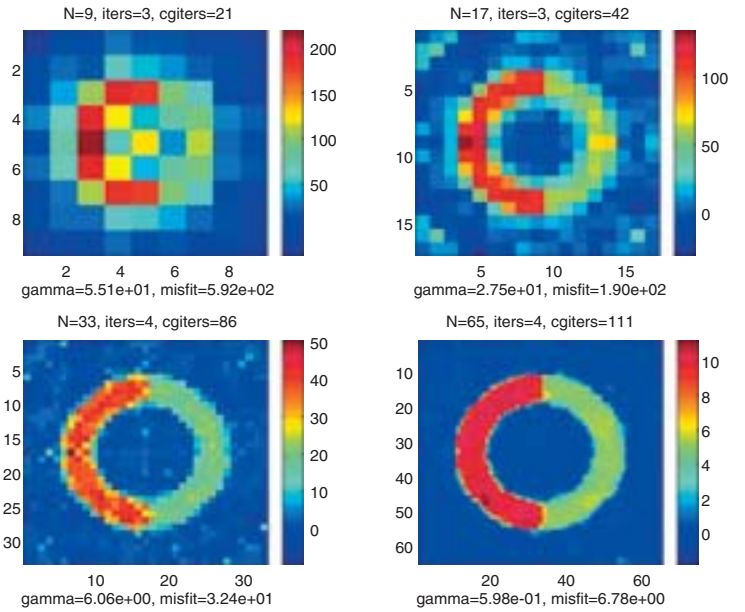


FIG. 6. *SPECT: data synthesized from two halves of a ring with* 10% *noise added on top. Here continuation with* $\beta = .5$ *is used. Note that* γ *get smaller faster than* h *because the surface becomes smoother.*

and Q indicates where in the domain the field u is being measured. Thus, $F(m) = QA(m)^{-1}q$. The matrix A discretizes an elliptic operator such as $\nabla \cdot (\sigma(\mathbf{x})\nabla)$, where $\sigma(\mathbf{x}) > 0$ is bounded away from 0, or Maxwell's equations in the frequency domain for a low frequency, and q are given sources. Both A and Q are typically large and sparse; see, e.g., [10, 5, 29, 21, 22, 2, 5, 8].

The discretized inverse problem in this case is much more ill-conditioned than in the previous problem instances. See [22] for realistic examples where the recovered field u approximates the unpolluted data b well while the quality of the reconstructed model is significantly poorer. Thus, one wonders in the context of piecewise continuous surfaces whether there is enough accurate data in applications to allow for an honest identification of discontinuities using such diffusive forward operators.

As a representative example we have considered the resistivity problem

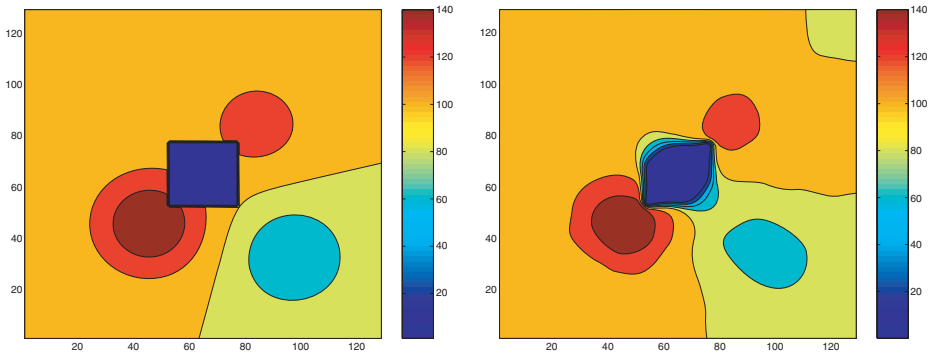
$$\nabla \cdot (m^{-1}\nabla u) = q \tag{18}$$

with natural boundary conditions on a unit domain in $2D$. The right-hand side is chosen with source and sink,

$$q = \exp(-10((x+0.6)^2 + (y+0.6)^2)) \\ - \exp(-10((x-0.6)^2 + (y-0.6)^2)),$$

and the "true model," depicted in Figure 7(a), contains discontinuities as well as smooth, nonconstant parts. We use this true model to generate a field on a 129×129 cell-centered grid and contaminate this with 1% white noise to yield the "observed data," b.

A standard finite volume discretization is applied to the forward problem, with harmonic averaging for m^{-1}.



(a) "True model" (b) Huber with $\beta = 10^{-5}$

FIG. 7. *The "true model" and the recovered model for Example* 1 *of Experiment* 3.

Figure 7(b) displays the recovered model using Huber's switching with $\beta = 10^{-5}$, which yields a final $\gamma = 4.6$ and an almost ideal misfit 1.01×10^{-2}. This is an excellent result.

However, the difference between a misfit of 1% and 2% is much too fine to define a cutting edge between a "good" and a "bad" model in more realistic situations, where we do not know the "true model" either. In fact, in most practical cases the noise level is unknown and one uses other statistical techniques such as GCV (e.g., [38]) to evaluate it. As it turns out one can vary the discontinuity locations significantly and still retain a misfit within 2%; see [2]. Unfortunately, therefore, in realistic experiments we may not know whether our recovered discontinuities are in the right place, not even approximately.

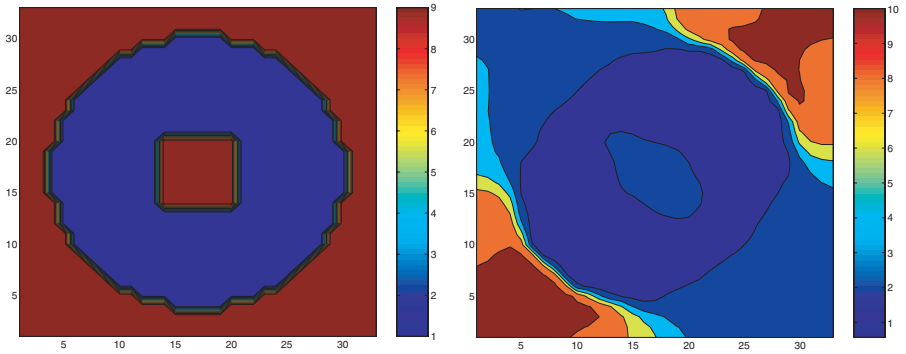
We then have considered a "true model" which only has two values and is given by the function

$$\mathrm{m}(x, y) = \begin{cases} 10, & -.2 < x, y < .2, \\ 10, & x^2 + y^2 > .8^2, \\ 1 & \text{otherwise.} \end{cases}$$

See Figure 8(a). The same setting as before is employed except that an exact solver is used for the linear equations encountered during the solution process and the grid size is 33×33.

The result using the Huber regularization with $\beta = 10^{-6}$ yielding $\gamma = 2.2$ is displayed in Figure 8(b). It is rather far from the "true" ring model. In particular, the cross diagonal symmetry is strongly violated, even though the misfit is an almost ideal 1.02×10^{-2}. Repeating the same experiment with a smaller $\gamma = .34$ yields a misfit of 8.7×10^{-3}, but the quality of the reconstructed model is not improved.

From these two examples it is clear that simply trusting the reconstruction because the misfit is "sufficiently small" cannot be advocated. Considering the maximum norm of the predicted minus the observed fields proves insufficient as well. It can then even be argued that displaying a smooth blob, such as obtained when using the least squares regularization with β small, is less committing than displaying a discontinuous solution, especially with only a few constant values, and as such is more commensurate with the actual information at hand.



(a) "True model" (b) Huber with $\beta = 10^{-6}$

FIG. 8. *The "true model" and the recovered model for Example* 2 *of Experiment* 3.

5. Conclusions and further comments. A new formula for selecting the Huber switching parameter γ, having selected β before, is proposed. This formula [illegible] what must be smooth and what can be a jump. The question of the automatic selection of γ (or ε for modified TV) is usually ignored among TV users, although it is important and has been considered by others who proposed a criterion based on robust statistics. However, we have demonstrated that our adaptive choice (6) is better, particularly for picking up edges.

Once γ has been determined, the constant ε in modified TV, if one insists on using this variant, is selected as $\varepsilon = 0.5\gamma$. We have shown this to be a good choice. The varying nature of either parameter provides further important information on the progress of the iteration process, and this together with the possibility of choosing larger values γ yield a method with capabilities beyond pure TV regularization.

A global convergence result for lagged diffusivity, similar to that known for modified TV, has been proved for the Huber variant. Moreover, with the above choice of ε numerical performance of the two variants has proved very close in many cases tried. This gives further reinforcement for our proposed formula.

Multilevel continuation has been proposed by keeping β fixed when refining the grid and by using bilinear interpolation of m for the next first iterate. The advantage in not having to calculate β from scratch on the fine grid can be huge.

Encouraging results are obtained for both the denoising and the SPECT problems. However, for diffusive forward problems the use of Huber or modified TV has tended to be unreliable. Note that in such problems it is possible to vary the discontinuity interface location significantly without violating the discrepancy principle for a given noise level.

Recent work has been carried out using level sets to recover the model under the assumption (which we do not utilize with the Huber and modified TV regularizations) that it is piecewise constant with known constant values (thus, it is a shape optimization problem [33, 27, 20, 14]). The stronger assumptions allow such methods to recover surfaces in situations where the Huber and TV regularizations fail, especially when the data are prescribed (more realistically) only in a ring next to the boundary, possibly in multiple sets. If the true model happens not to conform to these strong model assumptions, however, then there is no natural mode of recovery from determining a wrong model in this situation, especially when the data are sparse.

Both formulae (6) and (7) may fail, as will usual approaches for determining β, when processing an image with highly differing noise characteristics in different regions. A more local approach is then required.

We remark that in the image processing literature one often chooses to penalize even less through discontinuities than when using TV, e.g., using a Gaussian function or a Tukey biweight; see, e.g., [34, 31, 15]. However, this leads to nonconvex functionals and occasionally also to several local minima, which seems excessive in our more complex context where the forward problem operator is not simply the identity. We do not consider these more adventurous choices further here.

Appendix A. In this appendix we prove Theorem 2. To recall, we consider minimization of the functional $T(m)$ given by (1) where $F(m) = Jm$, J is a constant matrix with a full column rank. Set matrices $A_{ij}^T = \nabla_{i,j} \in \mathbb{R}^{N\times 2}, 1 \le i,j \le N$, so that the necessary conditions (2) can be written as

$$T_m(m) = \beta \sum_{i,j=1}^{N} \left(\frac{A_{ij}A_{ij}^T m}{|A_{ij}^T m|_\gamma} \right) + J^T(Jm - b) = 0,$$

where $|A_{ij}^T m|_\gamma = max\{\gamma, |A_{ij}^T m|\} > 0$. Then we have the following for T.

LEMMA 3. [illegible] T [illegible]

[illegible] Since $T(m) \ge \frac{1}{2}\|Jm - b\|^2$, T is coercive and bounded below by 0. Since T is twice *Fréchet* differentiable, we have for its Hessian

$$\begin{aligned} T_{mm} &= \beta \textstyle\sum_{i,j=1}^{N} A_{ij} \left(\frac{1}{|A_{ij}^T m|_\gamma} \left(I_2 - \frac{A_{ij}^T m \otimes A_{ij}^T m}{|A_{ij}^T m|_\gamma^2} \right) + \rho''(|A_{ij}^T m|) \frac{A_{ij}^T m \otimes A_{ij}^T m}{|A_{ij}^T m|_\gamma^2} \right) A_{ij}^T \\ &\quad + J^T J, \end{aligned}$$

where ρ is defined by (5) and $\bigotimes$ denotes the Kronecker product. By Cauchy–Schwarz, this yields

$$\begin{aligned} (A_{ij}^T m, A_{ij}^T v)^2 &\le (A_{ij}^T m, A_{ij}^T m)(A_{ij}^T v, A_{ij}^T v) \le |A_{ij}^T m|_\gamma^2 |A_{ij}^T v|^2 \\ &\Longrightarrow |A_{ij}^T v|^2 - \frac{(A_{ij}^T m, A_{ij}^T v)^2}{|A_{ij}^T m|_\gamma^2} \ge 0. \end{aligned}$$

Moreover, we have $\rho''(\tau) \ge 0$ if $\tau \ge 0$. So, the Hessian of T satisfies, $\forall\, v \ne 0$,

$$\begin{aligned} T_{mm}(v,v) &= \beta \textstyle\sum_{i,j=1}^{N} \left(\frac{1}{|A_{ij}^T m|_\gamma} \left(|A_{ij}^T v|^2 - \frac{(A_{ij}^T m, A_{ij}^T v)^2}{|A_{ij}^T m|_\gamma^2} \right) + \rho''(|A_{ij}^T m|) \frac{(A_{ij}^T m, A_{ij}^T v)^2}{|A_{ij}^T m|_\gamma^2} \right) \\ &\quad + v^T J^T J v > 0. \end{aligned}$$

Hence T is a strictly convex functional. ∎

Thus problem (1) has a unique solution. We then define

$$G(v,m) = T(m) + (v - m, T_m(m)) + \frac{1}{2}((v-m), C(m)(v-m))$$

and follow [9] almost verbatim in recognizing that the lagged diffusivity algorithm is an instance of the generalized Weiszfeld method, so that the following lemma holds.

LEMMA 4.

1. $C(m) = \beta \sum_{i,j=1}^{N} (A_{ij}A_{ij}^T/|A_{ij}^T m|_\gamma) + J^T J$ [illegible]
2. $\lambda_{min}(C(m)) \ge \mu = \lambda_{min}(J^T J) > 0\ \forall\, m$
3. $T(v) \le G(v,m)\ \forall\, v$

Using Lemmas 3 and 4 we can now establish the global convergence for the Huber switching function with γ adaptively defined by (6), as stated in Theorem 2. We address each item in turn:

1. From the above properties, we have $T(m^s) \leq G(m^s, m^{s-1})$. Moreover, the fixed point iteration equation is equivalent to the necessary condition for the generalized Weiszfeld method, which yields

$$m^s = \operatorname*{argmin}_v G(v, m^{s-1}) \Rightarrow G(m^s, m^{s-1}) \leq G(m^{s-1}, m^{s-1}) = T(m^{s-1}).$$

2. Recall that

$$\begin{aligned} T(m^s) &\leq G(m^s, m^{s-1}) \\ &= T(m^{s-1}) + (m^s - m^{s-1}, T_m(m^{s-1})) \\ &\quad + \tfrac{1}{2}((m^s - m^{s-1}), C(m^{s-1})(m^s - m^{s-1})) \\ &= T(m^{s-1}) - \tfrac{1}{2}((m^s - m^{s-1}), C(m^{s-1})(m^s - m^{s-1})). \end{aligned}$$

Then taking into account that $\lambda_{min}(C(m^{s-1})) \geq \mu > 0$, we have

$$\frac{1}{2}\mu|m^s - m^{s-1}|^2 \leq \frac{1}{2}((m^s - m^{s-1}), C(m^{s-1})(m^s - m^{s-1})) \leq T(m^{s-1}) - T(m^s).$$

It transpires that

$$0 \leq |m^s - m^{s-1}| \leq \sqrt{2\mu^{-1}(T(m^{s-1}) - T(m^s))}.$$

Note that T is bounded below and $\{T(m^s)\}$ is monotonically decreasing. Thus, $\{T(m^s)\}$ converges, implying that $\lim_{s\to\infty} |T(m^s) - T(m^{s-1})| = 0$; hence also $\lim_{s\to\infty} |m^s - m^{s-1}| = 0$.

3. Since $\{T(m^s)\}$ is monotonically decreasing, it is bounded above of course. The fact that T is coercive then guarantees that $\{m^s\}$ is bounded. By compactness, it suffices to show that the limit of every convergent subsequence of $\{m^s\}$ is the global minimum of (1). Let $\{m^{s_i}\}$ be a subsequence of $\{m^s\}$ that converges to m^*. Since $G_v(v, m) = T_m(m) + C(m)(v - m)$, T is continuously differentiable and C is continuous, we have that G_v is continuous. From the above $\lim_{s_i\to\infty} |m^{s_i} - m^{s_i-1}| = 0$, and we also have $\lim_{s_i\to\infty} m^{s_i} = \lim_{s_i\to\infty} m^{s_i-1} = m^*$. Then, taking limits for $G_{m^{s_i}}(m^{s_i}, m^{s_i-1})$, we deduce the following equation:

$$\begin{aligned} 0 &= \lim_{s_i\to\infty} G_{m^{s_i}}(m^{s_i}, m^{s_i-1}) = G_{m^{s_i}}(\lim_{s_i\to\infty} m^{s_i}, \lim_{s_i\to\infty} m^{s_i-1}) \\ &= G_{m^{s_i}}(m^*, m^*) = T_m(m^*) + C(m^*)(m^* - m^*) = T_m(m^*). \end{aligned}$$

Hence the sequence $\{m^s\}$ converges to the unique global minimum m^* of our problem (1).

REFERENCES

[1] U. Ascher and E. Haber, *Grid refinement and scaling for distributed parameter estimation problems*, Inverse Problems, 17 (2001), pp. 571–590.

[2] U. Ascher and E. Haber, *Computational methods for large distributed parameter estimation problems with possible discontinuities*, in Proceedings of the Inverse Problems, Design and Optimization Symposium, M. Colaco, H. Orlande and G. Dulikravich, eds., 2004, pp. 201–208.

[3] G. Biros and O. Ghattas, *Parallel Lagrange–Newton–Krylov–Schur methods for PDE-constrained optimization. Part* I: *The Krylov–Schur solver*, SIAM J. Sci. Comput., 27 (2005), pp. 687–713.
[4] G. Biros and O. Ghattas, *Parallel Lagrange–Newton–Krylov–Schur methods for PDE-constrained optimization. Part* II: *The Lagrange–Newton solver and its application to optimal control of steady flows*, SIAM J. Sci. Comput., 27 (2005), pp. 714–739.
[5] L. Borcea, J. G. Berryman, and G. C. Papanicolaou, *High-contrast impedance tomography*, Inverse Problems, 12 (1996), pp. 835–858.
[6] A. Chambolle, *An algorithm for total variation minimization and applications*, J. Math Imaging Vision, 20 (2004), pp. 89–97.
[7] T. F. Chan, G. H. Golub, and P. Mulet, *A nonlinear primal-dual method for total variation-based image restoration*, SIAM J. Sci. Comput., 20 (1999), pp. 1964–1997.
[8] T. Chan and X. Tai, *Identification of discontinuous coefficients in elliptic problems using total variation regularization*, SIAM J. Sci. Comput., 25 (2003), pp. 881–904.
[9] T. F. Chan and P. Mulet, *On the convergence of the lagged diffusivity fixed point method in total variation image restoration*, SIAM J. Numer. Anal., 36 (1999), pp. 354–367.
[10] M. Cheney, D. Isaacson and J. C. Newell, *Electrical impedance tomography*, SIAM Rev., 41 (1999), pp. 85–101.
[11] J. Claerbout and F. Muir, *Robust modeling with erratic data*, Geophys., 38 (1973), pp. 826–844.
[12] J. E. Dendy, Jr., *Black box multigrid*, J. Comput. Phys., 48 (1982), pp. 366–386.
[13] D. C. Dobson and C. R. Vogel, *Convergence of an iterative method for total variation denoising*, SIAM J. Numer. Anal., 34 (1997), pp. 1779–1791.
[14] O. Dorn, E. L. Miller, and C. M. Rappaport, *A shape reconstruction method for electromagnetic tomography using adjoint fields and level sets*, Inverse Problems, 16 (2000), pp. 1119–1156.
[15] F. Durand and J. Dorsey, *Fast bilateral filtering for the display of high-dynamic-range images*, ACM SIGGraph 2002, pp. 257–266.
[16] I. Elbakri and J. Fessler, *Statistical image reconstruction for polyenergetic x-ray computed tomography*, IEEE Trans. Med. Imag., 21(2) (2002), pp. 89–99.
[17] H. W. Engl, M. Hanke, and A. Neubauer, *Regularization of Inverse Problems*, Kluwer, Dordrecht, 1996.
[18] C. Farquharson and D. Oldenburg, *Non-linear inversion using general measures of data misfit and model structure*, Geophys., 134 (1998), pp. 213–227.
[19] E. Haber, *Numerical Strategies for the Solution of Inverse Problems*, Ph.D. thesis, University of British Columbia, Vancouver, 1997.
[20] E. Haber, *A multilevel, level-set method for optimizing eigenvalues in shape design problems*, J. Comput. Phys., 198 (2004), pp. 518–534.
[21] E. Haber and U. Ascher, *Preconditioned all-at-one methods for large, sparse parameter estimation problems*, Inverse Problems, 17 (2001), pp. 1847–1864.
[22] E. Haber, U. Ascher, and D. Oldenburg, *Inversion of* 3*D electromagnetic data in frequency and time domain using an inexact all-at-once approach*, Geophys., 69 (2004), pp. 1216–1228.
[23] E. Haber, D. Oldenburg, and A. Celler, *Recovery of dynamic parameters in SPECT*, IEEE Trans. Nuc. Sci., 67 (1997), pp. 129–135.
[24] P. C. Hansen, *Rank Deficient and Ill-Posed Problems*, SIAM, Philadelphia, 1998.
[25] P. J. Huber, *Robust estimation of a location parameter*, Ann. Math. Stat., 35 (1964), pp. 73–101.
[26] S. Knapek, *Matrix-dependent multigrid homogenization for diffusion problems*, SIAM J. Sci. Comput., 20(2) (1999), pp. 515–533.
[27] A. Leitao and O. Scherzer, *On the relation between constraint regularization, level sets, and shape optimization*, Inverse Problems, 19 (2003), pp. L1–L11.
[28] S. McCormick and J. Wade, *Multigrid solution of a linearized, regularized least-squares problem in electrical impedance tomography*, Inverse Problems, 9 (1993), pp. 697–713.
[29] G. A. Newman and D. L. Alumbaugh, *Three-dimensional magnetotelluric inversion using nonlinear conjugate gradients*, Geophys. J. Internat., 140 (2000), pp. 410–418.
[30] J. Nocedal and S. Wright, *Numerical Optimization*, Springer, New York, 1999.
[31] P. Perona and J. Malik, *Scale-space and edge detection using anisotropic diffusion*, IEEE Trans. on Pattern Anal. and Machine Intelligence, 12(7) (1990), pp. 629–639.
[32] L. Rudin, S. Osher, and E. Fatemi, *Nonlinear total variation based noise removal algorithms*, Phys. D, 60 (1992), pp. 259–268.
[33] F. Santosa, *A level-set approach for inverse problems involving obstacles*, ESAIM Control Optim. Calc. Var., 1 (1996), pp. 17–33.

[34] G. Sapiro, *Geometric Partial Differential Equations and Image Analysis*, Cambridge University Press, London, 2001.
[35] O. Scherzer, *Scale-space methods and regularization for denoising and inverse problems*, Adv. in Image and Electron Phys., 128 (2003), pp. 445–530.
[36] U. Trottenberg, C. Oosterlee, and A. Schuller, *Multigrid*, Academic Press, London, 2001.
[37] C. Vogel, *Negative results for multilevel preconditioners in image deblurring*, in Scale-Space Theories in Computer Vision, M. Nielsen, P. Johansen, O. F. Olsen, and J. Weickert, eds., Springer-Verlag, New York, 1999.
[38] C. Vogel, *Computational Methods for Inverse Problem*, SIAM, Philadelphia, 2002.

SIAM J. SCI. COMPUT.
Vol. 28, No. 1, pp. 359–381
© 2006 Society for Industrial and Applied Mathematics

LIMITER-FREE THIRD ORDER LOGARITHMIC RECONSTRUCTION*

ROBERT ARTEBRANT† AND H. JOACHIM SCHROLL†

Abstract. A third order conservative reconstruction, in the context of finite volume schemes for hyperbolic conservation laws, is constructed based on logarithmic functions. This logarithmic method reconstructs without the use of a limiter, any preprocessing of input data, special treatments for local extrema, or shock solutions. Also the method is local in the sense that data from only the nearest neighbors are required.

We test the new reconstruction method in several numerical experiments, including nonlinear systems in one and two space dimensions.

Key words. conservation law, finite volume method, high order reconstruction

AMS subject classifications. 35L65, 35L45, 65M06, 76N15

DOI. 10.1137/040620187

1. Introduction.

In the present paper we develop a new reconstruction method for the computation of accurate numerical fluxes within the context of finite volume (FV) methods for conservation laws. These FV schemes are derived from the integro-differential formulation of the conservation law:

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_\Omega u(x,t)\,\mathrm{d}x + \int_{\partial\Omega} f(u(x,t))\cdot n\,\mathrm{d}S = 0. \tag{1.1}$$

Within a spatial domain Ω the total amount of a certain quantity u is conserved; i.e., it does not change with time except by flow across the boundary $\partial\Omega$. The conserved quantity is the total amount or the [illegible] of u, while the dynamics of this average is determined by [illegible] of the flux $f(u(t,x))$ along the boundary. From given cell averages the [illegible] task is to accurately determine these point values along the boundary.

The very successful essentially nonoscillatory (ENO) reconstruction technique, developed in [3], uses polynomials and an adaptive selection of the stencil according to local smoothness. Solutions to hyperbolic conservation laws are generally only piecewise smooth, so the adaptive stencil procedure is a way to avoid interpolating discontinuous data. In [5], the weighted ENO (WENO) was developed, improving ENO in several ways and using a convex combination of all candidate stencils instead of just one. For recent developments of WENO, see [10] and references therein.

In this paper we are interested in a [illegible] reconstruction on a 3-point stencil. Further we would like to avoid any limiter affecting accuracy and costly modifications enforcing nonoscillatory properties. That is, in this setting the reconstruction problem essentially boils down to that of approximating functions. Then it is appropriate to consider a wider class of reconstructing functions, perhaps leading to a nonlinear approximation problem; as Rice puts it in [8] on the problem of approximating $f(x)$:

*Received by the editors December 5, 2004; accepted for publication (in revised form) June 22, 2005; published electronically March 24, 2006. This work was supported by the European network HYKE, funded by the EC as contract HPRN-CT-2002-00282.

http://www.siam.org/journals/sisc/28-1/62018.html

†Centre for Mathematical Sciences, Lund University, Box 118, SE–221 00 Lund, Sweden (robert.artebrant@na.lu.se, achim.schroll@na.lu.se).

"The key to efficient approximation is to find an approximating function which can take on the same nature or behaviour as $f(x)$." This consideration makes it reasonable to try approximating functions other than polynomials; polynomials are unable to take on very large slopes followed by a flat region, which frequently occur in the piecewise smooth solutions to hyperbolic conservation laws.

In [6] Marquina proposes local reconstruction techniques (LHR and LHHR) based on the hyperbola as a reconstructing function. The hyperbola is more flexible than a polynomial; it is able to assume this shape with a large slope followed by a flatter part, making it possible to avoid discontinuities inside the cells and locating them at the cell boundaries. The hyperbola is, however, monotonic, and so the order reduces from three to one and a half at local extrema. Inspired by these reconstructions, a third order reconstruction for monotone data (LLR) and a fifth order (if existent) reconstruction (DLR) were developed in [1]. The [illegible] (LDLR) developed in the present paper improves over LLR and DLR in the following aspects: In contrast to LLR, the LDLR can reconstruct local extrema at full third order accuracy, and its variation within a cell is controlled. The capability to reconstruct local extrema may have a major impact on simulation results, as demonstrated in Example 12, section 3.4. In contrast to DLR, the LDLR exists for all input data, is defined on a 3-point stencil, and does not use any smoothness indicators.

In the next section the LDLR is constructed. Then in section 3 several numerical experiments are presented, testing the LDLR and comparing it to LHHR, ENO3, and WENO3. Higher order extension is presented in section 4, and concluding remarks follow (section 5). Finally in the appendix we present a possible extension of Marquina's LHHR, following the principle to obtain the LDLR.

2. Local double logarithmic reconstruction. In this section the piecewise reconstruction of an unknown smooth function using two logarithmic functions is described. The resulting reconstruction method uses data from only the central cell and its nearest neighbors and is therefore called local double logarithmic reconstruction (LDLR). The grid data are the cell averages and approximations to the lateral derivatives of the function to be reconstructed.

We are interested in a third order reconstruction of low variation without the use of a limiter. A simple Taylor expansion proves the principle used to obtain the third order of accuracy

LEMMA 2.1. [illegible] $f, g \in C^3[a,b]$ [illegible] $h = b - a$ [illegible]

$$\int_a^b f(x)dx = \int_a^b g(x)dx, \tag{2.1}$$

$$(f-g)'(a) = \mathcal{O}(h^2) = (f-g)'(b); \tag{2.2}$$

[illegible] $(f-g)(x) = \mathcal{O}(h^3)$ [illegible] $x \in [a,b]$

2.1. The ansatz function. We consider the computational grid $x_i = ih$, i integer and h a positive number, with cells $C_i = [x_{i-1/2}, x_{i+1/2}]$.

The reconstruction is piecewise defined in each cell; therefore we shall consider without restriction a reference cell C_0. The reconstructing function in this cell has the form

$$r_0(x) \sim A + B\log(x+C) + D\log(x+E).$$

We define for $x \in C_0$

$$\phi_0(x) = -\frac{ch}{a}\log\left[x - x_0 - \frac{h}{2}\left(\frac{2}{a} - 1\right)\right] - \frac{dh}{b}\log\left[x - x_0 - \frac{h}{2}\left(\frac{2}{b} - 1\right)\right], \tag{2.3}$$

where a, b, c, and d are parameters to be determined, and let

$$r_0(x) = v_0 + \phi_0(x) - \frac{1}{h}\int_{C_0}\phi_0(\xi)d\xi \tag{2.4}$$

be the reconstructing function.

For (2.4) to be a well-defined smooth function, we must pose some restrictions on the coefficients a and b. The lemma below gives the allowed range of these parameters.

LEMMA 2.2. [illegible] $\phi(x) = \frac{h}{\lambda}\log\left[x - x_0 - \frac{h}{2}\left(\frac{2}{\lambda} - 1\right)\right]$ [illegible] $\psi(x) = \phi(x) - \frac{1}{h}\int_{C_0}\phi(\xi)d\xi$ [illegible] $x \in C_0$ [illegible] ψ [illegible] $\lambda \in (-\infty, 1)$ [illegible]. Let $t(x) = x - x_0 + h/2 - h/\lambda$ be the argument of the log-function. The function ϕ has a singularity for $x \in C_0$ when $t(x) = 0$. Since t is linear in x and

$$\min_{x\in C_0} t(x) = -\frac{h}{\lambda} \quad \text{and} \quad \max_{x\in C_0} t(x) = h - \frac{h}{\lambda}, \tag{2.5}$$

t has a zero in C_0 for $\lambda \in [1, \infty)$. For $\lambda < 0$ we see that $t > 0$ and therefore ϕ, and hence ψ, is real and smooth for λ negative. Now consider $\lambda \in (0, 1)$. From (2.5) it follows that $t(x) < 0$ for $\lambda \in (0, 1)$; hence $\phi(x) = \frac{h}{\lambda}(\log|t(x)| + i\pi)$ and

$$\begin{aligned}\frac{\lambda}{h}\psi(x) &= \log|t(x)| + i\pi - \frac{1}{h}\int_{C_0}\log|t(x)| + i\pi dx \\ &= \log|t(x)| - \frac{1}{h}\int_{C_0}\log|t(x)|dx,\end{aligned}$$

and thus ψ is real and smooth for $\lambda \in (0, 1)$. Finally, ψ has a removable singularity at $\lambda = 0$ by setting $\psi(x)|_{\lambda=0} = x_0 - x$. □

2.2. Order conditions and symmetry. To reach third order we impose the conditions set in Lemma 2.1; given v_0, the cell average of the unknown function in C_0, and d_1 and d_2 second order approximations to the left and right derivatives, we require

$$\frac{1}{h}\int_{C_0} r_0(x)dx = v_0, \tag{2.6}$$

$$r_0'\left(x_0 - \frac{h}{2}\right) = d_1, \tag{2.7}$$

$$r_0'\left(x_0 + \frac{h}{2}\right) = d_2. \tag{2.8}$$

The reconstructing function (2.4) preserves the cell average v_0 by construction. Conditions (2.7)–(2.8) lead to the following pair of equations:

$$c + d = d_1, \tag{2.9}$$

$$-\frac{d(a-1) + c(b-1)}{(a-1)(b-1)} = d_2. \tag{2.10}$$

Thus we have four parameters to satisfy two equations. We solve for two parameters, leaving the others undetermined for later use. Simplicity and symmetry suggest solving for c and d, which appear linearly. We obtain

$$c = \frac{(a-1)(d_2(1-b)-d_1)}{b-a}, \tag{2.11}$$

$$d = d_1 - c. \tag{2.12}$$

Now the order conditions are satisfied, and we turn to considering other properties of the reconstruction that are feasible using the remaining parameters.

The reconstruction should be symmetric (to the extent allowed by the ansatz function) in the sense that if $r_0'(x_0 - h/2) = -r_0'(x_0 + h/2)$, then $r'(x_0) = 0$. This can be realized by setting $d_1 = -d_2$ and solving for b. We get

$$0 = r_0'(x_0)|_{d_1=-d_2} = 2d_2 \frac{ab - a - b}{(a-2)(a-1)},$$

that is,

$$b = \frac{a}{a-1}. \tag{2.13}$$

As shown by Lemma 2.2, the reconstructing function is well defined, provided $a < 1$ and $b < 1$. The parameter b in (2.13), seen as a function of a, has the range $(-\infty, 1)$ for $a \in (-\infty, 1)$. We also observe that $a = b$ (the reconstructing function degenerates to LLR [1]) is only possible for $a = b = 0$.

In the next section we use the remaining parameter a to control the variation of the reconstructing function.

2.3. Local variation boundedness. To construct a reconstruction procedure of sufficiently low variation we apply Marquina's concept of local variation boundedness [6].

DEFINITION 2.3 (local variation bounded). [illegible] local variation [illegible] f [illegible] C_i [illegible] $LV(f_i) = TV(f)|_{C_i}$ [illegible] local variation bounded [illegible] C_i [illegible] $LV(f_i) = \mathcal{O}(h)$ [illegible] h [illegible]

The interesting case is if a function f stays LVB in the case of a discontinuity within the cell, i.e., $f'(x_i - h/2) = \mathcal{O}(1/h)$ and $f'(x_i + h/2) = \mathcal{O}(1)$ or vice versa.

Assuming a [illegible], the local variation of the reconstructing function (2.4) can be expressed in terms of its parameters a, b, c, and d:

$$LV(r_0) = h \left| \frac{cb \log(1-a) + da \log(1-b)}{ab} \right|. \tag{2.14}$$

In the previous section we determined c, d, and b in (2.11)–(2.13) so that order and symmetry conditions were satisfied. We plug (2.11)–(2.13) into (2.14) to obtain an expression for the local variation containing the free parameter a:

$$LV(r_0) = h \frac{a-1}{a-2} \left| \frac{\log(1-a)(d_1 + d_2)}{a} \right|. \tag{2.15}$$

To set a variation condition on a we consider the case of a discontinuity within the cell, i.e., $d_1 = \mathcal{O}(1/h)$ and $d_2 = \mathcal{O}(1)$. Inspection of (2.15) leads us to the following variation lemma for r_0 (2.4) with coefficients c, d, and b from (2.11)–(2.13)

LEMMA 2.4. [illegible] $d_1 = \mathcal{O}(1/h)$ $d_2 = \mathcal{O}(1)$ [illegible] $d_1 d_2 > 0$ [illegible] $a = 1 - mh^q$ $0 < m \le \hat{m} < \infty$ [illegible] $LV(r_0) = \mathcal{O}(h^{|q|}|\log(h)|)$

[illegible]. Let us assume $q > 0$; the case $q < 0$ is similar. We compute (2.15) with $d_1 = M/h$ and $d_2 = N$ with $MN > 0$: $LV(r_0) = h\frac{mh^q}{1+mh^q}\left|\frac{\log(mh^q)(M/h+N)}{1-mh^q}\right| = \frac{mh^q}{|1-m^2h^{2q}|}|\log(m) + q\log h||M + Nh| = \mathcal{O}(h^q|\log h|)$. □

A special feature of LDLR is that [illegible] are reconstructed within a cell. Next we shall see that even in this case the local variation is under control. It turns out that the local variation of the reconstructing function in the case $d_1 d_2 < 0$ is not bigger than in the case of monotone data $|d_1|$ and $|d_2|$ (or $-|d_1|$ and $-|d_2|$).

The variation controlling parameter is assumed to be $a \in (0, 1)$, and its symmetric counterpart is $b = a/(a-1)$. Now let

$$k_0 = \frac{a-1}{b-a} = \frac{(a-1)^2}{a(2-a)} > 0,$$
$$k_1 = 1 - b = \frac{1}{1-a} > 0.$$

For $d_1 = M$, $d_2 = N$ with $MN > 0$ we write the linear coefficients from (2.11)–(2.12) as

$$c = k_0 k_1 N - k_0 M,$$
$$d = (1 + k_0)M - k_0 k_1 N.$$

Computing the coefficients for $N \to -N$, we obtain

$$\widetilde{c} = -k_0 k_1 N - k_0 M,$$
$$\widetilde{d} = (1 + k_0)M + k_0 k_1 N.$$

Define the functions

$$f(x) = \frac{2h}{2a(x_0 - x) + h(2-a)},$$

$$g(x) = \frac{2h(a-1)}{2a(x_0 - x) - h(2-a)}.$$

We can write the derivative of the reconstructing function (2.4) for the two cases as

$$r_0'(x) = cf(x) + dg(x) = k_0 k_1 N(f - g) - k_0 M(f - g) + Mg,$$
$$\widetilde{r_0}'(x) = \widetilde{c}f(x) + \widetilde{d}g(x) = -k_0 k_1 N(f - g) - k_0 M(f - g) + Mg.$$

Consider the functions $\sigma(x) = f(x) - g(x)$ and $\tau(x) = g(x) - k_0[f(x) - g(x)]$ such that $r_0'(x) = k_0 k_1 N\sigma(x) + M\tau(x)$:

$$\sigma(x) = \frac{2ha(2-a)(2(x_0 - x) - h)}{[2a(x_0 - x) + h(2-a)][2a(x_0 - x) - h(2-a)]},$$

$$\tau(x) = \frac{2h(2(x_0 - x) + h)(a-1)}{[2a(x_0 - x) + h(2-a)][2a(x_0 - x) - h(2-a)]}.$$

The denominator of σ and τ is

$$n(x) = [2a(x_0 - x)]^2 - [h(2-a)]^2.$$

For the spatial cell $C_0 = [x_0 - h/2, x_0 + h/2]$ we have $\max_{x \in C_0} n(x) = n(x_0 \pm h/2) = 4h^2(a-1)$. Since $a \in (0,1)$, it holds that $\max_{x \in C_0} n(x) < 0$, and thus $n(x) < 0$ for all $x \in C_0$. Now consider the numerator of σ,

$$t_\sigma(x) = 2ha(2-a)(2(x_0 - x) - h).$$

Here $t_\sigma(x)$ is linear in x and decreasing since $a \in (0,1)$. We have $\max_{x \in C_0} t_\sigma(x) = t_\sigma(x_0 - h/2) = 0$, and thus $t_\sigma(x) \le 0$, for all $x \in C_0$. The denominator is always negative, and the numerator always nonpositive, so we get

$$\sigma(x) = f(x) - g(x) \ge 0 \qquad \forall x \in C_0.$$

Now consider the numerator of τ,

$$t_\tau(x) = 2h(2(x_0 - x) + h)(a-1).$$

Here $t_\tau(x)$ is linear in x and increasing since $a \in (0,1)$. We have $\max_{x \in C_0} t_\tau(x) = t_\tau(x_0 + h/2) = 0$, and thus $t_\tau(x) \le 0$ for all $x \in C_0$. Therefore we find

$$\tau(x) = g(x) - k_0[f(x) - g(x)] \ge 0 \qquad \forall x \in C_0.$$

Let us assume $M > 0$ and $N > 0$; then

$$r_0'(x) = k_0 k_1 N \sigma(x) + M\tau(x) > 0$$

and

$$\begin{aligned} r_0'(x) + \widetilde{r_0}'(x) &= 2M\tau(x) \ge 0, \\ r_0'(x) - \widetilde{r_0}'(x) &= 2k_0 k_1 N \sigma(x) \ge 0, \end{aligned}$$

and so $|\widetilde{r_0}'(x)| \le |r_0'(x)|$ for all $x \in C_0$. The other cases follow by symmetry. In summary we have the following claim.

LEMMA 2.5. [illegible] $\widetilde{r_0}$ [illegible] $d_1 d_1 < 0$ [illegible] r_0 [illegible] $|d_1|$ [illegible] $|d_2|$ [illegible] $-|d_1|$ [illegible] $-|d_2|$ [illegible]

$$LV(\widetilde{r_0}) \le LV(r_0).$$

The following shows that Lemma 2.4 holds also in the nonmonotone case.

COROLLARY 2.6. [illegible] $d_1 = \mathcal{O}(1/h)$ $d_2 = \mathcal{O}(1)$ [illegible] $a = 1 - mh^q$ $0 < m \le \hat{m} < \infty$ [illegible] $LV(r_0) = \mathcal{O}(h^{|q|}|\log(h)|)$

In the following our goal is to find properties of $a(d_1, d_2)$ as a function of the lateral derivatives d_1 and d_2 yielding a limiter-free global third order local reconstruction.

The first condition is that the reconstructing function (2.4) be a well defined smooth function. As shown by Lemma 2.2, we must have

$$a(d_1, d_2) < 1 \qquad \forall d_1, d_2 \in \mathbb{R}. \tag{2.16}$$

Furthermore we shall require

$$a(\lambda, \lambda) = 0 \qquad \forall \lambda \in \mathbb{R}, \tag{2.17}$$

which ensures that in the case $d_1 \approx d_2$ the reconstructing function (2.4) is approximately the linear function; cf. the proof of Lemma 2.2 and the removable singularity.

Let us consider the ansatz

$$a(d_1, d_2) = 1 - 2\frac{|d_1|^q |d_2|^q}{|d_1|^{2q} + |d_2|^{2q}} \tag{2.18}$$

with $q > 0$. Condition (2.17) is trivially verified. Condition (2.16), however, is not satisfied since $a : \mathbb{R}^2 \to [0, 1]$, but we shall overlook this for the moment and turn to verifying the assumption on a in Corollary 2.6. We set $d_1 = M/h$ and $d_2 = N$ and evaluate a:

$$a\left(\frac{M}{h}, N\right) = 1 - 2\frac{|MN|^q}{|M|^{2q} + |N|^{2q} h^{2q}} h^q. \tag{2.19}$$

The bounds

$$\frac{|MN|^q}{|M|^{2q} + |N|^{2q}} \le \frac{|MN|^q}{|M|^{2q} + |N|^{2q} h^{2q}} \le \left|\frac{N}{M}\right|^q \tag{2.20}$$

show that the condition on a is satisfied with an m such that $2\frac{|MN|^q}{|M|^{2q}+|N|^{2q}} \le m \le 2\left|\frac{N}{M}\right|^q$. By Corollary 2.6 we have that in the interesting case $d_1 = \mathcal{O}(1/h)$ and $d_2 = \mathcal{O}(1)$ (or vice versa)

$$LV(r_0) = \mathcal{O}(h^q |\log h|). \tag{2.21}$$

By adjusting the exponent q large enough for the line Ch to lie above $-h^q \log h$, we can construct an LVB reconstruction. Thus we solve for q in

$$-h^q \log h = Ch, \quad C > 0, \tag{2.22}$$

to obtain

$$q = \frac{\log(Ch) - \log(-\log h)}{\log h}. \tag{2.23}$$

Now (2.23) is maximized for $h \in (0, 1)$, yielding

$$q^* = 1 + \frac{1}{C \exp(1)}. \tag{2.24}$$

2.4. The algorithm. In section 2.2 we found the coefficients c, d, and b of (2.11)–(2.13), and in the previous section an ansatz for the parameter a of (2.18) satisfying the assumption on a in Corollary 2.6. The condition (2.16), $a < 1$, was temporarily relaxed to $a \le 1$; the condition (2.17), $a(\lambda, \lambda) = 0$ for all λ, was satisfied ensuring that equal lateral derivatives give the linear function. Now since the coefficient c has the denominator $b - a$, which is zero if and only if $a = 0$, we will relax condition (2.17) in favor of satisfying (2.16). To this end we modify our ansatz (2.18) in the following simple way: Given the cell size h, we set $\text{TOL} = 0.1h^q$ and

$$a = (1 - \text{TOL})\left(1 + \text{TOL} - \frac{2|d_1|^q |d_2|^q + \text{TOL}}{|d_1|^{2q} + |d_2|^{2q} + \text{TOL}}\right). \tag{2.25}$$

The range of (2.25) is $[\text{TOL}(1 - \text{TOL}), 1 - \text{TOL}^2)$. Condition (2.16) is satisfied, and condition (2.17) is relaxed to $a(\lambda, \lambda) = \text{TOL}(1 - \text{TOL})$. Finally, to determine the exponent q we set $C = 1$ in (2.22); then (2.24) gives $q = 1.4$. Of course, using a smaller

C (a larger q) results in a reconstruction procedure of smaller local variation. There might be applications where a different q is preferred (see section 3.1, Example 4); in selecting q there is a trade-off between resolution and robustness. In general, however, changing the exponent from 1.4 is not recommended.

We are usually interested in the third order approximations of the point values at the cell boundaries. Instead of using the reconstructing function (2.4) directly we compute the lateral values conveniently by

$$r_0\left(x_0 \pm \frac{h}{2}\right) = v_0 + ch\eta^{\pm}(a) + dh\eta^{\pm}(b), \tag{2.26}$$

where

$$\eta^{+}(t) = -\frac{\log(1-t)+t}{t^2} \tag{2.27}$$

and

$$\eta^{-}(t) = \frac{(t-1)\log(1-t)-t}{t^2}. \tag{2.28}$$

To avoid two evaluations of each η-function one can exploit the fact that

$$\eta^{\pm}\left(b = \frac{a}{a-1}\right) = (a-1)\eta^{\mp}(a). \tag{2.29}$$

The functions $\eta^{\pm}$ have a removable singularity at zero by putting $\eta^{\pm}(0) = \pm\frac{1}{2}$. For numerical computations it is recommended that you interpolate the values around the removable singularity at zero.

For completeness we describe in quite some detail the use of LDLR in an FV scheme. The scalar conservation law

$$u_t + f(u)_x = 0 \tag{2.30}$$

in integrated form is given by (1.1), which holds for all spatial domains Ω. Leaving time continuous, the FV method discretizes (1.1) in space by requiring it to hold for all computational cells $C_i = [x_{i-1/2}, x_{i+1/2}]$:

$$\frac{\mathrm{d}}{\mathrm{d}t}\bar{u}_i(t) = -\frac{1}{h}\left(f(u(x_{i+1/2},t)) - f(u(x_{i-1/2},t))\right), \tag{2.31}$$

where

$$\bar{u}_i(t) = \frac{1}{h}\int_{x_{i-1/2}}^{x_{i+1/2}} u(x,t)dx$$

is the cell average. The flux $f(u(x_{i+1/2},t))$ is replaced by a numerical flux $\hat{f}_{i+1/2} = \hat{f}(u^-(x_{i+1/2},t), u^+(x_{i+1/2},t))$, with the superscripts $\pm$ indicating that the value is taken from the right or left, respectively. LDLR is not tied to any particular numerical flux function; for example, one can use the Lax–Friedrichs flux

$$\begin{aligned}\hat{f}_{i+1/2} = {} & 0.5\left[f(u^-(x_{i+1/2},t)) + f(u^+(x_{i+1/2},t))\right] \\ & -0.5\alpha\left[u^+(x_{i+1/2},t) - u^-(x_{i+1/2},t)\right],\end{aligned} \tag{2.32}$$

where α is an upper bound of the wave propagation speed. The semidiscrete FV scheme reads

$$\frac{\mathrm{d}}{\mathrm{d}t}\bar{u}_i(t) = -\frac{1}{h}\left(\hat{f}_{i+1/2} - \hat{f}_{i-1/2}\right). \tag{2.33}$$

The reconstruction task is now to obtain the arguments of the numerical flux function to higher order. LDLR determines $u^-(x_{i+1/2})$ by the following steps:

Compute the lateral derivatives $d_1 = (\bar{u}_i - \bar{u}_{i-1})/h$ and $d_2 = (\bar{u}_{i+1} - \bar{u}_i)/h$;
Compute the coefficient $a(d_1, d_2)$ of (2.25) and the coefficients c, d, b of (2.11)–(2.13);
Use (2.26) to find $u^-(x_{i+1/2}) = \bar{u}_i + ch\eta^+(a) + dh\eta^+(b)$.

Similarly the right-sided limit $u^+(x_{i+1/2})$ is computed as follows:

Compute the derivatives $d_1 = (\bar{u}_{i+1} - \bar{u}_i)/h$ and $d_2 = (\bar{u}_{i+2} - \bar{u}_{i+1})/h$;
Compute the coefficient $a(d_1, d_2)$ of (2.25) and the coefficients c, d, b of (2.11)–(2.13);
Use (2.26) to find $u^+(x_{i+1/2}) = \bar{u}_{i+1} + ch\eta^-(a) + dh\eta^-(b)$.

At this point we want to emphasize that LDLR requires only the evaluation of universal, fixed formulas (2.25), (2.11)–(2.13), and (2.26) regardless of the in-data, resulting in simple coding and efficiency (see section 3.3).

Finally, third order in time is accomplished by integrating the system (2.33) by the third order SSP-Runge–Kutta method from [2] under the stability condition

$$\lambda\frac{\Delta t}{h} = \mathrm{CFL} \le 1, \tag{2.34}$$

where λ is an upper bound of the maximal propagation speed.

For systems of conservation laws the reconstruction procedure is applied componentwise in this paper.

3. Numerical experiments. In this section we compute several numerical examples to test the local double logarithmic reconstruction and compare it to those of the LHHR, ENO3, and WENO3 techniques.

All LDLR computations except one in Example 4 in the next section are performed with the exponent $q = 1.4$; cf. section 2.3–2.4.

3.1. Convergence and accuracy tests.

Example 1. A convergence test. We test the rate of convergence for the LDLR with a [illegible] as the governing equation. Consider the Euler system

$$u_t + f(u)_x = 0$$

with

$$u = (\rho, \rho v, E), \quad f(u) = (\rho v, \rho v^2 + p, v(E + p)),$$

and

$$p = (\gamma - 1)\left(e - \rho\frac{u^2}{2}\right), \quad \gamma = 1.4.$$

With initial data $\rho(0, x) = 1 + 0.2\sin(x)$, $v \equiv 1$, and $p \equiv 1$ the Euler system has the solution $\rho(t, x) = 1 + 0.2\sin(x - t)$, $v \equiv 1$, and $p \equiv 1$ (see [7]). We compute approximations to the solution at $t = 1$, on the interval $-\pi \le x \le \pi$ with periodic boundary

TABLE 3.1
Euler convergence test: LDLR.

N	L_1-error	L_1-order	L_∞-error	L_∞-order
10	1.77e-02	-	4.00e-03	-
20	2.15e-03	3.04	5.43e-04	2.88
40	2.61e-04	3.04	6.87e-05	2.98
80	3.20e-05	3.03	8.62e-06	2.99
160	3.97e-06	3.01	1.09e-06	2.99
320	4.94e-07	3.01	1.36e-07	3.00
640	6.16e-08	3.00	1.70e-08	3.00
1280	7.69e-09	3.00	2.13e-09	3.00
2560	9.60e-10	3.00	2.65e-10	3.01

TABLE 3.2
Euler convergence test: LHHR.

N	L_1-error	L_1-order	L_∞-error	L_∞-order
10	2.28e-01	-	5.01e-02	-
20	9.78e-02	1.22	2.85e-02	0.81
40	1.21e-02	3.02	6.13e-03	2.22
80	1.74e-03	2.79	2.04e-03	1.59
160	3.02e-04	2.53	6.12e-04	1.74
320	5.78e-05	2.39	2.12e-04	1.52
640	1.16e-05	2.31	7.19e-05	1.56
1280	2.43e-06	2.26	2.51e-05	1.52
2560	5.18e-07	2.23	8.86e-06	1.50

conditions, by local double logarithmic reconstruction of the conserved variables in the Van Leer upwind scheme [15]. The error in density is displayed in Table 3.1. Full third order convergence is clearly verified. Note that the third order is reached also at local extrema, as seen in the L_∞-norm. This is in contrast to monotone reconstructions like MUSCL [14], LHHR, and LLR, which suffer from clipping of extrema leading to reduced order of accuracy clearly visible in the L_∞-norm. We recomputed the convergence test with the LHHR reconstruction; the result is shown in Table 3.2.

EXAMPLE 2. Linear advection. The equation

$$u_t + u_x = 0, \quad -1 < x < 1,$$

is solved with periodic boundary condition and initial data as proposed in [4]. The signal consists of a smooth and narrow combination of Gaussians, a square wave, a triangle wave, and a half ellipse. We refer to [4] for the details. LDLR, LHHR, and ENO3 approximations at $t = 20$, obtained with 200 and 400 cells under CFL $= 0.6$, are shown in Figure 3.1. Considering the 200-cell computations, since LHHR does not reconstruct extrema, this approximation is far less resolved than the other two. The ENO3 approximation shows less symmetry than the other approximations, while the LDLR approximation clearly takes on negative values. Looking at the 400-cell computation, the LDLR shows the best resolution especially at corners, and the undershoots are negligible. The ENO3 is still not symmetric, and the LHHR is the most diffusive approximation.

EXAMPLE 3. Burgers' equation. The nonlinear equation

$$u_t + \left(\frac{u^2}{2}\right)_x = 0$$

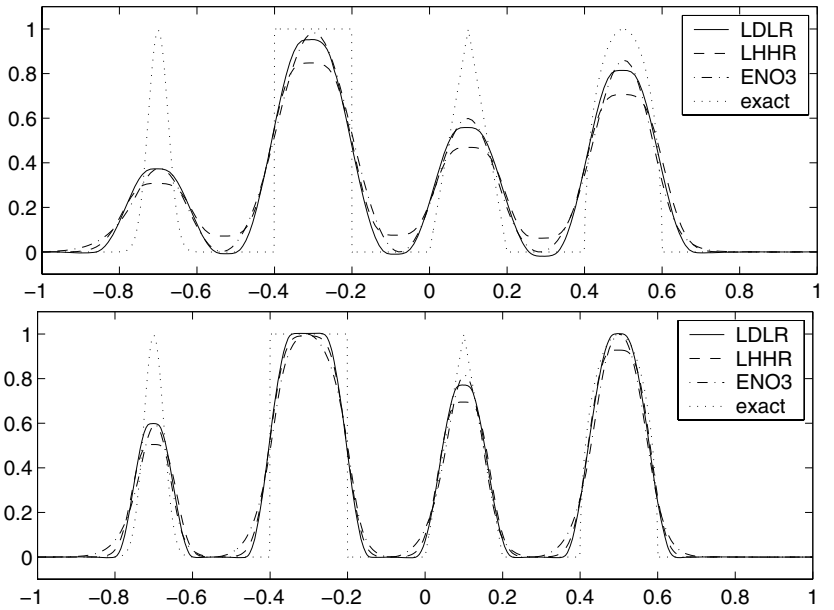


FIG. 3.1. *Advection. Upper row:* 200 *cells. Lower row:* 400 *cells.*

with initial data $u_0(x) = 1 + 1/2\sin(\pi(x-1))$ and 2-periodic boundary conditions is solved under CFL = 0.6. In Figure 3.2 the LDLR approximation at $t = 1.5$ and the computed total variation are depicted for 40, 80, and 160 cells. Despite the reconstruction of local extrema, we observe that the total variation is bounded by $\mathrm{TV}(u_0)$ for all times. Note that a monotone decreasing variation cannot be expected for a third order scheme [13]. Especially on the coarsest mesh we also observe a slight overshoot in two cells behind the right-moving shock wave. This effect is due to the bounded nonzero variation of the double logarithmic reconstruction. Recall that, in contrast to standard methods, unlimited reconstruction is applied across the discontinuity. The right column in Figure 3.2 shows that the overshoot vanishes as the mesh is refined and does not cause any instability. We conclude that the 40-point simulation is underresolved.

4. Buckley–Leverett equation. The nonlinear and nonconvex scalar Buckley–Leverett problem

$$u_t + \left(\frac{4u^2}{4u^2 + (1-u)^2}\right)_x = 0$$

is solved using local double logarithmic reconstruction of the conserved variable and the Lax–Friedrich flux (2.32). The initial data is $u = 1$ for $-\frac{1}{2} \le x \le 0$ and $u = 0$ elsewhere. Approximations using 120 and 360 cells with CFL = 0.7 are shown in Figure 3.3. We note that there is a slight "overshoot" in the compound wave where the rarefaction profile meets the shock. Such compound waves, which are typical for nonconvex problems, are numerically difficult to treat due to reduced stability properties. More diffusive schemes can compensate for this sensitivity in the problem and may give better results!

A higher value of the variation controlling exponent $q = 2$ in (2.25) decreases the total variation (increases the diffusion) of the LDLR scheme and in fact damps out the overshoot; see Figure 3.4.

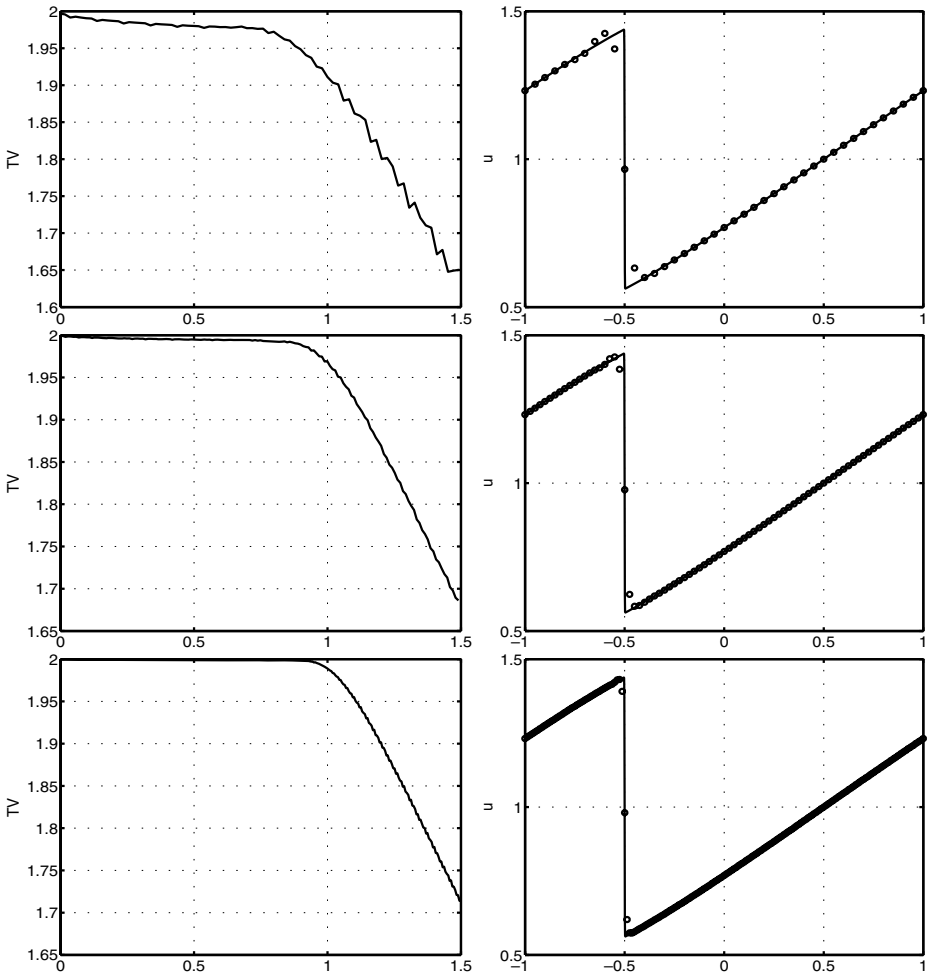


FIG. 3.2. *Burgers' equation: LDLR approximation.*

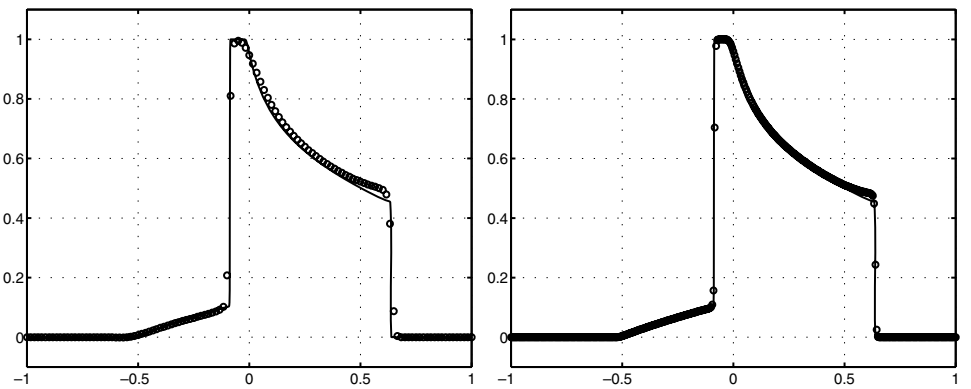


FIG. 3.3. *Buckley–Leverett: LDLR approximation.*

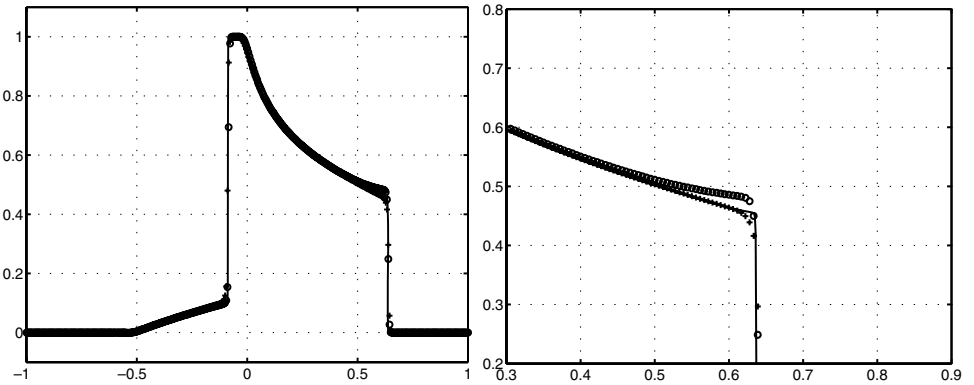


FIG. 3.4. *Buckley–Leverett: LDLR;* $q = 1.4$ (o), $q = 2.0$ (+).

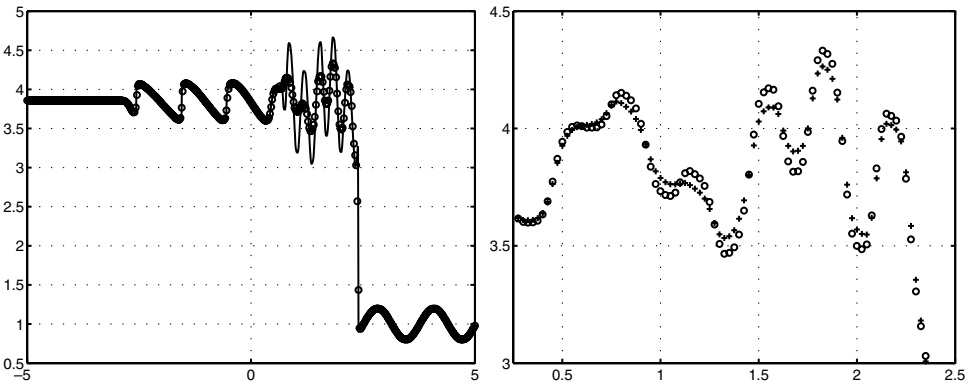


FIG. 3.5. *Shu–Osher shock-acoustic problem; LDLR* (o) *and LHHR* (+) .

3.2. One-dimensional Euler system.

EXAMPLE 5. Shu–Osher shock-acoustic problem. In this test problem, proposed by Shu and Osher [12], a supersonic shock wave moves into a smooth acoustic entropy wave. The initial data are

$$u(x,0) = (\rho, v, p) = \begin{cases} (3.857143, 2.629369, 10.333333), & -5 \le x < -5, \\ (1.0 + 0.2\sin(5x), 0.0, 1.0), & -4 \le x \le 5. \end{cases}$$

Behind the shock the acoustic wave is amplified. We run the experiment with componentwise reconstruction of the conserved variables in connection with Van Leer flux splitting. The number of cells is $N = 400$ and CFL $= 0.8$ for the computations. Depicted in Figure 3.5 are the LDLR approximation and a comparison between the LDLR and LHHR at $t = 1.8$. We observe that LDLR better resolves the waves behind the shock—especially near $x = 1.1$, where LHHR damps out two local extrema.

EXAMPLE 6. Sod problem. The initial data are

$$u(x,0) = (\rho, v, p) = \begin{cases} (1.0, 0.0, 1.0), & 0 \le x < 0.5, \\ (0.125, 0.0, 0.1), & 0.5 \le x \le 1. \end{cases}$$

We compute approximations using 200 cells with reconstructions of the conserved variables and the Lax–Friedrichs numerical flux. For ENO3 the constraint CFL $= 0.6$

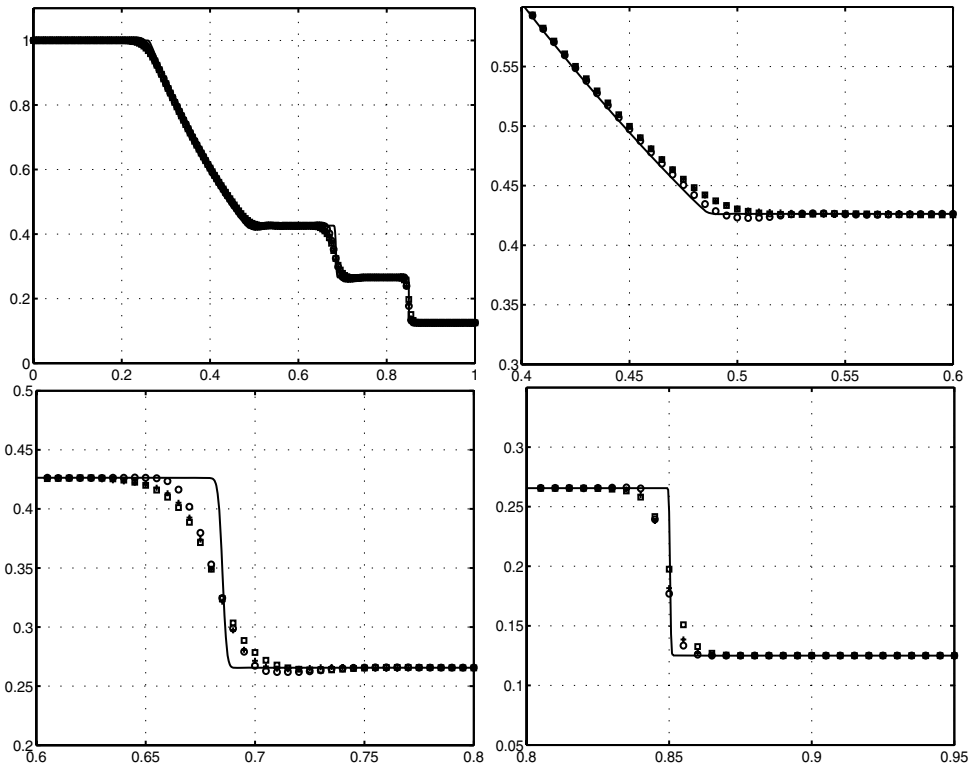


FIG. 3.6. *Sod problem; LDLR* (o) *, ENO*3 (+)*, and WENO*3 (□).

is used, and for LDLR and WENO3 we set CFL = 0.8. The approximations at $t = 0.2$ are shown in Figure 3.6. The local double logarithmic reconstruction introduces some very slight undershoots but on the other hand, resolves, the contact discontinuity in fewer cells than ENO and WENO; also the shock appears somewhat sharper.

EXAMPLE 7. Lax problem. The initial data are

$$u(x,0) = (\rho, v, p) = \begin{cases} (0.445, 0.698, 3.528), & 0 \le x < 0.5, \\ (0.5, 0.0, 0.571), & 0.5 \le x \le 1. \end{cases}$$

The computations with the Van Leer flux split and LDLR and ENO3 reconstructions of the conserved variables are on a 200 grid and subject to the constraint CFL = 0.8 for LDLR and WENO3 and CFL = 0.6 for ENO3. The approximations at $t = 0.13$ are displayed in Figure 3.7. We note that the LDLR approximation captures the discontinuities better than ENO3 and WENO3, especially the contact, and that no over- or undershoots are present.

EXAMPLE 8. Shockblast interaction. This experiment was proposed as a benchmark in [16]. The initial data are

$$u(x,0) = (\rho, v, p) = \begin{cases} (1.0, 0.0, 10^3), & 0 \le x < 0.1, \\ (1.0, 0.0, 10^{-2}), & 0.1 \le x < 0.9, \\ (1.0, 0.0, 10^2), & 0.9 \le x \le 1. \end{cases}$$

Reflective boundary conditions are used. Numerical solutions on 400 and 800 cell grid are computed with LDLR and LHHR applied on the Van Leer fluxes under CFL

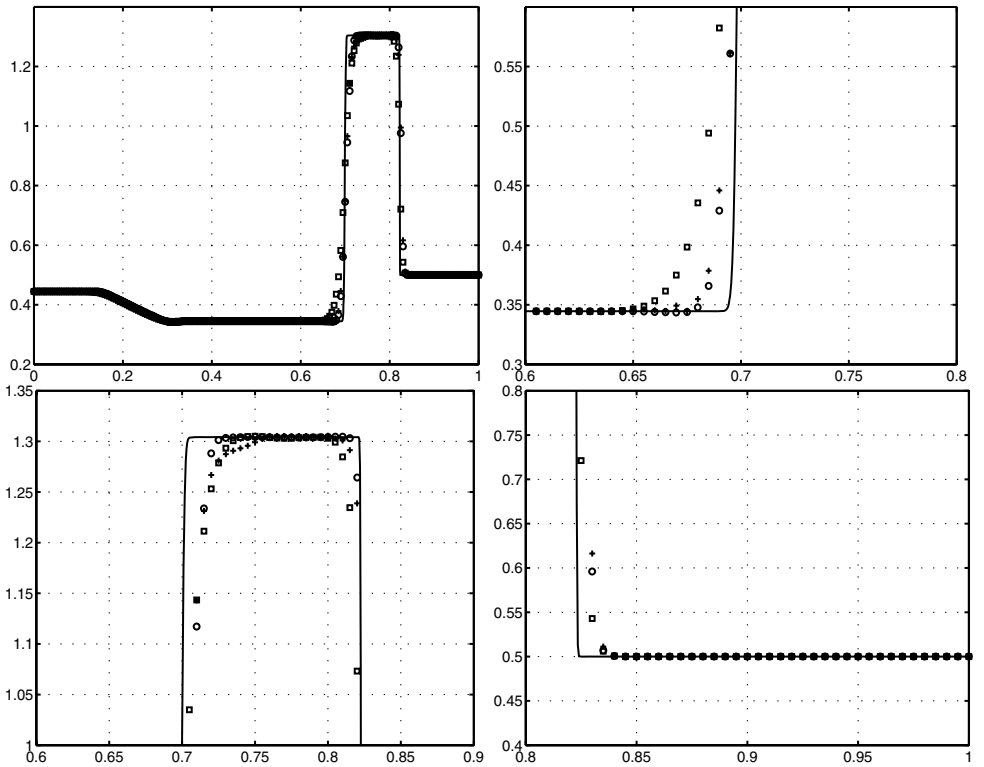


FIG. 3.7. *Lax problem; LDLR* (o)*, ENO3* (+)*, and WENO3.* (□)

$= 0.8$. In Figure 3.8 the approximations at $t = 0.038$ are displayed. For the 800-cell computations we observe that the lower peak is better resolved with LDLR; the LDLR approximations are, despite the full reconstruction of extrema, essentially free from over/undershoots and oscillations.

3.3. Computational cost. For a sound investigation of the efficiency of LDLR compared to other methods a large variety of problems should be tested; such tests are outside the scope of this paper. We are well aware that the performance of different methods depends on coding and hardware issues. Still we will need a "benchmark" method, LHHR, to set the LDLR results in perspective. In the present case things simplify a bit as both methods (LDLR and LHHR) rely on the same building principle and are of the same (formal) order. In fact, the basic FV code is exactly the same with only the reconstruction subroutine exchanged. Nevertheless the results presented below shall be taken as only a rough indication.

The tests were performed on a 500 MHz SunBlade 100 work station with 512 MB RAM using MATLAB Version 5.3.1.29215a (R11.1) on SOL2. The implementation avoids to a large extent for-loops and if-statements by extensive use of the dot and colon notation and by using logical arrays. The Sod and Lax problems are used as test problems with 200 cells and CFL $= 0.8$, as described in Examples 6 and 7 in section 3.2. Each example was run ten times in a row with LDLR and LHHR in sequence, and the average times were recorded as well as the number of flops used. The LDLR implementation exploited the symmetry in the η-functions as described in section 2.4. The same type of interpolation around the removable singularities in

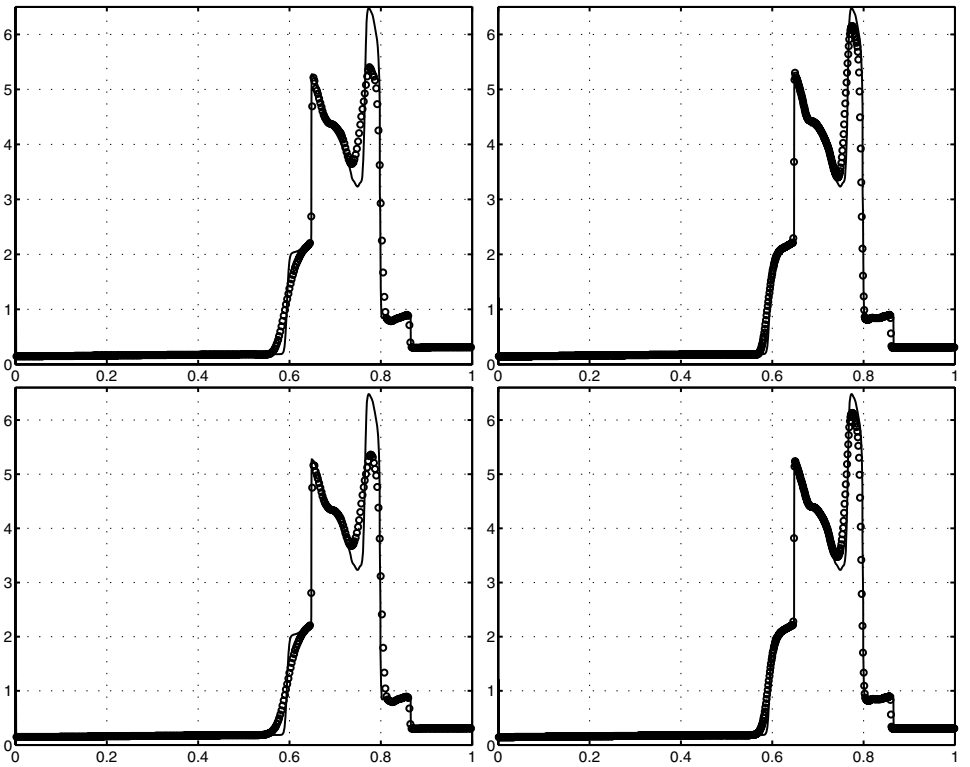


FIG. 3.8. *Shock-blast interaction. Upper row: LDLR* 400/800 *cells. Lower: LHHR* 400/800 *cells.*

TABLE 3.3
Computational cost comparison.

Sod problem; Lax–Friedrichs flux.

	Time (s)	Flops
LHHR	4.9763	12609198
LDLR	4.0252	16271849
LDLR*	4.8292	11627822

Lax problem; van Leer splitting.

	Time (s)	Flops
LHHR	9.5043	19481599
LDLR	7.7880	25296270
LDLR*	9.1732	17623662

the LDLR η-functions was used for the LHHR η-function. The LDLR was used in two ways: The results in the LDLR row in Table 3.3 were obtained by using LDLR also on constant states; the results in the LDLR* row were obtained as advised for LHHR in [6], that is, no reconstruction if both lateral derivatives are below the cell size squared. The results in Table 3.3 indicate that the computational cost for both schemes is comparable and that the LDLR, despite using two extra parameters, is not more expensive. For a comparison to ENO3 we refer to [6], where it is stated that the cost of LHHR is a bit higher than ENO3, but, on the other hand, LHHR is stable and accurate for high CFL numbers which reduce computational cost. The same stands true for LDLR.

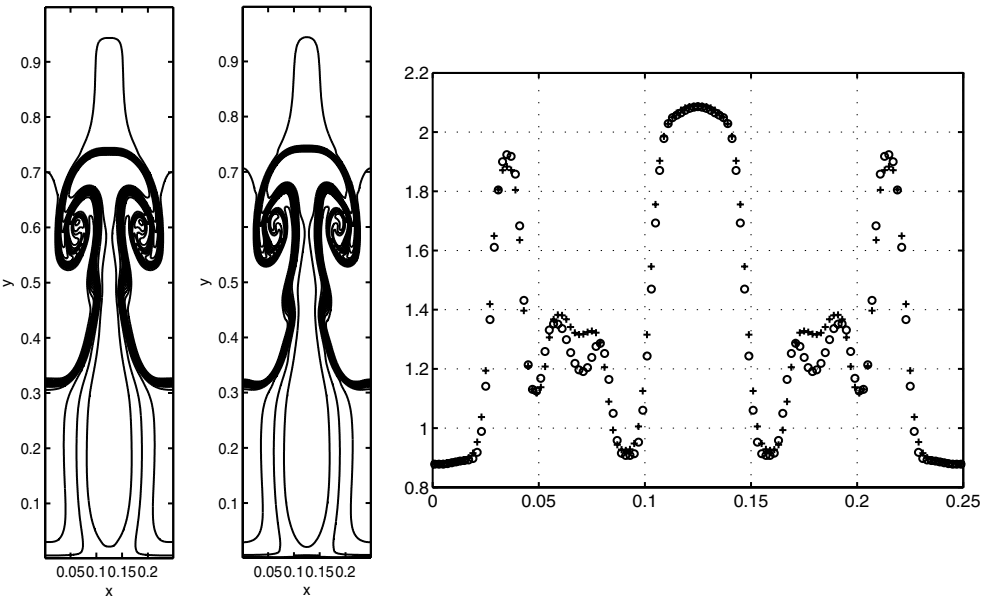


FIG. 3.9. *Rayleigh–Taylor instability. Density contours from* 0.952269 *to* 2.14589 *by step* 0.075. *From left to right: LHHR, LDLR, and vortex cut with LDLR* (o) *and LHHR* (+).

3.4. Two-dimensional (2d) Euler system. The 2d tests are computed using the dimension-by-dimension approach. We compute the numerical examples using the third order reconstructions on the conserved variables in Van Leer's upwind scheme.

[illegible] 9. Rayleigh–Taylor instability. See, for example, [11] and references therein for other simulations of this problem. The governing equations are the Euler equations with $\gamma = 5/3$ and source terms ρ and ρv added to the third and fourth equations, respectively. The computational domain is $[0, 1/4] \times [0, 1]$; the interface is located at $y = 0.5$ with initial data $\rho = 2$, $u = 0$, $p = 2y + 1$, and $v = -0.025c \cos 8\pi x$ for $0 \le y < 1/2$ and $\rho = 1$, $u = 0$, $p = y + 3/2$, and $v = -0.025c \cos 8\pi x$ for $1/2 \le y \le 1$. Reflective boundary conditions are imposed at the left and right boundary; at the top boundary we set $\rho = 1$, $u = 0$, $p = 2.5$, and $v = 0$, and on the bottom boundary $\rho = 2$, $u = 0$, $p = 1$, and $v = 0$. In Figure 3.9 LHHR and LDLR simulations on a 125×500 mesh under CFL $= 0.9$ at the final time $t = 1.95$ are depicted. A cut through the vortex at $y = 0.6$ shows that the local extrema are much better resolved for LDLR, especially near $x = 0.07$ and $x = 0.18$.

[illegible] 10. Double Mach reflection. For a detailed discussion of this problem, see [16]. Initially, on the computational domain $[0, 4] \times [0, 1]$, a Mach 10 shock is located at a 60° angle with the x-axis at $(x, y) = (1/6, 0)$. The boundary conditions are as follows: for $0 \le x \le 1/6$ exact postshock conditions are imposed, and for $1/6 < x \le 4$ reflective conditions are used; at the top boundary the flow values are set to describe the motion of the shock, and at the left and right boundaries inflow and outflow conditions, respectively, are used. To the right of the shock the fluid initially has a density of 1.4 and pressure of 1. We perform the computations with LDLR and LHHR on a 2000×500 grid under CFL $= 0.7$. Density contours at $t = 0.2$ are shown in Figure 3.10. We note the difference in resolution for LDLR and LHHR of the complex flow. In particular, the partially rolled up slip line and the local extrema at $(x = 2.62, y = 0.065)$ are sharper by the LDLR approximation.

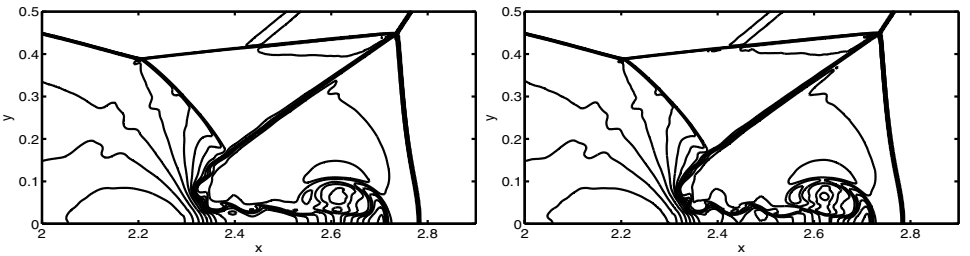


FIG. 3.10. *Double Mach reflection. Density contours from* 1.5 *to* 22.9705 *by step* 0.7156833. *Left: LHHR. Right: LDLR.*

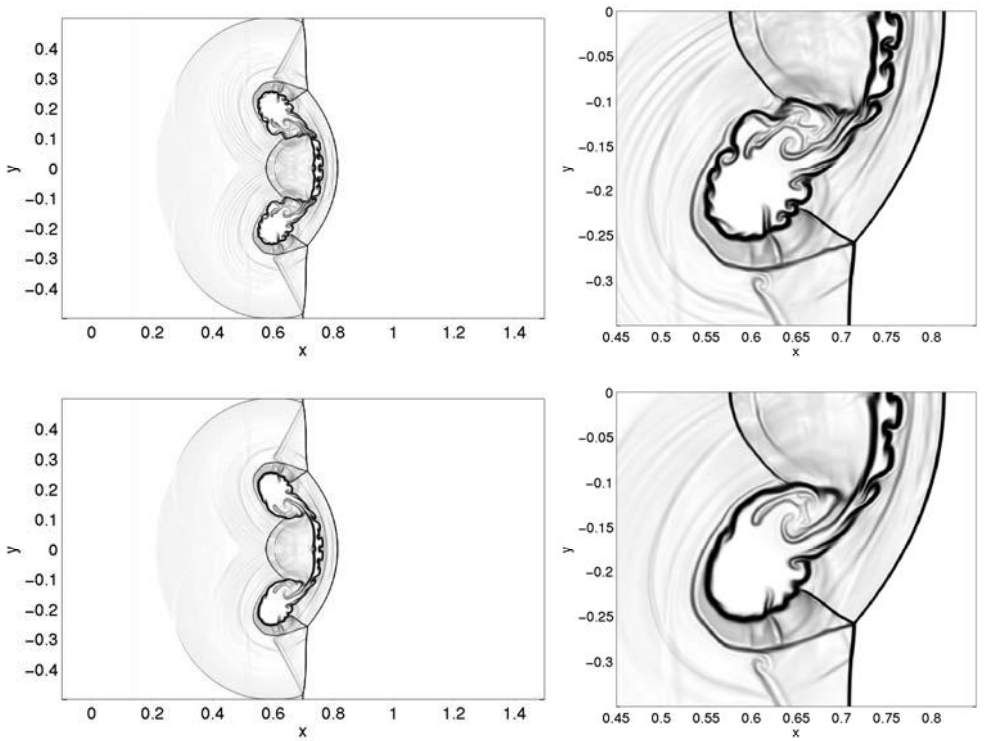


FIG. 3.11. *Shock-bubble, emulated Schlieren images at* $t = 0.2$. *Upper row: LDLR. Lower: LHHR.*

FIG. 3.11. Shock-bubble interaction. For a problem description and the precise data we refer to [9] and the web-page [17]. The LDLR and LHHR approximations are computed on a 1600×500 grid with reflective boundary condition at $y = 0$ under CFL $= 0.9$. In Figure 3.11 the results at $t = 0.2$ are displayed, and Figure 3.12 shows the approximations at $t = 0.4$. Comparing the LDLR and LHHR approximations, we observe more resolved contact discontinuities with the local double logarithmic method, resulting in considerably more vortex structure.

FIG. 3.12. A shock vortex interaction. In this experiment we simulate a shock wave in air diffracting down a step. The domain of interest is the unit square. On the bottom and the lower half of the left side a reflective boundary condition models a solid wall. The upper and right side are absorbing (artificial) boundaries. On the

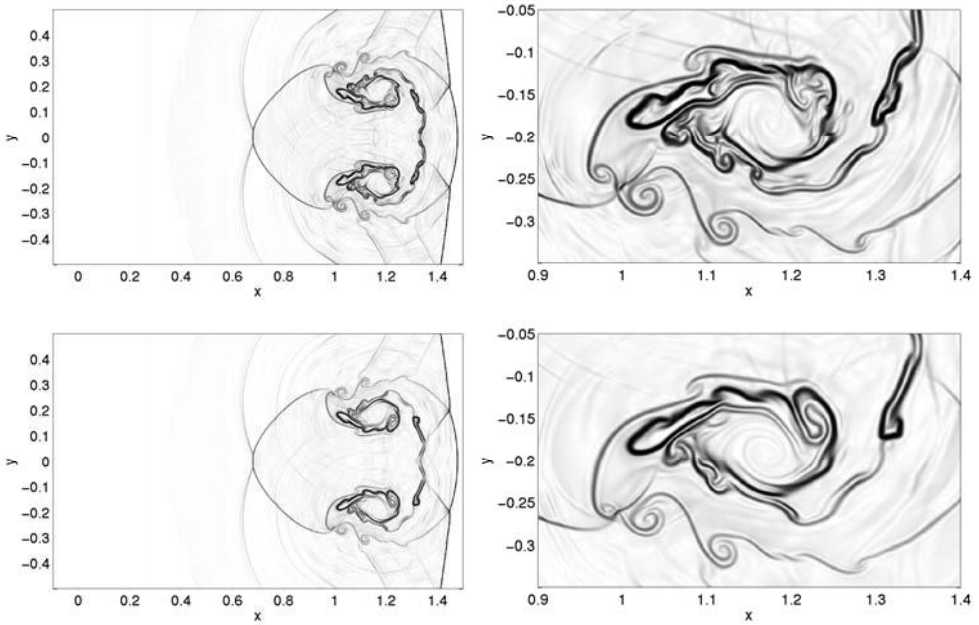


FIG. 3.12. *Shock-bubble, emulated Schlieren images at* $t = 0.4$. *Upper row: LDLR. Lower: LHHR.*

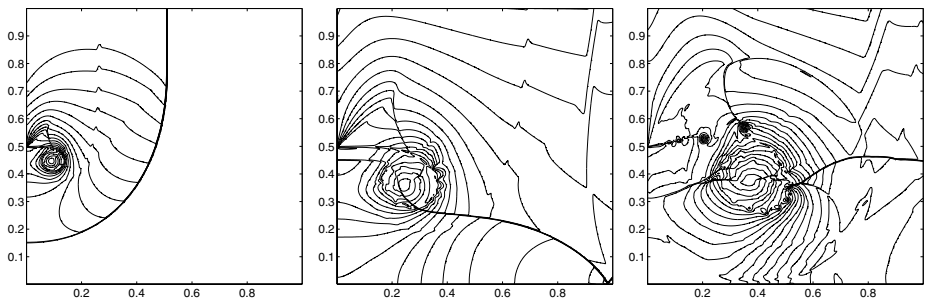


FIG. 3.13. *Shock-vortex. Density contours illustrating time evolution.*

upper half of the left side we put inflow conditions with $\rho = 1$, $p = 3$ and the flow speed at an angle of $\alpha = \pi/10$ connected to $\rho = 1$, $u = v = 0$, $p = 1$ inside the square by a single shock. The computation is run up to $t = 1$ on an 800×800 mesh under CFL $= 0.9$ with the reconstruction applied to Van Leer's partial fluxes. In Figure 3.13 the time evolution is illustrated. First we see the diffraction of the shock down the step and the formation of a large vortex in the wake. Next the shock is reflected on the bottom to pass through the vortex. A Schlieren animation of the flow is available on the web page [18]. The last movie frame is reproduced in Figure 3.14. Figure 3.15 shows enlarged details in the south-east part of the main vortex. Here the third order reconstruction of local extrema shows its effect: In the LDLR simulation we observe a sheet of several new vortices which are not resolved in the LLR simulation.

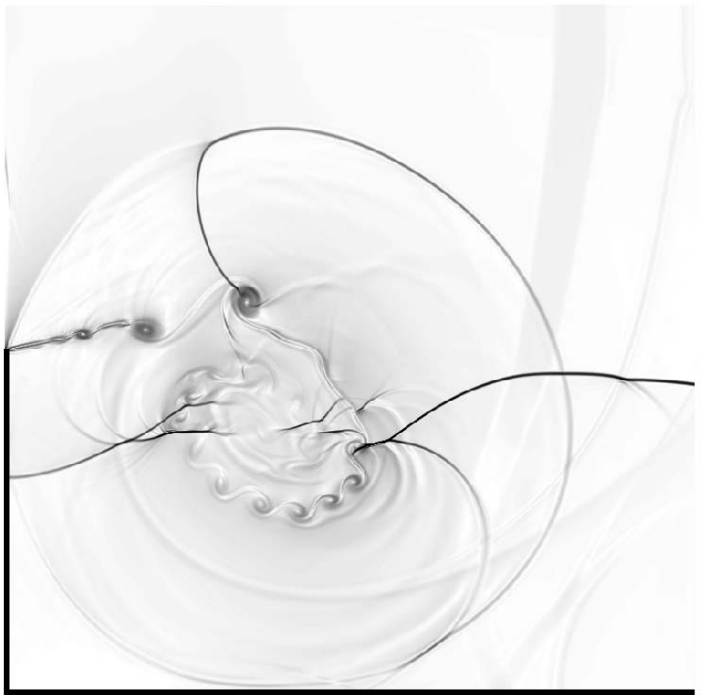


FIG. 3.14. *Shock-vortex. LDLR, emulated Schlieren image at* $t = 1.0$.

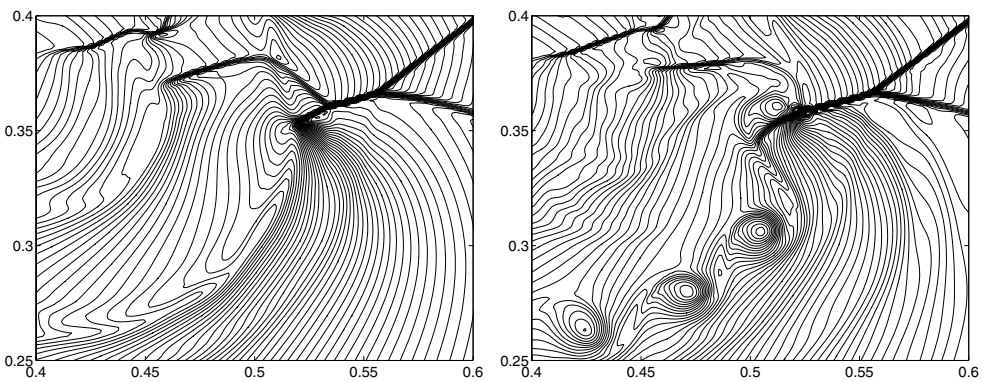


FIG. 3.15. *Shock-vortex, density contours. Left: LLR. Right: LDLR.*

4. Higher order extension. We are also interested in the local double logarithmic ansatz for use in a hybrid coupling to the fifth order DLR [1]. As described in [1], the DLR coefficients for fifth order do not always exist. In these cases we instead compute the LDLR coefficients for the third order. Note that the ansatz function is the same for the DLR and LDLR; the coefficients are, however, calculated with different objectives. The resulting scheme is applied to the test problem in Example 5 in section 3.2. The smoothness indicator in [1] is applied to the local characteristic projections of the required numerical derivatives of the conserved variables. We use

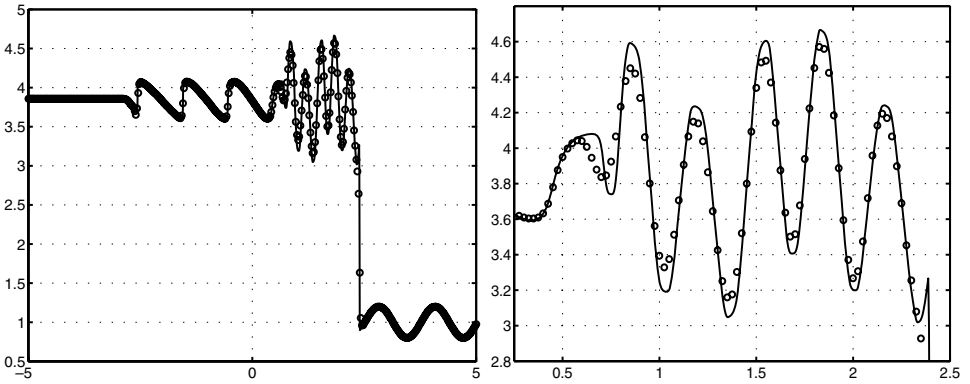


FIG. 4.1. *Shu–Osher shock-acoustic problem. DLR approximation.*

the Van Leer flux splitting with CFL = 0.8 and the same number of cells as in Example 5, section 3.2. The third order SSP–Runge Kutta method is used to advance in time. The result, displayed in Figure 4.1, shows that the waves behind the shock are much better resolved than with the third order schemes.

5. Concluding remarks. The developed reconstruction method uses an ansatz function of five parameters to get third order accuracy, leaving two parameters for symmetry and to control the variation of the reconstructing function. The numerical experiments conducted in this paper indicate that it is possible to obtain approximations to the solutions of hyperbolic conservation laws by a third order scheme on a 3-point spatial stencil without using explicit limiters on the slope information, provided that the local variation is controlled by other means. Moreover we have shown that the proposed reconstruction LDLR is not more expensive to compute, but is more accurate than, comparable third order methods.

Appendix A. LDHR—An extension of LHHR. The main reason for introducing the logarithmic ansatz is that the nonlinear system arising for a fifth order version is easily solvable in contrast to a hyperbolic ditto. However, for the third order version introduced in this paper the coefficients needed appear linearly. Therefore it is possible to extend the LHHR method [6] to reconstruct local extrema. In the following we present this local double hyperbolic reconstruction (LDHR).

The reconstructing function in the reference cell C_0 has the form

$$r_0(x) \sim A + \frac{B}{x+C} + \frac{D}{x+E}.$$

We define for $x \in C_0$

$$\phi_0(x) = -\frac{ch^2}{a^2}\frac{1}{\left[x - x_0 - \frac{h}{2}\left(\frac{2}{a} - 1\right)\right]} - \frac{dh^2}{b^2}\frac{1}{\left[x - x_0 - \frac{h}{2}\left(\frac{2}{b} - 1\right)\right]}, \tag{A.1}$$

where a, b, c, and d are the parameters, and let as for LDLR

$$r_0(x) = v_0 + \phi_0(x) - \frac{1}{h}\int_{C_0} \phi_0(\xi)d\xi \tag{A.2}$$

be the reconstructing function.

We satisfy the condition of Lemma 2.1 by solving for c and d, yielding

$$c = \frac{(a-1)^2(d_2(b-1)^2 - d_1)}{(a-2+b)(b-a)}, \tag{A.3}$$

$$d = d_1 - c. \tag{A.4}$$

Symmetry again implies $b = a/(a-1)$, and we get the expression for the local variation

$$LV(r_0) = h\frac{1-a}{(a-1)^2+1}|d_1 + d_2|. \tag{A.5}$$

The local variation in case of a discontinuity and with $a = 1 - h^q$ is of $\mathcal{O}(h^q)$. So we can, for example, use the parameter a given in (2.25) with $q = 1$ and TOL=$0.1h$ to get a LVB method. For computational purposes we use the lateral functions

$$\eta^+(t) = \frac{(t-1)\log(1-t) - t}{t^2(t-1)} \tag{A.6}$$

and

$$\eta^-(t) = \frac{\log(1-t) + t}{t^2} \tag{A.7}$$

and let

$$r_0\left(x_0 \pm \frac{h}{2}\right) = v_0 + ch\eta^{\pm}(a) + dh\eta^{\pm}(b). \tag{A.8}$$

Numerical tests show that this double hyperbolic reconstruction behaves essentially as the LDLR developed and tested in this paper.

REFERENCES

[1] R. Artebrant and H. J. Schroll, *Conservative logarithmic reconstructions and finite volume methods*, SIAM J. Sci. Comput., 27 (2005), pp. 294–314.

[2] S. Gottlieb, C. W. Shu, and E. Tadmor, *Strong stability-preserving high-order time discretization methods*, SIAM Rev., 43 (2001), pp. 89–112.

[3] A. Harten, B. Engquist, S. Osher, and S. R. Chakravarthy, *Uniformly high order accurate essentially non-oscillatory schemes*, III, J. Comput. Phys., 71 (1987), pp. 231–303.

[4] G.-S. Jiang and C.-W. Shu, *Efficient implementation of weighted ENO schemes*, J. Comput. Phys., 126 (1996), pp. 202–228.

[5] X.-D. Liu, S. Osher, and T. Chan, *Weighted essentially nonoscillatory schemes*, J. Comput. Phys., 115 (1994), pp. 200–212.

[6] A. Marquina, *Local piecewise hyperbolic reconstruction of numerical fluxes for nonlinear scalar conservation laws*, SIAM J. Sci. Comput., 15 (1994), pp. 892–915.

[7] J. Qiu and C.-W. Shu, *On the construction, comparison, and local characteristic decomposition for high-order central WENO schemes*, J. Comput. Phys., 183 (2002), pp. 187–209.

[8] J. R. Rice, *The Approximation of Functions*, Vol. 1, Addison–Wesley, Reading, MA, 1964.

[9] H. J. Schroll, *Relaxed high resolution schemes for hyperbolic conservation laws*, J. Sci. Comput., 20 (2004), pp. 251–279.

[10] S. Serna and A. Marquina, *Power ENO methods: A fifth-order accurate weighted power ENO method*, J. Comput. Phys., 194 (2004), pp. 632–658.

[11] J. Shi, Y.-T. Zhang, and C.-W. Shu, *Resolution of high order WENO schemes for complicated flow structures*, J. Comput. Phys., 186 (2003), pp. 690–696.

[12] C. W. Shu and S. Osher, *Efficient implementation of essentially non-oscillatory shock-capturing schemes*, II, J. Comput. Phys., 83 (1989), pp. 32–78.

[13] E. Tadmor, *Numerical viscosity and the entropy condition for conservative difference schemes*, Math. Comp., 43 (1984), pp. 369–381.
[14] B. van Leer, *Towards the ultimate conservative difference scheme* V. *A second order sequel to Godunov's method*, J. Comput. Phys., 32 (1979), pp. 101–136.
[15] B. van Leer, *Flux-vector splitting for the Euler equations*, in Lecture Notes in Phys. 170, E. Krause, ed., Springer, Berlin, 1982, pp. 507–512.
[16] P. Woodward and P. Colella, *The numerical simulation of two-dimensional fluid flow with strong shocks*, J. Comput. Phys., 54 (1984), pp. 115–173.
[17] H. Holden, *A shock-bubble interaction*, www.math.ntnu.no/∼holden/fronttrack/gas/sb/
[18] A. Schroll, *Diffraction of a shock wave at a corner*, www.maths.lth.se/na/staff/achim/research

SIAM J. SCI. COMPUT.
Vol. 28, No. 1, pp. 382–402
© 2006 Society for Industrial and Applied Mathematics

AN ADAPTIVE HYBRID FEM/FDM METHOD FOR AN INVERSE SCATTERING PROBLEM IN SCANNING ACOUSTIC MICROSCOPY*

LARISA BEILINA† AND CHRISTIAN CLASON‡

Abstract. Scanning acoustic microscopy based on focused ultrasound waves is a promising new tool in medical imaging. In this work we apply an adaptive hybrid FEM/FDM (finite element methods/finite difference methods) method to an inverse scattering problem for the time-dependent acoustic wave equation, where one seeks to reconstruct an unknown sound velocity $c(x)$ from a single measurement of wave-reflection data on a small part of the boundary, e.g., to detect pathological defects in bone. Typically, this corresponds to identifying an unknown object (scatterer) in a surrounding homogeneous medium.

The inverse problem is formulated as an optimal control problem, where we use an adjoint method to solve the equations of optimality expressing stationarity of an associated augmented Lagrangian by a quasi-Newton method. To treat the problem of multiple minima of the objective function, the optimization procedure is first performed on a coarse grid to smooth the high frequency error, generating a starting point for optimization steps on successively refined meshes. Local refinement based on the results of previous steps will improve computational efficiency of the method.

As the main result then, an a posteriori error estimate is proved for the error in the Lagrangian, and a corresponding adaptive method is formulated, where the finite element mesh is refined from residual feedback. The performance of the adaptive hybrid method and the usefulness of the a posteriori error estimator for problems with limited boundary data are illustrated in three dimensional numerical examples.

Key words. inverse scattering, parameter identification, a posteriori error estimator, adaptive mesh refinement, transient wave equation, hybrid finite element/difference method

AMS subject classifications. 65M32, 65M50

DOI. 10.1137/050631252

1. Introduction.

Inverse scattering is a rapidly expanding area of computational mathematics with a wide range of applications including nondestructive testing of materials, shape reconstruction, nonmicroscopic ultrasound imaging, subsurface depth imaging of geological structures, and seismic prospection. The current work is devoted to an adaptive hybrid finite element/difference method (FEM/FDM) for an inverse scattering problem for the time-dependent three dimensional acoustic wave equation, with a special focus on the application of scanning acoustic microscopy (cf. [12]) in medical imaging. This problem takes the form of reconstructing a parameter from a single set of boundary displacement data measured using acoustic microscopy. Since the shear component of the elastic wave speed is much less than the longitudinal component in biological materials (by several orders of magnitude for soft tissue), an approximate common mathematical model for the displacement in water and sample is a scalar wave equation with (longitudinal) wave speed $c = \sqrt{(\lambda + 2\mu)/\rho}$, where ρ is the density and λ, μ are the Lamé constants of linear elasticity. More precisely, we will consider the problem of obtaining quantitative

*Received by the editors May 11, 2005; accepted for publication (in revised form) August 16, 2005; published electronically March 24, 2006.

http://www.siam.org/journals/sisc/28-1/63125.html

†Department of Mathematics, University of Basel, CH–4051 Basel, Switzerland (Larisa.Beilina@unibas.ch). The work of this author was supported by Swiss National Foundation grant 200020-105135.

‡Department of Mathematics, Technische Universität München, D–85748 Garching bei München, Germany (clason@ma.tum.de).

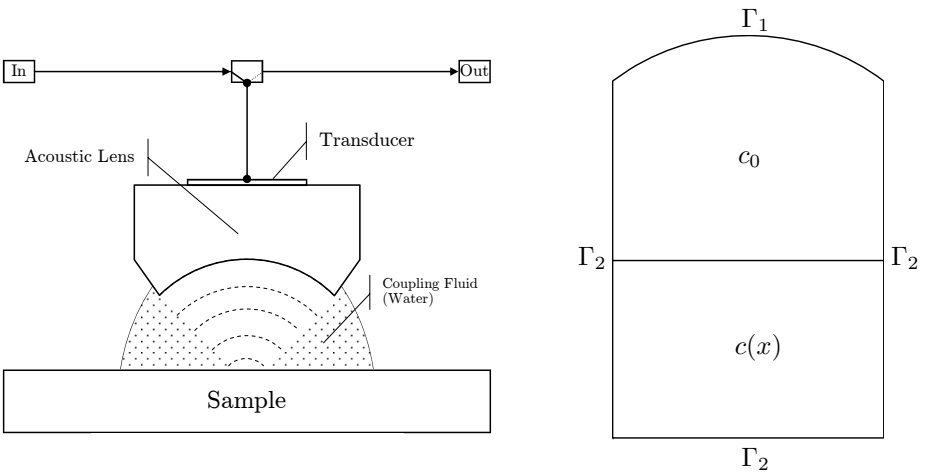


FIG. 1.1. *Sketch of a scanning acoustic microscope (left) and a cut of the corresponding three dimensional computational domain (right). Here,* c_0 *corresponds to the region filled with water,* $c(x)$ *to the sample, and* Γ_1 *to the lens boundary where a pulse is initiated and reflected echoes are measured.*

elasto-mechanical parameters of human bone by identifying the coefficient $c(x)$ in a bounded domain $\Omega \subset \mathbb{R}^3$, where short acoustic impulses are emitted on a part of the boundary $\Gamma_1 \subset \partial\Omega$, which are backscattered by material inhomogeneities and recorded again on Γ_1 (cf. Figure 1.1). In scanning acoustic microscopy, as in many other applications, Γ_1 is only a small part of the boundary, which introduces special difficulties for the reconstruction. Note that this problem differs significantly from another problem often considered in inverse scattering: that of reconstructing internal boundaries from far field measurements (cf. [16]).

Much research has been done to identify a coefficient in a three dimensional hyperbolic equation from boundary measurements. However, most previous papers treat the determination of the coefficient in the zeroth-order part of the hyperbolic operator—see, for example, Romanov [39] (using linearization), Imanuvilov and Yamamoto [25], Puel and Yamamoto [37], Feng et al. [22], and Rakesh [38] (requiring knowledge of the Dirichlet-to-Neumann map). The method of Carleman estimates introduced by Bukhgeim and Klibanov in [13] can prove the uniqueness and the conditional stability by a finite number of observations (cf. [26, 29] and the literature cited there). However, in the method of Carleman estimates it is necessary to assume that initial or external forces should satisfy positivity conditions, which restrict many practical applications.

Without such an assumption the problem of uniqueness and stability is still open. More generally, Bardos, Lebeau, and Rauch [2], using microlocal analysis, justified the rays of geometrical optics as tools for characterizing the stability of controllability for arbitrary domains. Their main result is the (nearly sharp) condition that every such ray must meet the controlled boundary in at least one nonglancing point.

There are different methods for computing the solution of the inverse problem for the acoustic wave equation in the time-domain—see, for example, [11, 40, 46]. Shirota [41] investigated a variational method for the reconstruction of the coefficient in an acoustic wave equation, which gave reasonable reconstruction for a set of given Cauchy data on most of the boundary. But the quality of the reconstruction deteriorated when

the measurement boundary was restricted, limiting the use for cases such as acoustic microscopy.

In order to reconstruct the parameter in a stable manner, we use an adjoint method with added Tikhonov regularization. Adjoint methods are well known in optimal control of partial differential equations [30]. They were also developed to solve inverse problems in different areas under the name of the propagation-backpropagation, time reversal, or phase conjugation method (cf. [32, 36, 42] and the literature cited there). The minimization problem is reformulated as the problem of finding a stationary point of a Lagrangian involving a forward wave equation (the state equation), a backward wave equation (the adjoint equation), and an equation expressing that the gradient with respect to the wave speed c vanishes. The optimum is sought in an iterative process solving in each step the forward and backward wave equations and updating the material coefficients by a quasi-Newton method. In Tikhonov regularization (cf. [18, 43, 45]), a small penalty term is added, stabilizing this ill-posed problem. To treat the problem of multiple local minima arising from the high frequency content of the data, we employ an adaptive approach, where we first solve the inverse problem on a coarse grid to smooth the high frequency components of the error, and then successively refine the mesh locally and use the results of previous iterations as an initial guess for a local optimization in order to capture finer details of the solution. Similar to multigrid methods (cf. [1, 14, 15, 35]), this will extend the basin of attraction of the global minimum, while at the same time improving the computational efficiency by the possibility of local refinement. Given the need for fast evaluation in applications like medical imaging and the high resolution possible with acoustic microscopy, the last point is of special importance.

The main contribution of this work therefore is to derive an a posteriori error estimate for the Lagrangian involving the residuals of the state equation, adjoint state equation, and the gradient with respect to the parameter (following [4, 5, 7, 8, 9, 20, 21, 27]) and employ this in an adaptive method where the spatial discretization is refined locally with feedback from the a posteriori error estimator. We present a numerical example showing the effectiveness of the computational inverse scattering in scanning acoustic microscopy using adaptive error control.

Additionally, often the surrounding body is homogeneous, and the material inhomogeneities occupy only a small portion of the body. In such cases the numerical solution of the wave equation is efficiently performed by a hybrid finite element/finite difference method developed in [6, 10], where the finite difference method is used in the structured part and finite elements are used in the unstructured part of the mesh. We exploit the flexibility of mesh refinement and adaptation of the finite element method in a domain including the object, and the efficiency of a structured mesh finite difference method in the surrounding homogeneous domain. The hybrid scheme can be viewed as a finite element scheme on a partially structured mesh which gives a stable coupling of the two methods.

An outline of the work is the following: in section 2 we formulate the inverse scattering problem and give the formulation of the adjoint method, in section 3 we introduce the finite element discretization, in section 4 we present a fully discrete version used in the computations, and in section 5 we describe the optimization by a quasi-Newton method. Section 6 contains the main result of this work: we prove an a posteriori error estimate and formulate an adaptive algorithm for the solution of the inverse problem. In section 7, we present computational results demonstrating the effectiveness of the adaptive finite element/difference method on an inverse scattering problem for scanning acoustic microscopy in three dimensions.

2. The inverse scattering problem. We consider the scalar wave equation in a bounded domain $\Omega \subset \mathbb{R}^3$, with boundary $\Gamma = \Gamma_1 \cup \Gamma_2$:

$$\begin{cases} \alpha \frac{\partial^2 v}{\partial t^2} - \triangle v &= 0, \quad \text{in } \Omega \times (0,T), \\ v(\cdot,0) &= 0, \quad \frac{\partial v}{\partial t}(\cdot,0) = 0, \text{ in } \Omega, \\ \partial_n v\big|_{\Gamma_1} &= v_1, \quad \text{on } \Gamma_1 \times (0,t_1], \\ \partial_n v\big|_{\Gamma_1} &= 0, \quad \text{on } \Gamma_1 \times (t_1,T], \\ \partial_n v\big|_{\Gamma_2} &= 0, \quad \text{on } \Gamma_2 \times [0,T], \end{cases} \tag{2.1}$$

where $v(x,t)$ is the scalar longitudinal displacement in an isotropic medium, t is the time variable, T is a final time, and $\alpha(x) = \frac{1}{c(x)^2}$ with $c(x) = \sqrt{(\lambda(x) + 2\mu(x))/\rho(x)}$ representing the wave speed depending on $x \in \Omega$. We assume that the density $\rho(x)$ and the Lamé coefficients $\lambda(x)$, $\mu(x)$ are strictly positive. The impulse wave $v_1(x,t)$ is initialized at the spherical boundary of the lens Γ_1 and propagated in time $(0,t_1]$ into Ω, which represents the region of investigation by the acoustic microscope (cf. 1.1).

Our goal is to find the coefficient function $\alpha(x)$ which minimizes the quantity

$$E(v,\alpha) = \frac{1}{2}\int_0^T \int_{\Gamma_1} (v - v_{obs})^2 \, dxdt + \frac{1}{2}\gamma \int_\Omega (\alpha - \alpha_0)^2 \, dx, \tag{2.2}$$

over the set $C_M = \{\alpha \in C(\overline{\Omega}) |\ M^{-1} \le \alpha(x) \le M\}$ for a fixed $M > 0$. Here v_{obs} is the observed data on Γ_1, and v (here understood as its trace on Γ) satisfies (2.1), thus depending on α. The second term involving α_0, an initial guess value for α, is a Tikhonov regularization. The choice of the regularization parameter γ is important in order to get a good reconstruction. This value will depend on the quality of the measured data, with poor data typically demanding more regularization. Several parameter choice rules exist if the noise level is known explicitly, which can be shown to be optimal (cf. [44] and literature cited there). The case of stochastic noise is discussed in [28]. Unfortunately, the noise level is hard to estimate a priori in our application, since it depends strongly on the operating conditions of the microscope and the specific sample to be measured. However, in [9], it was shown that γ can be chosen adaptively to get a best reconstruction by computing stability factors appearing in the solution to a dual linearized problem involving the Hessian of the Lagrangian. In our future research, we plan to study the behavior of the a posteriori error estimator connected to the Hessian problem for the application in scanning acoustic microscopy.

To solve this minimization problem with an adjoint method, we consider the augmented Lagrangian

$$L(u) = E(v,\alpha) + \int_0^T \int_\Omega \left(-\alpha \frac{\partial \lambda}{\partial t}\frac{\partial v}{\partial t} + \nabla\lambda\nabla v \right) \, dxdt - \int_0^{t_1} \int_{\Gamma_1} v_1 \lambda \, d\Gamma dt, \tag{2.3}$$

where $u = (v,\lambda,\alpha)$, and search for a stationary point with respect to u satisfying for all $\bar{u} = (\bar{v},\bar{\lambda},\bar{\alpha})$

$$L'(u)(\bar{u}) = 0, \tag{2.4}$$

where L' is the gradient of L. Equation (2.4) expresses that for all $\bar{u}$,

$$0 = \frac{\partial L}{\partial \lambda}(u)(\bar{\lambda}) = \int_0^T \int_\Omega \left(-\alpha \frac{\partial \bar{\lambda}}{\partial t}\frac{\partial v}{\partial t} + \nabla\bar{\lambda}\nabla v \right) \, dxdt - \int_0^{t_1} \int_{\Gamma_1} v_1 \bar{\lambda} \, d\Gamma dt, \tag{2.5}$$

(2.6)
$$0 = \frac{\partial L}{\partial v}(u)(\bar{v}) = \int_0^T \int_{\Gamma_1} (v - v_{obs})\ \bar{v}\ dxdt + \int_0^T \int_\Omega \left(-\alpha \frac{\partial \lambda}{\partial t} \frac{\partial \bar{v}}{\partial t} + \nabla \lambda \nabla \bar{v} \right)\ dxdt,$$

(2.7)
$$0 = \frac{\partial L}{\partial \alpha}(u)(\bar{\alpha}) = -\int_0^T \int_\Omega \frac{\partial \lambda(x,t)}{\partial t} \frac{\partial v(x,t)}{\partial t} \bar{\alpha}\ dxdt + \gamma \int_\Omega (\alpha - \alpha_0)\bar{\alpha}\ dx, \qquad x \in \Omega.$$

The equation (2.5) is a weak form of the state equation (2.1); (2.6) is a weak form of the adjoint state equation (which is solved backward in time)

(2.8)
$$\begin{cases} \alpha \frac{\partial^2 \lambda}{\partial t^2} - \triangle \lambda & = 0,\ x \in \Omega,\ 0 < t < T, \\ \lambda(\cdot, T) & = \frac{\partial \lambda(\cdot, T)}{\partial t} = 0, \\ \partial_n \lambda & = (v - v_{obs}) \quad \text{on } \Gamma \times [0, T], \end{cases}$$

and (2.7) expresses stationarity with respect to α. From standard results in optimal control theory (e.g., Chapter IV, Lemma 7.1 in [30]), we know that the solution of (2.8) exists and is unique.

The existence of minimizers of (2.2) can be proven using the techniques in [22]. Assuming compatibility conditions for v_1, one can prove a priori estimates in $H^1(\Omega \times [0,T])$ for the gradient of v (cf. [24,31]). Taking a minimizing sequence α_n of (2.2) in C_M with $\lim_{n\to\infty} E(v(\alpha_n), \alpha_n) = \inf_{\alpha \in C_M} E(v(\alpha), \alpha)$ and denoting $v_n = v(\alpha_n)$, we have from the strong convergence of ∇v_n in $L^2(\Omega \times [0,T])$ and the boundedness of α_n^{-1} the convergence of $\alpha_n^{-1} \nabla v_n$ in $L^2(\Omega \times [0,T])$. Passing to the limit in the weak formulation of (2.1) guarantees the existence of the limit v, and hence we conclude from the lower semicontinuity of the L^2 norm with respect to weak convergence that the limit (v, α) is a minimizer of (2.2).

To compute the solution of the minimization problem, we will use a quasi-Newton method with limited storage, which is described in section 5.

3. Finite element discretization. We now formulate a finite element method for (2.4) based on using continuous piecewise linear functions in space and time. We discretize $\Omega \times (0,T)$ in the usual way, denoting by $K_h = \{K\}$ a partition of the domain Ω into tetrahedra K ($h = h(x)$ being a mesh function representing the local diameter of the elements), and we let $J_k = \{J = J_1 \cup J_2\}$ be a partition of the time interval $(0,T)$ into time intervals $J = (t_{k-1}, t_k]$ of uniform length $\tau = t_k - t_{k-1}$, where $J_1 = (t_{k-1}, t_k],\ t \in (0, t_1]$, and $J_2 = (t_{k-1}, t_k],\ t \in (t_1, T]$. In fully discrete form, the resulting method corresponds to a centered finite difference approximation for the second-order time derivative and a usual finite element approximation of the Laplacian.

To formulate the finite element method for (2.4) we introduce the finite element spaces V_h, W_h^v, and W_h^λ defined by

$$\begin{aligned}
V_h &:= \{v \in L_2(\Omega) : v \in P_0(K), \forall K \in K_h\}, \\
W_1^v &:= \{v \in H^1(\Omega \times J_1) : v(\cdot, 0) = 0,\ \partial_n v|_{\Gamma_1} = v_1,\ \partial_n v|_{\Gamma_2} = 0\}, \\
W_2^v &:= \{v \in H^1(\Omega \times J_2) : v(\cdot, 0) = 0,\ \partial_n v|_{\Gamma} = 0\}, \\
W^\lambda &:= \{\lambda \in H^1(\Omega \times J) : \lambda(\cdot, T) = 0,\ \partial_n \lambda|_{\Gamma} = 0\}, \\
W_h^v &:= \{v \in W_1^v \cup W_2^v : v|_{K \times (J_1 \cup J_2)} \in P_1(K) \times (P_1(J_1) \cup P_1(J_2)), \forall K \in K_h, \forall J \in J_k\}, \\
W_h^\lambda &:= \{v \in W^\lambda : v|_{K \times J} \in P_1(K) \times P_1(J), \forall K \in K_h, \forall J \in J_k\},
\end{aligned}$$

where $P_1(K)$ and $P_1(J)$ are the sets of linear functions on K and J, respectively.

We define $U_h = W_h^v \times W_h^\lambda \times V_h$. The finite element method now reads: Find $u_h \in U_h$ such that

$$L'(u_h)(\bar{u}) = 0 \quad \forall \bar{u} \in U_h. \tag{3.1}$$

4. Fully discrete scheme. Expanding v, λ in terms of the standard continuous piecewise linear functions $\varphi_i(x)$ in space and $\psi_i(t)$ in time and substituting this into (2.5)–(2.6), we obtain the following system of linear equations:

$$M(\mathbf{v}^{k+1} - 2\mathbf{v}^k + \mathbf{v}^{k-1}) = \tau^2 P_1^k - \tau^2 K \left(\frac{1}{6}\mathbf{v}^{k-1} + \frac{2}{3}\mathbf{v}^k + \frac{1}{6}\mathbf{v}^{k+1} \right), \tag{4.1}$$

$$M(\boldsymbol{\lambda}^{k+1} - 2\boldsymbol{\lambda}^k + \boldsymbol{\lambda}^{k-1}) = -\tau^2 S^k - \tau^2 K \left(\frac{1}{6}\boldsymbol{\lambda}^{k-1} + \frac{2}{3}\boldsymbol{\lambda}^k + \frac{1}{6}\boldsymbol{\lambda}^{k+1} \right), \tag{4.2}$$

with initial conditions for both v and λ:

$$v(0) = \frac{\partial v}{\partial t}(0) = 0, \tag{4.3}$$

$$\lambda(T) = \frac{\partial \lambda}{\partial t}(T) = 0. \tag{4.4}$$

Here, M is the mass matrix in space, K is the stiffness matrix, $k = 1, 2, 3 \ldots$ denotes the time level, F^k, S^k are the load vectors, $\mathbf{v}$ is the unknown discrete field values of v, $\boldsymbol{\lambda}$ is the unknown discrete field values of λ, and τ is the time step.

The explicit formulas for the entries in system (4.1)–(4.2) at each element e can be given as

$$M_{i,j}^e = (\alpha\varphi_i, \varphi_j)_e, \tag{4.5}$$

$$K_{i,j}^e = (\bigtriangledown\varphi_i, \bigtriangledown\varphi_j)_e, \tag{4.6}$$

$$P_{1_{j,m}} = (v_1, \varphi_j\psi_m)_{\Gamma_1 \times J_1}, \tag{4.7}$$

$$S_{j,m}^e = (v - \tilde{v}, \varphi_j\psi_m)_{e \times J}, \tag{4.8}$$

where $(.,.)_e$ denotes the $L_2(e)$ scalar product. The matrix M_e is the contribution from element e to the global assembled matrix in space M, K^e is the contribution from element e to the global assembled matrix K, P_1 is the contribution of the boundary term on Γ_1, and S^e is the contribution from element e to the assembled source vector of the right-hand side of (2.8).

To obtain an explicit scheme we approximate M with the lumped mass matrix M^L, the diagonal approximation obtained by taking the row sum of M; see, e.g., [23]. By multiplying (4.1)–(4.2) with $(M^L)^{-1}$ and replacing the terms $\frac{1}{6}\mathbf{v}^{k-1} + \frac{2}{3}\mathbf{v}^k + \frac{1}{6}\mathbf{v}^{k+1}$ and $\frac{1}{6}\boldsymbol{\lambda}^{k-1} + \frac{2}{3}\boldsymbol{\lambda}^k + \frac{1}{6}\boldsymbol{\lambda}^{k+1}$ by $\mathbf{v}^k$ and $\boldsymbol{\lambda}^k$, respectively, we obtain an efficient explicit formulation:

$$\mathbf{v}^{k+1} = \tau^2 (M^L)^{-1} P_1^k + 2\mathbf{v}^k - \tau^2 (M^L)^{-1} K\mathbf{v}^k - \mathbf{v}^{k-1}, \tag{4.9}$$

$$\boldsymbol{\lambda}^{k-1} = -\tau^2 (M^L)^{-1} S^k + 2\boldsymbol{\lambda}^k - \tau^2 (M^L)^{-1} K\boldsymbol{\lambda}^k - \boldsymbol{\lambda}^{k+1}. \tag{4.10}$$

The discrete version of (2.7) takes the form

$$0 = -\int_0^T \int_\Omega \frac{\partial \lambda_h}{\partial t} \frac{\partial v_h}{\partial t} \bar{\alpha}\, dxdt + \gamma \int_\Omega (\alpha_h - \alpha_0)\bar{\alpha}\, dx, \qquad \forall \bar{\alpha} \in V_h. \tag{4.11}$$

5. Optimization by a quasi-Newton method. To solve the discrete problem (3.1), we use a quasi-Newton method with limited storage [33], where we compute a sequence α_h^k, $k = 0, 1, \dots$, of approximations of α_h with nodal values α_k given by

$$\alpha_{k+1} = \alpha_k - \rho^k H^k g_k. \tag{5.1}$$

Here, the step length ρ^n is computed with a line-search algorithm given in [34], and g_k are the nodal values of the gradient given by

$$g^k(x) = -\int_0^T \frac{\partial \lambda_h^k(x,t)}{\partial t} \frac{\partial v_h^k(x,t)}{\partial t} dt + \gamma(\alpha_k(x) - \alpha_0), \tag{5.2}$$

where v_h^k and λ_h^k solve the discrete analogues of (2.1) and (2.8).

H^k is given by the usual BFGS update formula of the Hessian (cf. [17])

$$H^{k+1} = (I - ds_k y_k^T) H^k (I - dy_k s_k^T) + \rho s_k s_k^T, \tag{5.3}$$

where $d = 1/(y_k^T s_k)$ and

$$s_k = \alpha_{k+1} - \alpha_k, \tag{5.4}$$

$$y_k = g_{k+1} - g_k. \tag{5.5}$$

Note that instead of explicitly computing the Hessian H^k in (5.1), we compute the product $H^{k+1} g^k$ from (5.3) to get

$$((I - ds_k y_k^T) H^k (I - dy_k s_k^T) + \rho s_k s_k^T) g^k = (I - ds_k y_k^T) H^k (g^k - dy_k s_k^T g^k) + \rho s_k s_k^T g^k, \tag{5.6}$$

involving only scalar products of vectors and computing $H^k g^k$ similarly. A modified version of H_k is stored implicitly, by using a certain number m of the vector pairs (s_i, y_i).

6. An a posteriori error estimate for the Lagrangian and an adaptive algorithm.

6.1. A posteriori error estimate. Following [8], we now present the main steps in the proof of an a posteriori error estimate for the Lagrangian. Let C denote various constants of moderate size. We start by writing an equation for the error e in the Lagrangian as

$$\begin{aligned} e = L(u) - L(u_h) &= \int_0^1 \frac{d}{d\epsilon} L(u\epsilon + (1-\epsilon)u_h) d\epsilon \\ &= \int_0^1 L'(u\epsilon + (1-\epsilon)u_h)(u - u_h) d\epsilon = L'(u_h)(u - u_h) + R, \end{aligned} \tag{6.1}$$

where R denotes (a small) second-order term. For full details of the arguments we refer the reader to [3] and [21].

Using the Galerkin orthogonality (3.1), the splitting $u - u_h = (u - u_h^I) + (u_h^I - u_h)$, where u_h^I denotes an interpolant of u, and neglecting the term R, we get the following error representation:

$$e \approx L'(u_h)(u - u_h^I) = (I_1 + I_2 + I_3), \tag{6.2}$$

where

$$I_1 = \int_0^T \int_\Omega \left(-\alpha_h \frac{\partial(\lambda - \lambda_h^I)}{\partial t} \frac{\partial v_h}{\partial t} + \nabla(\lambda - \lambda_h^I) \nabla v_h \right) \, dxdt - \int_0^{t_1} \int_{\Gamma_1} v_1 (\lambda - \lambda_h^I) \, d\Gamma dt, \tag{6.3}$$

$$I_2 = \int_0^T \int_\Omega (v_h - \tilde{v})(v - v_h^I) \, \delta_{obs} \, dxdt + \int_0^T \int_\Omega \left(-\alpha_h \frac{\partial \lambda_h}{\partial t} \frac{\partial (v - v_h^I)}{\partial t} + \nabla \lambda_h \nabla (v - v_h^I) \right) \, dxdt, \tag{6.4}$$

$$I_3 = -\int_0^T \int_\Omega \frac{\partial \lambda_h(x,t)}{\partial t} \frac{\partial v_h(x,t)}{\partial t} (\alpha - \alpha_h^I) \, dxdt + \gamma \int_\Omega (\alpha_h - \alpha_0)(\alpha - \alpha_h^I) dx. \tag{6.5}$$

To estimate (6.3) we integrate by parts in the first and second terms to get

$$\begin{aligned} \left| I_1 \right| = \Bigg| & \int_0^T \int_\Omega \left(\alpha_h \frac{\partial^2 v_h}{\partial t^2} (\lambda - \lambda_h^I) - \triangle v_h (\lambda - \lambda_h^I) \right) \, dxdt \\ & - \int_0^{t_1} \int_{\Gamma_1} v_1 (\lambda - \lambda_h^I) \, d\Gamma dt + \sum_K \int_0^T \int_{\partial K} \frac{\partial v_h}{\partial n_K} (\lambda - \lambda_h^I) \, dsdt \\ & - \sum_k \int_\Omega \alpha_h \left[\frac{\partial v_h}{\partial t}(t_k) \right] (\lambda - \lambda_h^I)(t_k) \, dx \Bigg|, \end{aligned} \tag{6.6}$$

where the terms $\frac{\partial v_h}{\partial n_K}$ and $\left[\frac{\partial v_h}{\partial t}\right]$ appear during the integration by parts and denote the derivative of v_h in the outward normal direction n_K of the boundary ∂K of element K and the jump of the derivative of v_h in time, respectively. In the third term of (6.6) we sum over the element boundaries, and each internal side $S \in S_h$ occurs twice. Denoting by $\partial_s v_h$ the derivative of a function v_h in one of the normal directions of each side S, we can write

$$\sum_K \int_{\partial K} \frac{\partial v_h}{\partial n_K} (\lambda - \lambda_h^I) \, ds = \sum_S \int_S \left[\partial_s v_h \right] (\lambda - \lambda_h^I) \, ds, \tag{6.7}$$

where $\left[\partial_s v_h\right]$ is the jump in the derivative $\partial_s v_h$ computed from the two elements sharing S. We distribute each jump equally to the two sharing triangles and return to a sum over element edges ∂K:

$$\sum_S \int_S \left[\partial_s v_h \right] (\lambda - \lambda_h^I) \, ds = \sum_K \frac{1}{2} h_K^{-1} \int_{\partial K} \left[\partial_s v_h \right] (\lambda - \lambda_h^I) \, h_K \, ds. \tag{6.8}$$

We formally set $dx = h_K ds$ and replace the integrals over the element boundaries ∂K by integrals over the elements K, to get

$$\left| \sum_K \frac{1}{2} h_K^{-1} \int_{\partial K} \left[\partial_s v_h \right] (\lambda - \lambda_h^I) \, h_K \, ds \right| \le C \max_{S \subset \partial K} h_K^{-1} \int_\Omega \left| \left[\partial_s v_h \right] \right| \cdot \left| (\lambda - \lambda_h^I) \right| \, dx, \tag{6.9}$$

where $\left[\partial_s v_h\right]\big|_K = \max_{S\subset\partial K}\left[\partial_s v_h\right]\big|_S$.

In a similar way we can estimate the fourth term in (6.6):

$$\begin{aligned}\left|\sum_k \int_\Omega \alpha_h \left[\frac{\partial v_h}{\partial t}(t_k)\right](\lambda-\lambda_h^I)(t_k)\,dx\right| &\le \sum_k \int_\Omega \alpha_h\tau^{-1}\cdot\left|\left[\frac{\partial v_h}{\partial t}(t_k)\right]\right|\cdot\left|(\lambda-\lambda_h^I)(t_k)\right|\,\tau dx\\ &\le C\sum_k\int_{J_k}\int_\Omega \alpha_h\tau^{-1}\cdot\left|\left[\partial v_{ht_k}\right]\right|\cdot\left|(\lambda-\lambda_h^I)\right|\,dxdt\\ &= C\int_0^T\int_\Omega \alpha_h\tau^{-1}\cdot\left|\left[\partial v_{ht}\right]\right|\cdot\left|(\lambda-\lambda_h^I)\right|\,dxdt,\end{aligned}\tag{6.10}$$

where

$$[\partial v_{ht_k}] = \max_{J_k}\left(\left[\frac{\partial v_h}{\partial t}(t_k)\right],\ \left[\frac{\partial v_h}{\partial t}(t_{k+1})\right]\right),\tag{6.11}$$

and $[\partial v_{ht}]$ is defined as the maximum of the two jumps in time on each time interval J_k appearing in (6.11):

$$[\partial v_{ht}] = [\partial v_{ht_k}] \quad \text{on } J_k.$$

Substituting both of the above expressions for the second and third terms in (6.6), we get

$$\begin{aligned}\left|I_1\right| \le &\left|\int_0^T\int_\Omega\left(\alpha_h\frac{\partial^2 v_h}{\partial t^2}-\triangle v_h\right)(\lambda-\lambda_h^I)\,dxdt\right|\\ &-\left|\int_0^{t_1}\int_{\Gamma_1} v_1(\lambda-\lambda_h^I)\,d\Gamma dt\right|\\ &+C\int_0^T\int_\Omega \max_{S\subset\partial K} h_k^{-1}\cdot\left|\left[\partial_s v_h\right]\right|\cdot\left|(\lambda-\lambda_h^I)\right|\,dxdt\\ &+C\int_0^T\int_\Omega \alpha_h\tau^{-1}\cdot\left|\left[\partial v_{ht}\right]\right|\cdot\left|(\lambda-\lambda_h^I)\right|\,dxdt.\end{aligned}\tag{6.12}$$

Next, we use a standard interpolation estimate for $\lambda-\lambda_h^I$ to get

$$\begin{aligned}\left|I_1\right| \le\ & C\int_0^T\int_\Omega\left|\alpha_h\frac{\partial^2 v_h}{\partial t^2}-\triangle v_h\right|\cdot\left(\tau^2\left|\frac{\partial^2\lambda}{\partial t^2}\right|+h^2\left|D_x^2\lambda\right|\right)\,dxdt\\ &-C\int_0^{t_1}\int_{\Gamma_1} v_1\cdot\left(\tau^2\left|\frac{\partial^2\lambda}{\partial t^2}\right|+h^2\left|D_x^2\lambda\right|\right)\,d\Gamma dt\\ &+C\int_0^T\int_\Omega \max_{S\subset\partial K} h_k^{-1}\cdot\left|\left[\partial_s v_h\right]\right|\cdot\left(\tau^2\left|\frac{\partial^2\lambda}{\partial t^2}\right|+h^2\left|D_x^2\lambda\right|\right)\,dxdt\\ &+C\int_0^T\int_\Omega \alpha_h\tau^{-1}\cdot\left|\left[\partial v_{ht}\right]\right|\cdot\left(\tau^2\left|\frac{\partial^2\lambda}{\partial t^2}\right|+h^2\left|D_x^2\lambda\right|\right)\,dxdt.\end{aligned}\tag{6.13}$$

Next, we note that the first integral in (6.13) disappears, since v_h is a continuous piecewise linear function. We then estimate $\frac{\partial^2\lambda}{\partial t^2} \approx \frac{[\partial\lambda_h/\partial t]}{\tau}$ and $D_x^2\lambda \approx \frac{[\partial\lambda_h/\partial n]}{h}$ to get

$$
\begin{aligned}
\left|I_1\right| &\le C\int_0^{t_1}\int_{\Gamma_1}\left|v_1\right|\cdot\left(\tau^2\left|\frac{\left[\frac{\partial\lambda_h}{\partial t}\right]}{\tau}\right|+h^2\left|\frac{\left[\frac{\partial\lambda_h}{\partial n}\right]}{h}\right|\right)\,d\Gamma dt\\
&+ C\int_0^T\int_\Omega \max_{S\subset\partial K} h_k^{-1}\left|\left[\partial_s v_h\right]\right|\cdot\left(\tau^2\left|\frac{\left[\frac{\partial\lambda_h}{\partial t}\right]}{\tau}\right|+h^2\left|\frac{\left[\frac{\partial\lambda_h}{\partial n}\right]}{h}\right|\right)\,dxdt \qquad (6.14)\\
&+ C\int_0^T\int_\Omega \alpha_h\tau^{-1}\left|\left[\partial v_{ht}\right]\right|\cdot\left(\tau^2\left|\frac{\left[\frac{\partial\lambda_h}{\partial t}\right]}{\tau}\right|+h^2\left|\frac{\left[\frac{\partial\lambda_h}{\partial n}\right]}{h}\right|\right)\,dxdt.
\end{aligned}
$$

We estimate I_2 similarly:

$$
\begin{aligned}
\left|I_2\right| &\le \int_0^T\int_\Omega\left|\left(\alpha_h\frac{\partial^2\lambda_h}{\partial t^2}(v-v_h^I)-\triangle\lambda_h(v-v_h^I)-(v_h-\tilde v)(v-v_h^I)\right)\right|\,dxdt\\
&+ C\int_0^T\int_\Omega \max_{S\subset\partial K} h_k^{-1}\cdot\left|\left[\partial_s\lambda_h\right]\right|\cdot\left|(v-v_h^I)\right|\,dxdt\\
&+ C\int_0^T\int_\Omega \alpha_h\tau^{-1}\cdot\left|\left[\partial\lambda_{ht}\right]\right|\cdot\left|(v-v_h^I)\right|\,dxdt\\
&\le C\int_0^T\int_\Omega\left|\left(\alpha_h\frac{\partial^2\lambda_h}{\partial t^2}-\triangle\lambda_h-(v_h-\tilde v)\right)\right|\cdot\left|(v-v_h^I)\right|\,dxdt\\
&+ C\int_0^T\int_\Omega \max_{S\subset\partial K} h_k^{-1}\cdot\left|\left[\partial_s\lambda_h\right]\right|\cdot\left|(v-v_h^I)\right|\,dxdt\\
&+ C\int_0^T\int_\Omega \alpha_h\tau^{-1}\cdot\left|\left[\partial\lambda_{ht}\right]\right|\cdot\left|(v-v_h^I)\right|\,dxdt\\
&\le C\int_0^T\int_\Omega\left|\left(\alpha_h\frac{\partial^2\lambda_h}{\partial t^2}-\triangle\lambda_h-(v_h-\tilde v)\right)\right|\left(\tau^2\left|\frac{\partial^2 v}{\partial t^2}\right|+h^2\left|D_x^2 v\right|\right)\,dxdt \qquad (6.15)\\
&+ C\int_0^T\int_\Omega \max_{S\subset\partial K} h_k^{-1}\cdot\left|\left[\partial_s\lambda_h\right]\right|\left(\tau^2\left|\frac{\partial^2 v}{\partial t^2}\right|+h^2\left|D_x^2 v\right|\right)\,dxdt\\
&+ C\int_0^T\int_\Omega \alpha_h\tau^{-1}\cdot\left|\left[\partial\lambda_{ht}\right]\right|\cdot\left(\tau^2\left|\frac{\partial^2 v}{\partial t^2}\right|+h^2\left|D_x^2 v\right|\right)\,dxdt\\
&\le C\int_0^T\int_\Omega\left|(v_h-\tilde v)\right|\cdot\left(\tau^2\left|\frac{\left[\frac{\partial v_h}{\partial t}\right]}{\tau}\right|+h^2\left|\frac{\left[\frac{\partial v_h}{\partial n}\right]}{h}\right|\right)\,dxdt\\
&+ C\int_0^T\int_\Omega \max_{S\subset\partial K} h_k^{-1}\left|\left[\partial_s\lambda_h\right]\right|\cdot\left(\tau^2\left|\frac{\left[\frac{\partial v_h}{\partial t}\right]}{\tau}\right|+h^2\left|\frac{\left[\frac{\partial v_h}{\partial n}\right]}{h}\right|\right)\,dxdt\\
&+ C\int_0^T\int_\Omega \alpha_h\tau^{-1}\cdot\left|\left[\partial\lambda_{ht}\right]\right|\cdot\left(\tau^2\left|\frac{\left[\frac{\partial v_h}{\partial t}\right]}{\tau}\right|+h^2\left|\frac{\left[\frac{\partial v_h}{\partial n}\right]}{h}\right|\right)\,dxdt.
\end{aligned}
$$

To estimate I_3 we use a standard approximation estimate of the form $\alpha - \alpha_h^I \approx hD_x\alpha$ to get

$$
\begin{aligned}
\big|I_3\big| &\le \int_0^T \int_\Omega \left|\frac{\partial \lambda_h(x,t)}{\partial t} \cdot \frac{\partial v_h(x,t)}{\partial t}\right| \cdot h \cdot \big|D_x\alpha\big| \, dxdt \\
&\quad + \gamma \int_\Omega (\alpha_h - \alpha_0) h \cdot \big|D_x\alpha\big| \, dx \\
&\le C \int_0^T \int_\Omega \left|\frac{\partial \lambda_h(x,t)}{\partial t} \cdot \frac{\partial v_h(x,t)}{\partial t}\right| \cdot h \cdot \left|\frac{[\alpha_h]}{h}\right| \, dxdt \\
&\quad + C\gamma \int_\Omega |\alpha_h - \alpha_0| \cdot h \cdot \left|\frac{[\alpha_h]}{h}\right| \, dx \\
&\le C \int_0^T \int_\Omega \left|\frac{\partial \lambda_h(x,t)}{\partial t} \cdot \frac{\partial v_h(x,t)}{\partial t}\right| \cdot \big|[\alpha_h]\big| \, dxdt \\
&\quad + C\gamma \int_\Omega |\alpha_h - \alpha_0| \cdot \big|[\alpha_h]\big| \, dx.
\end{aligned}
\tag{6.16}
$$

We therefore obtain the following result.

THEOREM 6.1. [illegible] $L(u) = L(v, \lambda, \alpha)$ [illegible] (2.3) [illegible] $L(u_h) = L(v_h, \lambda_h, \alpha_h)$ [illegible] $L(u)$ [illegible] e [illegible]

$$
\begin{aligned}
|e| \le \Bigg(&\int_0^{t_1} \int_{\Gamma_1} R_{v_1}\sigma_\lambda \, d\Gamma dt + \int_0^T \int_\Omega R_{v_2}\sigma_\lambda \, dxdt + \int_0^T \int_\Omega R_{v_3}\sigma_\lambda \, dxdt \\
&+ \int_0^T \int_\Omega R_{\lambda_1}\sigma_v \, dxdt + \int_0^T \int_\Omega R_{\lambda_2}\sigma_v \, dxdt + \int_0^T \int_\Omega R_{\lambda_3}\sigma_v \, dxdt \\
&+ \int_0^T \int_\Omega R_{\alpha_1}\sigma_\alpha \, dxdt + \int_\Omega R_{\alpha_2}\sigma_\alpha \, dx \Bigg),
\end{aligned}
\tag{6.17}
$$

[illegible]

$$
R_{v_1} = \big|v_1\big|, \quad R_{v_2} = \max_{S\subset\partial K} h_k^{-1}\big|\big[\partial_s v_h\big]\big|, \quad R_{v_3} = \alpha_h \tau^{-1}\big|\big[\partial v_{ht}\big]\big|, \tag{6.18}
$$

$$
R_{\lambda_1} = \big|v_h - \tilde{v}\big|, \quad R_{\lambda_2} = \max_{S\subset\partial K} h_k^{-1}\big|\big[\partial_s \lambda_h\big]\big|, \quad R_{\lambda_3} = \alpha_h \tau^{-1}\big|\big[\partial \lambda_{ht}\big]\big|, \tag{6.19}
$$

$$
R_{\alpha_1} = \left|\frac{\partial \lambda_h}{\partial t}\right| \cdot \left|\frac{\partial v_h}{\partial t}\right|, \quad R_{\alpha_2} = \gamma|\alpha_h - \alpha_0|, \tag{6.20}
$$

[illegible]

$$
\sigma_\lambda = C\tau \left|\left[\frac{\partial \lambda_h}{\partial t}\right]\right| + Ch\left|\left[\frac{\partial \lambda_h}{\partial n}\right]\right|, \tag{6.21}
$$

$$
\sigma_v = C\tau \left|\left[\frac{\partial v_h}{\partial t}\right]\right| + Ch\left|\left[\frac{\partial v_h}{\partial n}\right]\right|, \tag{6.22}
$$

$$
\sigma_\alpha = C\big|[\alpha_h]\big|. \tag{6.23}
$$

Note that R_{α_2} corresponds to the penalty term added in the Tikhonov regularization of (2.2). Indeed, as γ tends to zero, the error term R_{α_1} will dominate, and in the limit we arrive at the error estimator for the Lagrangian without regularization.

6.2. Adaptive algorithm. The main goal in adaptive error control for the Lagrangian is to find a mesh K_h with as few nodes as possible, such that $\|L(u) - L(u_h)\| < tol$. Instead of finding $L(u)$ analytically, we will use the a posteriori error estimate: We shall find a partition K_h such that the corresponding finite element approximation $L(u_h)$ satisfies

$$\begin{aligned}\Bigg(&\int_0^{t_1}\int_{\Gamma_1} R_{v_1}\sigma_\lambda \, d\Gamma dt + \int_0^T\int_\Omega R_{v_2}\sigma_\lambda \, dxdt + \int_0^T\int_\Omega R_{v_3}\sigma_\lambda \, dxdt \\ &+ \int_0^T\int_\Omega R_{\lambda_1}\sigma_v \, dxdt + \int_0^T\int_\Omega R_{\lambda_2}\sigma_v \, dxdt + \int_0^T\int_\Omega R_{\lambda_3}\sigma_v \, dxdt \\ &+ \int_0^T\int_\Omega R_{\alpha_1}\sigma_\alpha \, dxdt + \int_\Omega R_{\alpha_2}\sigma_\alpha \, dx\Bigg) < tol.\end{aligned} \tag{6.24}$$

The solution is found by an iterative process, where we start with a coarse mesh and successively refine the mesh by using the stopping criterion (6.24) with a view of using the minimal possible number of elements. More precisely, in the computations below we shall use the following adaptive algorithm:

1. Choose an initial mesh K_h and an initial time partition J_k of the time interval $(0, T)$.
2. Compute the solution v of the forward problem (2.1) on K_h and J_k with $\alpha = \alpha^{(n)}$.
3. Compute the solution λ of the adjoint problem (2.8) on K_h and J_k.
4. Update the velocity on K_h and J_k according to

$$\alpha^{(n+1)}(x) = \alpha^{(n)}(x) + \rho^{(n)}\left(\int_0^T \frac{\partial\lambda^n(x,t)}{\partial t}\frac{\partial v^n(x,t)}{\partial t}dt + \gamma(\alpha^{(n)}(x) - \alpha_0)\right).$$

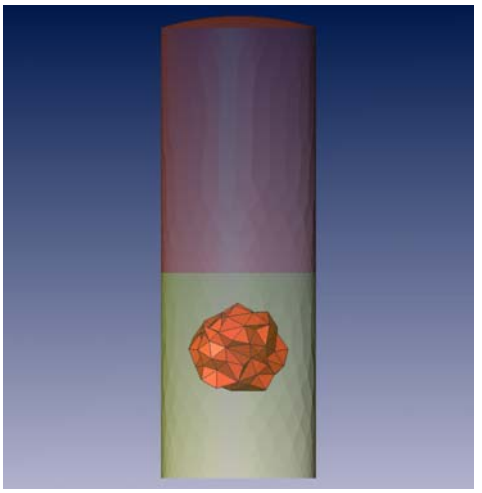
   Repeat steps 1–4 as long as the gradient quickly decreases.
5. Refine all elements where $(R_{\alpha_1} + R_{\alpha_2})\sigma_\alpha > tol$, and construct a new mesh K_h and a new time partition J_k. Here *tol* is a tolerance chosen by the user. Return to 1.

7. Numerical examples. We illustrate the efficiency and the performance of the hybrid FEM/FDM method on an inverse scattering problem for scanning acoustic microscopy in three dimensions. The geometry of the problem (see Figure 1.1) is taken from a specific microscope (WinSAM 2000, KSI Germany). The computational domain is set as $\Omega = [-10.0, 10.0] \times [-14.0, 16.0] \times [-10.0, 10.0]$, which is split into a finite element domain $\Omega_{FEM} = [-9.0, 9.0] \times [-10.0, -12.0] \times [-9.0, 9.0]$, with an unstructured mesh, and a surrounding domain Ω_{FDM}, with a structured mesh. The space mesh in Ω_{FEM} consists of tetrahedra and in Ω_{FDM} of hexahedra with mesh size $h = 1.0$. We apply the hybrid finite element/difference method presented in [10] with finite elements in Ω_{FEM} and finite differences in Ω_{FDM}. At all boundaries of Ω we use first-order absorbing boundary conditions [19].

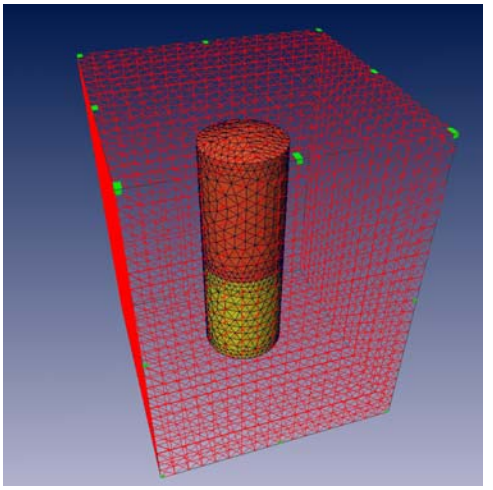
The forced acoustic field consists of a wave $v = (v_1, v_2, v_3)$ given as

$$v_i(x, y, z, t)|_{y=0} = -\left(\frac{\sin(100t - \pi/2) + 1}{10}\right) \cdot n_i, \qquad 0 \le t \le \frac{2\pi}{100}, \tag{7.1}$$

which initiates at the spherical boundary Γ_1 of the lens in Ω_{FEM} and propagates in normal direction $n = (n_1, n_2, n_3)$ into Ω. This acoustic field is a simple model of the



(A) *Geometry of the microscope with inclusion to be reconstructed.*



(B) *Surrounding mesh (outlined) with overlapping nodes at the boundary.*

FIG. 7.1. *Original mesh for computational domain* Ω_{FEM}.

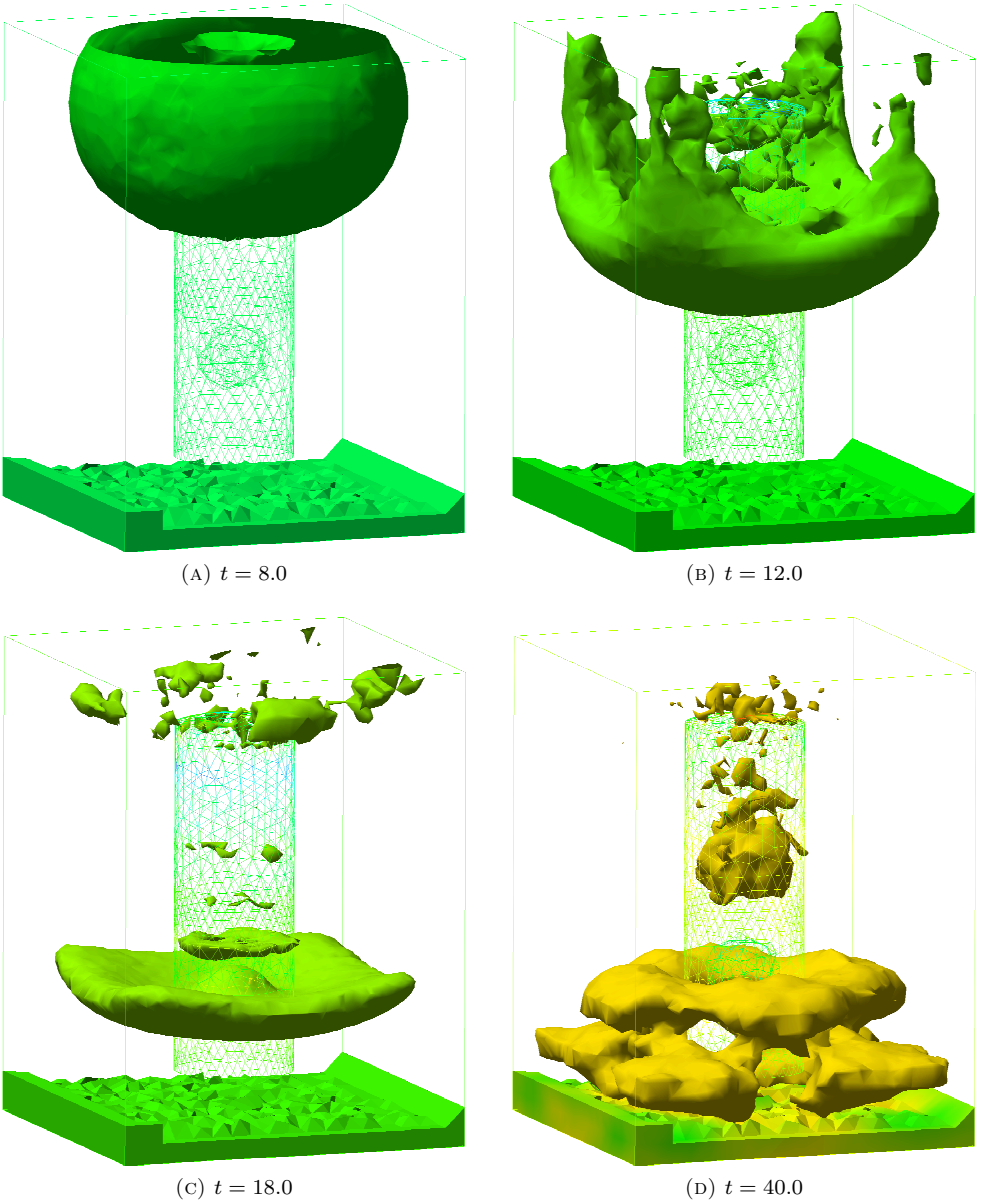
high-frequency excitation pulse generated by the transducer of the microscope. For real data, of course, it is advisable to take the source wavelet used in the specific microscope (which is known). As mentioned in the introduction, due to the multilevel strategy of our adaptive algorithm, we expect convergence for such multimodal sources as well. The observation points are also placed on Γ_1, corresponding to our application, where the same transducer records the reflected waves.

In all the computational tests we chose a time step τ according to the Courant–Friedrichs–Levy (CFL) stability condition

$$\tau \leq \frac{h}{ac_{\max}}, \tag{7.2}$$

where h is the minimal local mesh size, $c_{\max}$ is an a priori upper bound for the coefficient c, and a is a constant.

To improve the reconstruction and achieve better convergence we use the adaptive algorithm described in section 6.2. As we see from Theorem 6.1, the error in the Lagrangian consists of space-time integrals of different residuals multiplied by the interpolation errors. Thus, to estimate the error in the Lagrangian, we need to compute approximated values of $(v_h, \lambda_h, \alpha_h)$ together with residuals and interpolation errors. Since the residuals $R_{\alpha_1}, R_{\alpha_2}$ dominate, we neglect the terms $R_{v_2}, R_{v_3}, R_{\lambda_2}, R_{\lambda_3}$ in computation of the a posteriori error estimator. Further, we also neglect computation of the residuals R_{v_1}, R_{λ_1} since they indicate the error in the upper cylinder of the acoustic microscope, where we already know the value of $c(x) = c_0$. In the current work, the local refinement is based on the residuals since they already give good indications of where to adapt the mesh. The interpolation errors can be obtained following [9] by computing the Hessian of the Lagrangian, and will be included as part of a future work. Thus, the solution of the optimization problem is found in an iterative process, where we start with a coarse mesh shown in Figure 7.1 and evaluate the residuals by computing the Jacobian of the Lagrangian when the L^2 norm of $v - v_{obs}$ stops decreasing in the quasi-Newton iteration. Then we refine this mesh locally where the residuals are largest and construct a new time partition using



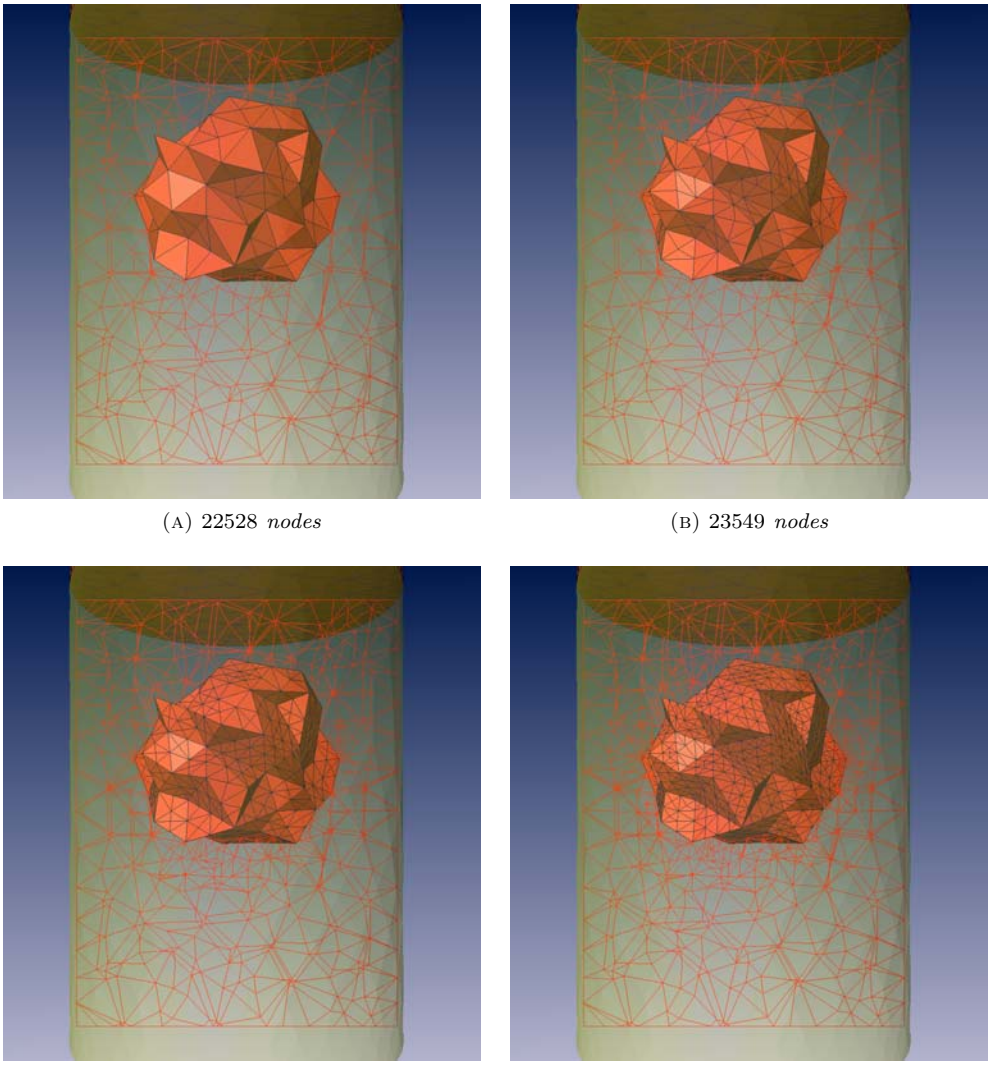
(A) $t = 8.0$ (B) $t = 12.0$

(C) $t = 18.0$ (D) $t = 40.0$

FIG. 7.2. *Solution of the forward problem* (4.9) *with exact value of the parameter* $c = 0.5$ *inside a spherical inclusion and* $c = 1.0$ *everywhere else in* Ω. *We show isosurfaces of the computed solution at different times inside* Ω_{FEM}.

(7.2). In all optimization steps, the number of stored corrections in the quasi-Newton method is $m = 5.0$.

To generate the data at the observation points, we solve the forward problem in the time interval $t = [0, 40.0]$ with the exact value of the parameter $c = 0.5$ inside a spherical inclusion and $c = 1.0$ everywhere else in Ω. In Figure 7.2 we present isosurfaces of the acoustic wavefield (i.e., the solution of the forward problem (4.9) with exact parameters) at different times inside Ω_{FEM}. The deformation of the wave



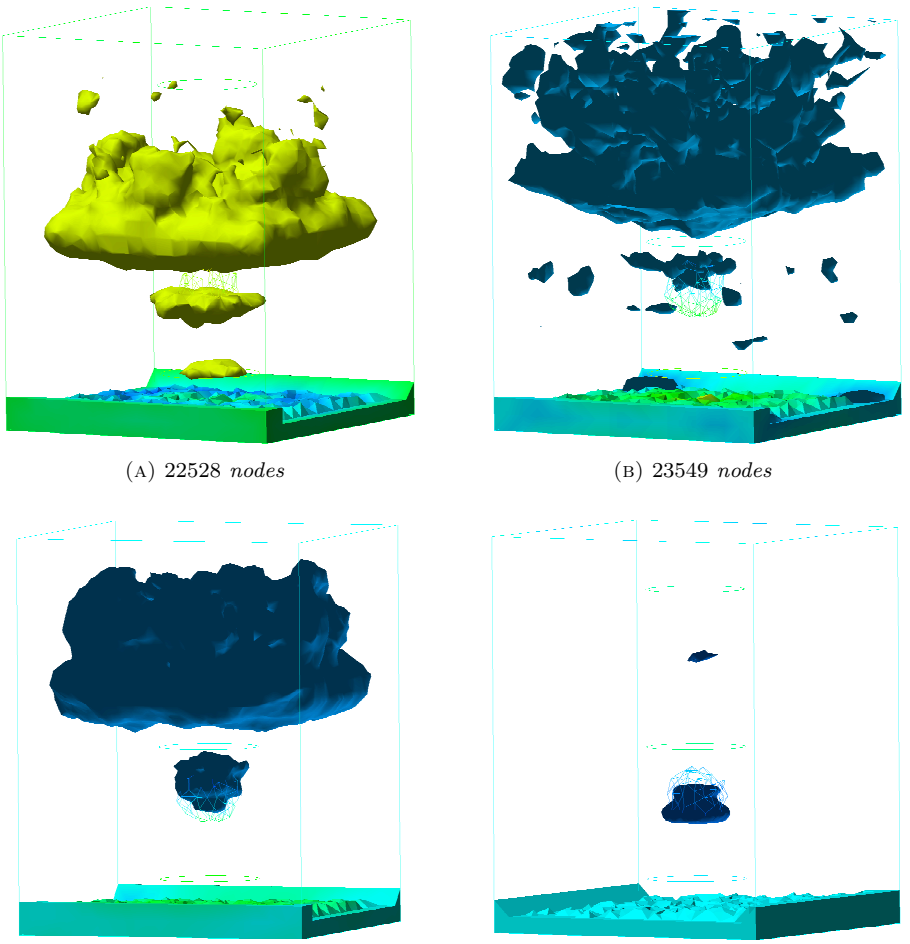
(A) 22528 *nodes* (B) 23549 *nodes*

(C) 26133 *nodes* (D) 33138 *nodes*

FIG. 7.3. *Successive adaptive refinements of the original mesh.*

packet due to the presence of the inclusion leads to reflections traveling back to the observation points, carrying information about the obstacle.

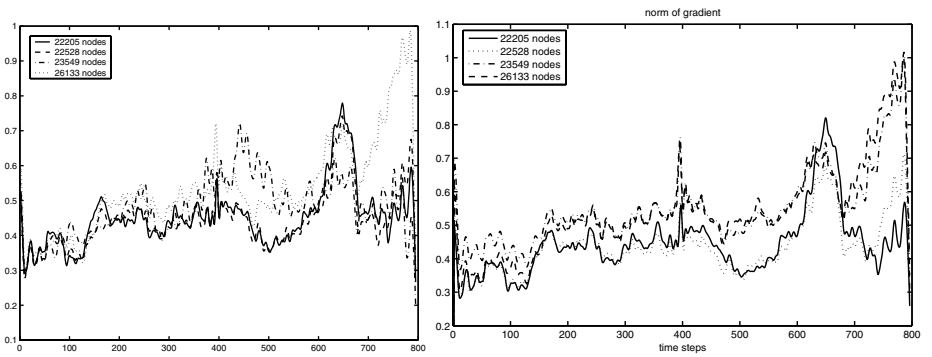
We start the optimization algorithm with guess values of the parameter $c = 1.0$ at all points in the computational domain and a regularization parameter $\gamma = 0.1$. The computations were performed on the four adaptively refined meshes shown in Figure 7.3. Here, we solved the adjoint problem backward in time from $t = 40.0$ down to $t = 0.0$. In Figure 7.4 we display isosurfaces of the computed solution of the adjoint problem on different adaptively refined meshes at the time $t = 0.0$. We observe that the isosurfaces of the adjoint solution are concentrated around the interface between the two cylinders and in the inclusion. There we will also perform local refinement of the mesh, since the residual R_{α_1} in the a posteriori error estimator includes the term $\left|\frac{\partial \lambda_h}{\partial t}\right|$ involving solution of the adjoint problem. In Figure 7.5 we show a comparison



(A) 22528 *nodes* (B) 23549 *nodes*

(C) 26133 *nodes* (D) 33138 *nodes*

FIG. 7.4. *Isosurfaces of the adjoint problem solution on different adaptively refined meshes. We show the solution on the third optimization iteration at time* $t = 0.0$ *in* Ω_{FEM}.

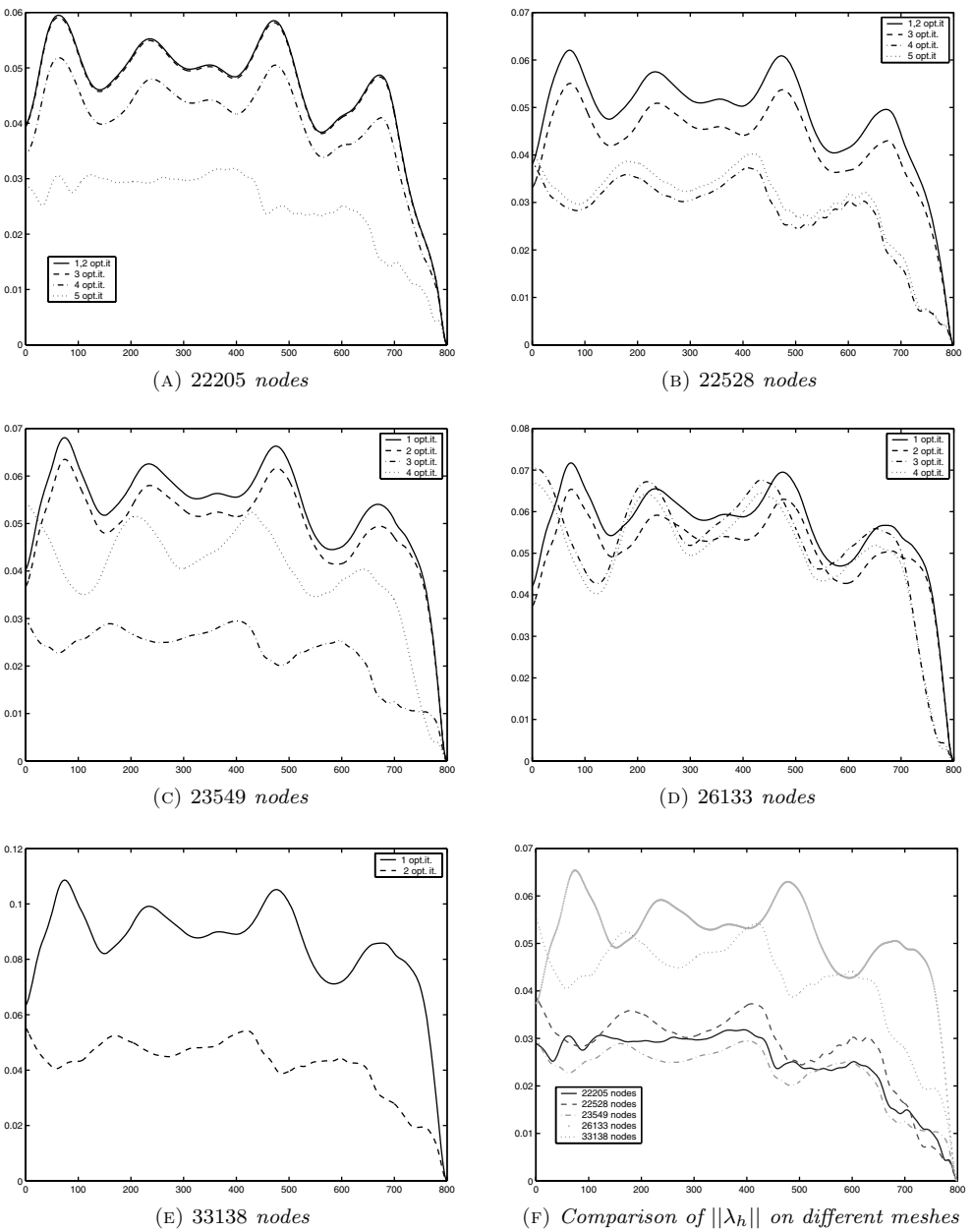


(A) R_{α_1} (B) $||\frac{\partial L}{\partial \alpha}||$

FIG. 7.5. *Comparison of* (A) R_{α_1} *and* (B) $||\frac{\partial L}{\partial \alpha}||$ *on different adaptively refined meshes. We present the smallest value of* R_{α_1} *and* $||\frac{\partial L}{\partial \alpha}||$ *on the corresponding meshes. Here the* x*-axis denotes time steps on* $[0, 40]$.

of R_{α_1} and $||\frac{\partial L}{\partial \alpha}||$ over the time interval $[0, 40]$ on different adaptively refined meshes. Here, the smallest values of the residual R_{α_1} and the L^2 norm in space of $\frac{\partial L}{\partial \alpha}$ are shown on the corresponding meshes.

The L^2 norms in space of the adjoint solution λ_h over the time interval $[0, 40]$ on different optimization iterations on adaptively refined meshes are shown in Figure 7.6. Figure 7.6(A)–(E) display the norm at each optimization iteration on the different



(A) 22205 *nodes* (B) 22528 *nodes*

(C) 23549 *nodes* (D) 26133 *nodes*

(E) 33138 *nodes* (F) *Comparison of* $||\lambda_h||$ *on different meshes*

FIG. 7.6. (A)–(E) L^2 *norms in space of the adjoint problem solution* λ_h *on adaptively refined meshes at different optimization iterations,* (F) *comparison of* $||\lambda_h||$ *on different adaptively refined meshes. Here the x-axis denotes time steps on* $[0, 40]$.

TABLE 7.1. $||v - v_{obs}||$ *on adaptively refined meshes.*

Opt.it.	22205 nodes	22528 nodes	23549 nodes	26133 nodes	33138 nodes
1	0.0506618	0.059448	0.0698214	0.0761904	0.120892
2	0.050106	0.0594441	0.0612598	0.063955	0.0358431
3	0.0358798	0.0465678	0.028501	0.0618176	
4	0.0244553	0.0413165			
5	0.0219676				

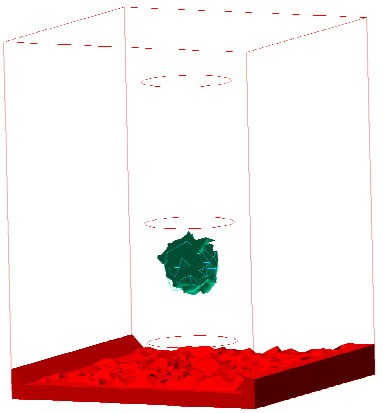
adaptively refined meshes. The norm of the adjoint solution decreases faster on finer meshes. Figure 7.6(F) shows a comparison of the L^2 norms in space of the adjoint solution λ_h on different meshes. Here, we present the smallest L^2 norm of λ_h on the corresponding meshes.

In Table 7.1 we give computed L^2 norms of $v - v_{obs}$ on different adaptively refined meshes at each optimization iteration while the norms decrease. The computational tests show that the best results are obtained on a four times adaptively refined mesh, where $||v - v_{obs}||$ is reduced approximately by a factor four between two optimization iterations. We note that the L^2 norms in space and time of $v - v_{obs}$ differ from coarser to finer meshes because of the increase in the number of observation points at the lens boundary during the refinement procedure. This also means that we capture higher and higher frequency content of the data with each refinement step.

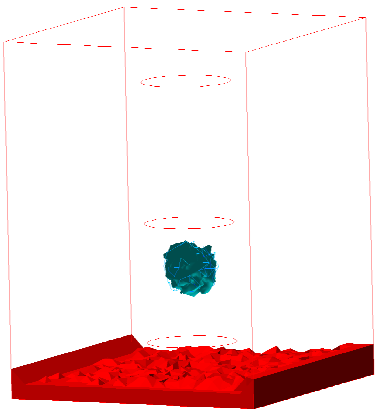
The reconstructed parameter c on different adaptively refined meshes in the final optimization iteration is presented in Figure 7.7. We show isosurfaces of the parameter field $c(x)$, indicating domains with a given parameter value. We see that although the qualitative reconstruction on the coarse grid is already good enough to recover the shape of the inclusion even from limited boundary data, the quantitative reconstruction becomes acceptable only on the refined grids. Additionally, with successive refinement, the boundary of the reconstructed inclusion becomes sharper (compare the isosurface in Figure 7.7(A), (B) with those in Figure 7.7(D), (E)). On the grid with 33138 nodes (Figure 7.7(F)), the parameter in the inclusion is calculated as $c \approx 0.51$, compared to the exact value of $c = 0.5$.

However, since the quasi-Newton method is only locally convergent, the values of the identified parameters are very sensitive to the starting values of the parameters in the optimization algorithm and also to the values of the regularization parameter γ and step length ρ in the velocity upgrade. Therefore, to achieve more stable reconstruction, we enforce that the parameter c belongs to C_M by putting box constraints on the computed parameters and using a smoothness indicator to update values of c at the new optimization iteration by local averaging over the neighboring elements.

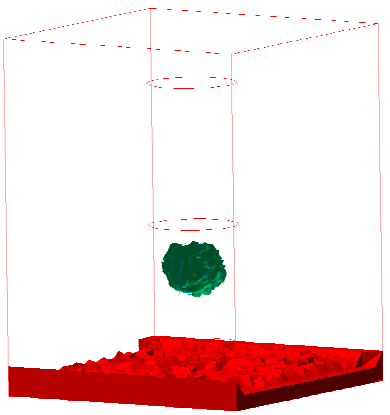
8. Conclusion. We present an explicit, adaptive, hybrid FEM/FDM method for an inverse scattering problem in scanning acoustic microscopy. The method is hybrid in the sense that different numerical methods, finite elements and finite differences, are used in different parts of the computational domain. The adaptivity is based on a posteriori error estimates for the associated Lagrangian in the form of space-time integrals of residuals multiplied by dual weights, which allows stable reconstruction of a parameter from data given on only a small part of the boundary. Their usefulness for adaptive error control is illustrated on an inverse scattering problem for scanning acoustic microscopy in three dimensions. Future work is concerned with choosing optimal regularization parameters for the functional to be minimized, and extending the adaptive algorithm to use an a posteriori error estimator for the parameter by solving an associated problem for the Hessian of the Lagrangian.



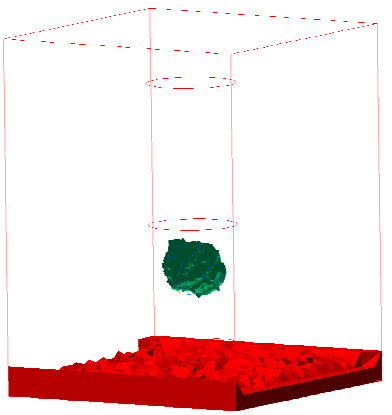
(A) 22528 *nodes,* $c \approx 0.66$



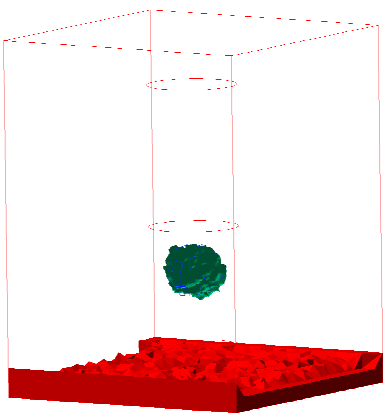
(B) 22528 *nodes,* $c \approx 0.623$



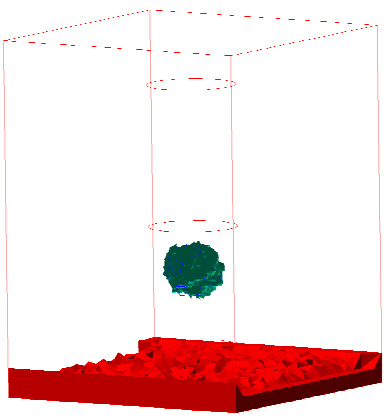
(C) 23549 *nodes,* $c \approx 0.556$



(D) 26133 *nodes,* $c \approx 0.547$



(E) 26133 *nodes,* $c \approx 0.531$



(F) 33138 *nodes,* $c \approx 0.51$

FIG. 7.7. *Reconstructed parameter* $c(x)$ *on different adaptively refined meshes. Isosurfaces of the parameter field* $c(x)$*, indicating domains with a given parameter value, are shown.*

Acknowledgments. We thank Prof. Mikhail Klibanov and Dr. Maxim Shishlenin for useful comments and suggestions. We would also like to express our thanks to both referees for their comments and suggestions on improving the presentation.

REFERENCES

[1] V. Akcelik, G. Biros, and O. Ghattas, *Parallel multiscale Gauss–Newton–Krylov methods for inverse wave propagation*, in Supercomputing '02: Proceedings of the 2002 ACM/IEEE conference on Supercomputing, Los Alamitos, CA, 2002, IEEE Computer Society Press, Piscataway, NJ, pp. 1–15.

[2] C. Bardos, G. Lebeau, and J. Rauch, *Sharp sufficient conditions for the observation, control, and stabilization of waves from the boundary*, SIAM J. Control Optim., 30 (1992), pp. 1024–1065.

[3] R. Becker and R. Rannacher, *An optimal control approach to a posteriori error estimation in finite element methods*, in Acta Numerica, 2001, Acta Numer., Cambridge University Press, Cambridge, UK, 2001, pp. 1–102.

[4] L. Beilina, *Adaptive FEM method for an inverse scattering problem*, J. Inverse Problems Inform. Tech., 3 (2002), pp. 73–116.

[5] L. Beilina, *Adaptive hybrid finite element/difference methods: Application to inverse elastic scattering*, J. Inverse Ill-posed Problems, 6 (2003), pp. 585–618.

[6] L. Beilina, *A hybrid FEM/FDM method for elastic waves*, J. Appl. Comput. Math., 2 (2003), pp. 13–29.

[7] L. Beilina, *Adaptive finite element/difference method for inverse elastic scattering waves*, J. Appl. Comput. Math., 2 (2004), pp. 158–174.

[8] L. Beilina and C. Johnson, *A hybrid FEM/FDM method for an inverse scattering problem*, in Numerical Mathematics and Advanced Applications—ENUMATH 2001, Springer-Verlag, Berlin, 2001, pp. 545–556.

[9] L. Beilina and C. Johnson, *A posteriori error estimation in computational inverse scattering*, Math. Models Methods Appl. Sci., 15 (2005), pp. 23–37.

[10] L. Beilina, K. Samuelsson, and K. Ahlander, *Efficiency of a hybrid method for the wave equation*, in Finite Element Methods (Jyväskylä 2000), GAKUTO Internat. Ser. Math. Sci. Appl. 15, Gakkotosho, Tokyo, 2001, pp. 9–21.

[11] M. I. Belishev and Y. V. Kurylev, *Boundary control, wave field continuation and inverse problems for the wave equation*, Appl. Comp. Math., 22 (1991), pp. 27–52.

[12] A. Briggs, *Acoustic Microscopy*, Clarendon Press, Oxford, UK, 1992.

[13] A. Bukhgeim and M. Klibanov, *Global uniqueness of a class of multidimensional inverse problems*, Sov. Math. Dokl., 24 (1981), pp. 244–247.

[14] C. Bunks, F. M. Saleck, S. Zaleski, and G. Chavent, *Multiscale seismic waveform inversion*, Geophysics, 60 (1995), pp. 1457–1473.

[15] G. Chavent and C. A. Jacewitz, *Determination of background velocities by multiple migration fitting*, Geophys., 60 (1995), pp. 476–490.

[16] D. Colton, J. Coyle, and P. Monk, *Recent developments in inverse acoustic scattering theory*, SIAM Rev., 42 (2000), pp. 369–414.

[17] J. E. Dennis, Jr., and J. J. Moré, *Quasi-Newton methods, Motivation and theory*, SIAM Rev., 19 (1977), pp. 46–89.

[18] H. W. Engl, M. Hanke, and A. Neubauer, *Regularization of Inverse Problems*, Kluwer Academic Publishers, Dordrecht, The Netherlands, 1996.

[19] B. Engquist and A. Majda, *Absorbing boundary conditions for the numerical simulation of waves*, Math. Comp., 31 (1977), pp. 629–651.

[20] K. Eriksson, D. Estep, P. Hansbo, and C. Johnson, *Introduction to adaptive methods for differential equations*, in Acta Numerica, 1995, Acta Numer., Cambridge University Press, Cambridge, UK, 1995, pp. 105–158.

[21] K. Eriksson, D. Estep, P. Hansbo, and C. Johnson, *Computational Differential Equations*, Studentlitteratur/Cambridge University Press, Lund, Sweden/Cambridge, UK, 1996.

[22] X. Feng, S. Lenhart, V. Protopopescu, L. Rachele, and B. Sutton, *Identification problem for the wave equation with Neumann data input and Dirichlet data observations*, Nonlinear Anal., 52 (2003), pp. 1777–1795.

[23] T. J. R. Hughes, *The Finite Element Method*, Prentice–Hall, Englewood Cliffs, NJ, 1987.

[24] M. Ikawa, *Hyperbolic Partial Differential Equations and Wave Phenomena*, American Mathematical Society, Providence, RI, 2000.

[25] O. Y. Imanuvilov and M. Yamamoto, *Global uniqueness and stability in determining coefficients of wave equations*, Comm. Partial Differential Equations, 26 (2001), pp. 1409–1425.
[26] V. Isakov, *Inverse Problems for Partial Differential Equations*, Springer, Berlin, 1998.
[27] C. Johnson, *Adaptive computational methods for differential equations*, in ICIAM99, Oxford University Press, Oxford, UK, 2000, pp. 96–104.
[28] J. Kaipio and E. Somersalo, *Statistical and Computational Inverse Problems*, Springer, Berlin, 2005.
[29] M. Klibanov and A. A. Timonov, *Carleman estimates for coefficient inverse problems and numerical applications*, VSP Publishing, Utrecht, The Netherlands, 2004.
[30] J. L. Lions, *Optimal Control of Systems Governed by Partial Differential Equations*, Springer, Berlin, 1971.
[31] J. L. Lions and E. Magenes, *Non-Homogeneous Boundary Value Problems and Applications. Vols.* I *and* II, Springer-Verlag, Berlin, 1972.
[32] F. Natterer and F. Wubbeling, *A propagation-backpropagation method for ultrasound tomography*, Inverse Problems, 11 (1995), pp. 1225–1232.
[33] J. Nocedal, *Updating quasi-Newton matrices with limited storage*, Math. Comput., 35, (1991), pp. 773–782.
[34] O. Pironneau, *Optimal Shape Design for Elliptic Systems*, Springer Verlag, Berlin, 1984.
[35] R.-E. Plessix, Y.-H. De Roeck, and G. Chavent, *Waveform inversion of reflection seismic data for kinematic parameters by local optimization*, SIAM J. Sci. Comput., 20 (1999), pp. 1033–1052.
[36] C. Prada, F. Wu, and M. Fink, *The iterative time reversal mirror: A solution to self-focusing in the pulse echo mode*, J. Acoust. Soc. Amer., 90 (1991), pp. 1119–1129.
[37] J.-P. Puel and M. Yamamoto, *Generic well-posedness in a multidimensional hyperbolic inverse problem*, J. Inverse Ill-posed Problems, 5 (1997), pp. 55–83.
[38] Rakesh, *Reconstruction for an inverse problem for the wave equation with constant velocity*, Inverse Problems, 6 (1990), pp. 91–98.
[39] V. G. Romanov, *Inverse Problems of Mathematical Physics*, VNU Science Press, Ultrecht, The Netherlands, 1987.
[40] V. G. Romanov, *Problem of determining the speed of sound*, Sib. Math., 30 (1990), pp. 598–605 (English translation by Plenum).
[41] K. Shirota, *A numerical method for the problem of coefficient identification of the wave equation based on a local observation of the boundary*, Comm. Korean Math. Soc., 16 (2001), pp. 509–518.
[42] H. Sielschott, *Rückpropagationsverfahren für die Wellengleichung in bewegtem Medium (Back-propagation method for the wave equation in moving media)*, Ph.D. thesis, Mathematisch-Naturwissenschaftliche Fakultät, Universität Münster, Münster, Germany, 2000.
[43] U. Tautenhahn, *Tikhonov regularization for identification problems in differential equations*, in Parameter Identification and Inverse Problems in Hydrology, Geology and Ecology (Karlsruhe, 1995), Kluwer Academic Publishers, Dordrecht, The Netherlands, 1996, pp. 261–270.
[44] U. Tautenhahn and Q.-N. Jin, *Tikhonov regularization and a posteriori rules for solving nonlinear ill posed problems*, Inverse Problems, 19 (2003), pp. 1–21.
[45] A. N. Tikhonov and V. Y. Arsenin, *Solution of Ill-posed Problems*, Wiley, New York, 1977.
[46] V. H. Weston, *Invariant imbedding and wave splitting in* $\boldsymbol{R}^d$, Inverse Problems, 8 (1992), pp. 919–947.

SIAM J. SCI. COMPUT.
Vol. 28, No. 2, pp. 403–420
© 2006 Society for Industrial and Applied Mathematics

WEIGHTED MATCHINGS FOR PRECONDITIONING SYMMETRIC INDEFINITE LINEAR SYSTEMS*

MICHAEL HAGEMANN† AND OLAF SCHENK†

Abstract. Maximum weight matchings have become an important tool for solving highly indefinite unsymmetric linear systems, especially in direct solvers. In this study we investigate the benefit of reorderings and scalings based on symmetrized maximum weight matchings as a preprocessing step for incomplete $\mathrm{LDL}^{\mathrm{T}}$ factorizations. The reorderings are constructed such that the matched entries form 1×1 or 2×2 diagonal blocks in order to increase the diagonal dominance of the system. During the incomplete factorization only tridiagonal pivoting is used. We report results for this approach and comparisons with other solution methods for a diverse set of symmetric indefinite matrices, ranging from nonlinear elasticity to interior point optimization.

Key words. symmetric indefinite matrix, preconditioning, maximum weight matching, incomplete factorization, tridiagonal pivoting

AMS subject classifications. 05C70, 65F10, 65F50

DOI. 10.1137/040615614

1. Introduction.

We consider the iterative solution of symmetric and indefinite linear systems $A\boldsymbol{x} = \boldsymbol{b}$, where A is preconditioned by an incomplete $\mathrm{LDL}^{\mathrm{T}}$ factorization, which is also indefinite. We propose a preconditioning method based on a combination of symmetric weighted matchings and tridiagonal pivoting techniques, which significantly improves the robustness of the preconditioning for important classes of symmetric indefinite linear systems. The objective of this approach is to restrict the amount of pivoting during the incomplete factorization phase by obtaining a good initial pivoting order through a weighted matching in a preprocessing step. The tridiagonal pivoting strategy chooses either the 1×1 pivot on the diagonal or the 2×2 pivot from the current column and the immediately succeeding one, based on a simple factor growth criterion. The use of weighted matchings as an approximation of the pivoting order in the context of sparse direct solution methods for symmetric indefinite linear systems was introduced by Duff and Gilbert in [10]. This work was recently elaborated by Duff and Pralet [13] and Schenk and Gärtner [33]. While those previous works concentrated on sparse direct solution methods, we present approaches for the use of these techniques in preconditioning.

The matching is effective only if the diagonal entries of A are not consistently larger than the off-diagonal entries. Therefore it has no effect on many shifted systems like Helmholtz problems or matrices arising in shift-invert eigensolvers. This is reflected in the results, where especially saddle-point problems benefit from the preprocessing.

After preprocessing, the incomplete factorization preconditioner is applied to the reordered and scaled original system. A reordering Q, based on a symmetrized maximum weight matching of the matrix A, moves large off-diagonal entries into 2×2 diagonal blocks. An additional fill-reducing reordering P_Q, which maintains the

*Received by the editors September 23, 2004; accepted for publication (in revised form) September 14, 2005; published electronically March 31, 2006. This work was supported by the Swiss Commission for Technology and Innovation (CTI) under grant 7036 ENS-ES.
http://www.siam.org/journals/sisc/28-2/61561.html
†Departement Informatik, University of Basel, Klingelbergstr. 50, 4056 Basel, Switzerland (michael.hagemann@unibas.ch, olaf.schenk@unibas.ch).

diagonal block structure, is applied to minimize the fill-in during the factorization phase. In the following, we call the combination of these two reorderings sym2x2block. Furthermore, the matrix is scaled by a symmetric scaling D_Q, which is also acquired through the weighted matching algorithm. Thus, the iterative method solves a linear system of the form

$$M^{-1}\hat{A}\hat{\boldsymbol{x}} = M^{-1}\hat{\boldsymbol{b}}, \tag{1.1}$$

where $\hat{A}$ is the scaled and reordered original linear system

$$\hat{A} = P_Q Q D_Q \; A \; D_Q Q^T P_Q^T. \tag{1.2}$$

The incomplete factorization preconditioner M consists of a lower triangular factor L and a block diagonal matrix D with 1×1 and 2×2 blocks.

Sparse symmetric indefinite linear systems arise in numerous areas ranging from incompressible flow computations to linear and nonlinear optimization, electromagnetic scattering, and finite element analysis, among others. Particularly for very large systems, efficient and scalable preconditioners are sought because the memory requirements and factorization costs of direct methods become increasingly prohibitive in this case.

1.1. Related work. The iterative solution of indefinite linear systems is an active research area, and a variety of approaches have been proposed. Many preconditioning approaches exploit the inherent block structure of the given problem type. Specifically for the preconditioning of saddle-point systems, a number of block preconditioning methods have been proposed, some of which are optimal in some sense. For a comprehensive survey of solution and preconditioning approaches, the reader is referred to [5]. Recent overviews of block preconditioning techniques include [2] and [38]. In the following we briefly review some of the algebraic approaches to solving symmetric indefinite linear systems.

Direct solvers are a common option for solving general indefinite problems. Since LU and LDL^T factorizations of indefinite matrices may break down or yield bad results due to small or zero diagonal entries, a number of techniques exist that try to avoid or alleviate these problems, some of which were specifically designed for symmetric matrices. There are basically three approaches, namely [illegible] techniques, [illegible] of the diagonal, and [illegible]. The classical pivoting approach for symmetric matrices is Bunch–Kaufman pivoting [8], which searches for 2×2 diagonal blocks during the factorization. An adaption to sparse matrices was introduced by Duff and Reid in [14] (see also [1]). In [20], Gould, Hu, and Scott performed a comprehensive study of current direct solvers for indefinite linear systems.

Fewer methods exist for symmetric incomplete factorizations. Gill et al. [19] propose using factors of MA27 with all negative diagonal entries in 2×2 blocks turned into positive ones, thus clustering the eigenvalues of the indefinite system around -1 and 1. Li and Saad [29] consider sparse incomplete LDL^T factorizations with Bunch–Kaufman pivoting. A number of unsymmetric methods have been proposed that are also applicable to symmetric systems. Chow and Saad in [9] conducted a comprehensive study of unsymmetric incomplete factorization techniques to solve indefinite systems. In [6], Benzi, Haws, and Tůma study unsymmetric permutations based on weighted matchings as a preprocessing step for various incomplete or approximate factorizations.

In [18] Freund and Jarre devise an indefinite block SSOR method that uses a greedy approximation of a maximum weight matching to find nonsingular 2×2

diagonal blocks in interior point optimization problems. Freund mentions a similar technique based on LDL^T factorizations in [17], but without giving further details.

1.2. Contributions. We conduct an experimental study on the use of weighted matchings in the preconditioning of symmetric indefinite linear systems with incomplete factorization techniques. We concentrate on simple approaches to leverage the information provided by the matchings. To the best of our knowledge, there are as yet no other studies that investigate the potential of these techniques in the context of preconditioning. Similar techniques have recently been explored for direct solvers [13, 33]. In order to assess the competitiveness of the approach, we conduct comparisons with several other relevant and comparable preconditioning methods and with a direct solver.

Section 2 introduces the matching algorithms used in this study. In section 3 we describe our approach to the problem of adapting the matchings to the preconditioning of symmetric matrices. The incomplete factorization is described in section 4. Numerical results and comparisons with other solution techniques are presented in section 5. In closing, we sketch some of the possible research directions we are currently considering.

2. Weighted matchings.

2.1. Introduction. The use of weighted matchings as a static approximation of the pivoting order in direct solvers was first proposed by Olschowka and Neumaier in 1996 [31]. The first implementations for sparse matrix problems were introduced by Duff and Koster [11] and Gupta and Ying [21]. The objective of this approach is to restrict the pivoting during the factorization phase, by obtaining a good pivoting order through a matching in a preprocessing step. Duff and Koster provide results of the effectiveness of this preprocessing step for both direct and iterative unsymmetric solvers [12]. Benzi, Haws, and Tůma [6] and Schenk, Röllin, and Gupta [35] conducted a more extensive analysis of these reorderings in the context of preconditioning. Up until recently, only unsymmetric permutations based directly on the matchings were considered. In 2002, Gilbert and Duff presented ideas on how to use matchings for the direct solution of symmetric systems while maintaining the symmetry [10]. Here, we build on some of their ideas and apply them in the context of preconditioning for symmetric indefinite systems.

2.2. Matching algorithms. The success of matchings as preprocessings in linear solvers is due in large part to the practical efficiency of the deployed matching algorithms. These algorithms work on associate graph representations of the matrices. In our case, the algorithms work on the [illegible] $G_A = (V_r, V_c, E)$, where V_r and V_c are vertex sets of cardinality n, representing the rows and columns, respectively, of the matrix and $E = \{(i,j) \mid a_{ij} \neq 0\}$ is the set of edges connecting the vertices in V_r and V_c.

Now we are looking for a subset $\mathcal{M} \subseteq E$ with the following properties: (a) for all vertices $v \in \{V_r, V_c\}$, exactly one edge $e \in \mathcal{M}$ is incident to v, and (b) the matched edges $e \in \mathcal{M}$ maximize a weight function $w(\cdot)$, with

$$w(\mathcal{M}) = \sum_{(i,j)\in\mathcal{M}} (C)_{ij}, \tag{2.1}$$

where C is a weight coefficient matrix for the edges of A, with $C = A$ as the simplest case. Such a subset $\mathcal{M}$ is called a [illegible] of G_A, and

the problem of finding it is known as the [illegible] or [illegible]. The first condition can always be satisfied if A is a structurally nonsingular matrix. The weight function (2.1) does not necessarily define a unique matching, however.

As of today, the most efficient algorithms for finding weighted matchings are based on [illegible]. Gupta and Ying state $\mathcal{O}(n(\tau + n \log n))$ as the upper bound for the time complexity of their implementation, where τ represents the number of nonzeros in A. Duff and Koster give an upper bound of $\mathcal{O}(n(\tau + n) \log n)$. For details of the implementations, see [12, 21]. An important feature of these algorithms in practice is that they typically behave much better than may be expected from the complexity analysis, which is also what we observed in our tests. Our implementation is based on [21].

2.3. Weight criteria. An important consideration for the matching is the choice of weighting. As the matching algorithms generally maximize (2.1), the criterion has to be formulated in terms of the weight coefficients c_{ij}. Typically, the coefficients c_{ij} are modified subsequently, either for better performance of the matching or to enforce the above conditions. See [21] for a survey of such techniques.

If we want to maximize the [illegible] A, the coefficient matrix C can simply be defined by $c_{ij} = |a_{ij}|$. The maximization of the [illegible] A can be acquired by using a simple logarithmic transformation:

$$c_{ij} = \begin{cases} \log |a_{ij}| & \text{if } a_{ij} \neq 0, \\ -\infty & \text{otherwise.} \end{cases} \tag{2.2}$$

Another criterion that is suggested in [12] is the maximization of the [illegible]. This requires a modified matching algorithm, called a "bottleneck matching." Since experimental results [6, 12, 35] suggest that the maximum product approach is generally the most beneficial, we use it exclusively.

2.4. Scalings. Perfect matchings of maximum weight have the property that dual variables u_i and v_i exist such that

$$\begin{cases} u_i + v_j = c_{ij} & \text{if } (i,j) \in \mathcal{M}, \\ u_i + v_j \leq c_{ij} & \text{otherwise.} \end{cases} \tag{2.3}$$

The augmenting path algorithms build these variables during the matching phase.

If the logarithmic transformation (i.e., (2.2)) was used as the weight matrix, the dual variables can be used to define row and column scalings D_r and D_c by setting $(D_r)_i = \exp(u_i)$ and $(D_c)_i = \exp(v_i)$, such that the following property holds:

$$|(A)_{ij}| = |(D_r A D_c)_{ij}| \quad \begin{cases} = 1 & \text{if } (i,j) \in \mathcal{M}, \\ \leq 1 & \text{otherwise.} \end{cases} \tag{2.4}$$

In theory, this scaling has two advantages. First, it provides a good [illegible] that is beneficial during the elimination process, especially for badly conditioned matrices. Second, the diagonal dominance of the permuted linear system may be increased. This is not always the case, however. Benzi, Haws, and Tůma [6] observed that the scaling can have a bad effect, probably because in some cases it weakens the diagonal dominance. We saw a similar behavior for some matrices, but the overall performance was better for the scaled systems.

Duff and Pralet show in [13] that for symmetric matrices the two scalings can be symmetrized by taking the componentwise square root of their product,

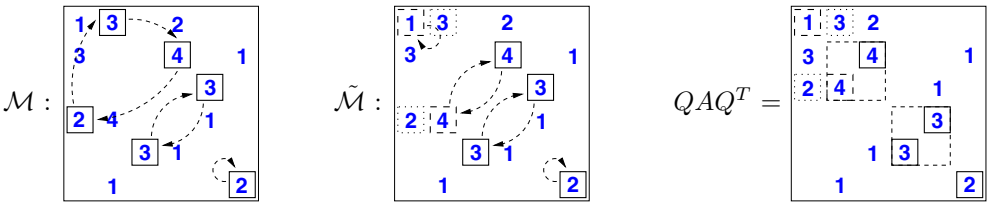


FIG. 3.1. *Illustration of the symmetric permutation Q, with $\mathcal{M} = (124)(35)(6)$ and $\tilde{\mathcal{M}} = (1)(24)(35)(6)$. The split matching $\tilde{\mathcal{M}}$ has two additional entries (indicated by dashed boxes), while two entries of the original matching were discarded (dotted boxes). The two 2-cycles are then permuted into 2×2 diagonal blocks.*

$D_Q = \sqrt{D_r \cdot D_c}$. Our tests show that the scaling is generally beneficial, although not for all matrices. This is in line with the observations in [6].

3. Construction of the sym2x2block reordering.

3.1. Symmetric matchings. In order to form diagonal 2×2 blocks out of the matched entries, we ideally want a [illegible] matching of maximum weight, i.e., a matching where $(i, j) \in \mathcal{M} \Rightarrow (j, i) \in \mathcal{M}$. Given such a matching, the entries (i, j) and (j, i) are readily reordered into 2×2 diagonal blocks. Duff and Pralet showed that the problem of finding symmetric maximum weight matchings in a symmetric matrix is equivalent to finding a matching in an undirected graph [13].

While these findings may be helpful in devising a special matching algorithm for this problem, to the best of our knowledge no method currently exists that is competitive with bipartite maximum weight matching algorithms. Therefore, we deploy the well-established matching algorithms and construct the desired reorderings in an additional postprocessing step. This additional step is of linear complexity and follows the approach presented by Duff and Gilbert [10], who note that in order to symmetrize a bipartite matching, one can [illegible] in the matchings such that only cycles of length one and two remain. The matching $\mathcal{M}$ in Figure 3.1, for example, contains three cycles, with the first cycle (124) being the only one longer than two. Interpreted as the associated row permutation, this cycle would permute row 1 into row 2, row 2 into row 4, and row 4 into row 1. To split this cycle, one can pick one of the corresponding diagonal entries (a_{11}, a_{22}, or a_{44}) as a singleton and pair the remaining entry with its symmetric counterpart. This procedure can be applied to larger cycles as well. Details of our implementation are given in section 3.3.

This approach, however, does not guarantee that we will find an optimal symmetric matching of maximum weight for the matrix. One can construct matrices whose maximum weight matching does not contain edges that appear in the symmetric maximum weight matching of the same matrix. Thus, a simple postprocessing step is in general not able to find the optimal symmetric matching. The implications and possible remedies of this problem are still an open research topic.

3.2. Overview. The construction of the preordering involves several steps. The splitting of cycles of length larger than two yields a set of 1- and 2-cycles, which can be used to build a reordering Q that moves the matched entries from the 2-cycles into 2×2 diagonal blocks. This part of the preprocessing is illustrated in Figure 3.1, and, as pseudocode, in lines 1–4 of Algorithm 3.1. Details are given in section 3.3.

After the block diagonal structure is established, an additional fill-reducing reordering is computed, which maintains the block diagonal structure. This is achieved through compression of the graph of QAQ^T by merging the 2×2 diagonal blocks

into simple vertices. The fill-reducing reordering P_Q can then be constructed from the reordering of the compressed graph and two helper arrays. This step is detailed in section 3.4 and reflected in lines 5–7 of the pseudocode.

ALGORITHM 3.1.

1. $\mathcal{M}$ D_r D_c =
2. =
3. $\tilde{\mathcal{M}}$ =
4. 1x1 = 22 $\tilde{\mathcal{M}}$
5. A_{comp} = 2x2 A Q 1, 1
6. P_{comp} = A_{comp}
7. P_Q = 2x2 P_{comp} 1, 1
8. D_Q = $\sqrt{D_r \cdot D_c}$
9. $\hat{A}$ = $P_Q Q$ $D_Q A D_Q$ $Q^T P_Q^T$

3.3. Splitting strategy. Starting from the cycle representation of the matching $\mathcal{M}$, a split version $\tilde{\mathcal{M}}$ is constructed. The splitting involves a number of considerations, as there are two possibilities for splitting an even cycle and m possibilities for splitting an m-cycle of odd length. An even cycle can be split starting from the first or the second entry, pairing the last and the first entry of the cycle. In an odd cycle, the choice of the first entry that will form the single 1-cycle determines the structure of the whole splitting.

Two criteria for the choice of the splitting have been proposed [10, 13]. One criterion focuses on the weight of the resulting transversal. In principle, this criterion can be applied to the scaled as well as to the original matrix. In [10], Duff and Gilbert consider only the scaled matrix, and show that the splitting of even cycles in that context always preserves the weight of the scaled original matching.

In [13], Duff and Pralet propose an alternative approach, referred to as "(relaxed) constrained orderings," which combines structural aspects of the splitting with criteria on the numerical values. The structural criterion focuses on the nonzero structure of the 2×2 block columns in the reordered matrix. Ideally, the two columns have the same nonzero structure, such that the combined block column does not introduce additional nonzero entries. Under certain conditions, some of the 2×2 blocks that have been determined based on the structural criterion are "relaxed," i.e., split into scalar pivots. The relaxation criteria depend on the numerical values in the 2×2 block. The objective of this approach is to acquire better reorderings from the subsequent fill-reducing method (see also section 3.4).

In our implementation, we just use the simple static splitting described below and do not split any of the 2×2 blocks found. The main reason for that approach is that we apply a numerical criterion for the choice of the pivot later during the factorization, when all relevant numerical information for the pivoting is available. Furthermore, our tests with relaxed approaches indicate that the split pivots can be a source of instability, while the benefit in terms of fill-in is marginal.

In splitting even cycles, we always begin with the first entry, whereas with odd cycles, we begin by choosing the largest diagonal entry in the cycle as the solitary entry. As most of the matchings consist mainly of 1- and 2-cycles, we did not investigate effects of other splitting strategies on the resulting fill-in.

After a set of 1- and 2-cycles has been acquired from the original matching, the function `build_diag22_reordering` constructs a reordering which yields the 2×2 diagonal block structure by moving both entries of each 2-cycle into adjacent rows and columns in the reordered matrix.

3.4. Combination with fill-reducing reorderings. In order to construct the factorization efficiently, care has to be taken that not too much fill-in is introduced during the elimination process. In order to combine the sym2x2block reordering with a fill-reducing reordering, we [illegible] the graph of the reordered system and apply a fill-reducing reordering to the compressed graph. In the compression step, the union of the structure of the two rows and columns corresponding to a 2×2 diagonal block are built and used as the structure of a single compressed row and column representing the union of the original ones. A fill-reducing reordering P_{comp} is then computed for the compressed graph. This reordering can then be expanded, so that it provides the reordering P_Q, which reduces the fill-in during the factorization, while maintaining the 2×2 block structure of the diagonal.

In Algorithm 3.1 this step is represented in lines 5–7. The graph of the reordered matrix is compressed, and the nodes that correspond to scalar pivots are marked in the [illegible]1x1 array. Furthermore, a reverse mapping [illegible] is constructed, which maps the nodes of the compressed graph back to their original index. Together with the permutation of the condensed system P_{comp}, these two arrays allow the construction of P_Q.

4. Incomplete LDL^T factorization with tridiagonal pivoting.

4.1. Algorithm. We use a symmetric variant of the Crout approach, which is also called a "left-looking" approach [15]. The implementation does not rely on BLAS routines, since the largest blocks we consider are the 2×2 block pivots. The L factor is constructed columnwise. To build the kth column of the factor, the nonzero entries[1] $A_{k:n,\,k}$ of the original matrix are copied, and updates from all factored columns from the left with couplings to k are added. Then a dropping strategy is applied in order to limit the amount of fill-in introduced by the updates. Finally, the remaining entries of the column are divided by the diagonal pivot entry. According to common practice, we refer to the updated, but not yet factored, part of the matrix as $A^{(k)}$, which is also called the "reduced matrix." As we are just updating one column at a time, we have access only to the part $A^{(k)}_{k:n,\,k}$ of the reduced matrix.

This approach has the nice property that in each step of the factorization we have all the values of a specific column of the factor at our disposal and can thus apply sophisticated dropping rules. After we have determined its favored entries, the column is used only for updates on later columns. Since we expect a preprocessed matrix with a "heavy" diagonal, we employ only a tridiagonal pivoting criterion due to Bunch [7], which chooses 1×1 or 2×2 pivots based solely on the diagonal and the subdiagonal entry in the column. If a 2×2 pivot is chosen, the two corresponding columns of the factor L are stored as a block column with identical sparsity pattern. The 2×2 pivots are explicitly inverted with a scaled inversion formula. Dropping in block columns is performed according to the ∞-norms of the 2×1 entries.

The following sections give a detailed description of the pivoting criterion and the dropping strategy.

4.2. Pivoting strategy. Symmetric pivoting strategies in general try to (a) find a suitable 1×1 or 2×2 diagonal pivot such that (b) the growth in the factors is bounded [1]. In the case of sparse matrices, an additional criterion to control the fill-in introduced by dynamic permutations is required [14]. Our experiments with the described preprocessing step indicated, however, that the preordering already

[1] We use MATLAB notation to indicate submatrices.

yields a good approximation to a feasible pivoting order. Only marginal benefits in terms of robustness could be achieved, almost regardless of the amount of work put into additional pivoting. While we could reproduce results similar to those found by Li and Saad [29], these Bunch–Kaufman-like techniques could not significantly improve the robustness and led to difficulties when the matrix contained dense rows or columns.

Thus we use a pivoting strategy that is as simple as possible. The choice of the pivot is based on a criterion introduced by Bunch [7]. Given a constant $\alpha := (\sqrt{5}-1)/2$, we choose a 2×2 pivot in step k if

$$|a_{kk}^{(k)}| \;\leq\; \alpha\, |a_{k+1,k}^{(k)}|^2, \tag{4.1}$$

where $a^{(k)}$ designates the entries in the reduced matrix after step k. The constant α is a weight that controls the choice of the pivot. It is obtained by equating the growth factor introduced by two consecutive 1×1 pivot factorization steps to the one introduced by a single factorization step with a 2×2 pivot. The original criterion includes a value σ, which reflects the largest entry in the reduced matrix. Since we calculate only one column of the reduced matrix and scale all the entries, we discarded this additional variable. See Higham [24] for a detailed derivation and for properties of this approach for the factorization of tridiagonal matrices.

The above pivoting rule does not cover the case of very small or zero pivots $a_{kk}^{(k)}$ with no suitable entry $a_{k+1,k}^{(k)}$. In order to avoid a breakdown of the algorithm, we use a simple [illegible] rule that prevents pivots from becoming smaller than ε_{piv} (see line 6 in Algorithm 4.1). We use a value of 10^{-8} for ε_{piv}, which is approximately the root of the machine precision and provided good overall results.

4.3. Dropping strategy. An important part of an incomplete factorization is the choice of an appropriate dropping strategy. This determines which entries will be kept in the factors, and should ideally strike a good balance between memory and computational savings and the quality of the preconditioning of the problem. Typical approaches include thresholds for minimal magnitudes in the factors, maximum fill-in of the factor [26, 32], and histogram-based approaches [3]. We base the dropping on the average number of entries per column of the matrix. Each column of the factor may contain at most γ_{col} entries, with $\gamma_{col} = \gamma \cdot \frac{nnz}{n}$, where nnz indicates the numbers of nonzeros in A. The entries in column k of $A^{(k)}$ are partially sorted, and the γ_{col} largest are inverted and taken across into the factor.

Another common criterion is to use a threshold criterion for the values in $A^{(k)}$. The choice of beneficial thresholds is, however, not well understood, and the benefits we saw in our experiments did not justify the added instabilities introduced by this approach. As suggested in earlier studies [26], we therefore chose the simpler strategy of using just γ as the only parameter controlling the amount of fill-in in the factors.

4.4. Summary of the incomplete factorization method. The following pseudocode provides a detailed overview of the incomplete factorization phase.

ALGORITHM 4.1. [illegible]

1. ***function:*** L D [illegible] ***ildlt-tp-factorization***
2. ***for*** [illegible] $1, \ldots, n$
3. [illegible] $a_{k:n,\,k}^{(k)}$ [illegible] $a_{k:n,\,1:k}^{(k)}$
4. ***if*** $|a_{k,k}^{(k)}| \;\geq\; \alpha |a_{k+1,k}^{(k)}|^2$: [illegible] 1x1 [illegible]
5. [illegible] γ_{col} [illegible] $a_{k:n,\,k}^{(k)}$

6. **if** $|a_{kk}| \leq \varepsilon_{piv}$ $d_{kk} := a_{kk}^{(k)}$ $d_{kk} := (\mathrm{sign}(a_{kk}^{(k)})\varepsilon_{piv})$
7. $L_{k+1:n,\,k} := d_{kk}^{-1}\, a_{k+1:n,\,k}^{(k)}$
8. ***else:*** 2x2
9. $a_{k+1:n,\,k+1}^{(k)}$
10. γ_{col} (1×2)
11. $D_k^{-1} := (a_{k:k+1,\,k:k+1}^{(k)})^{-1}$
12. $a_{k+1:n,\,k:k+1}^{(k)}$ D_k^{-1}
    $(L_{k+1:n,\,k:k+1})^T := D_k^{-1}(a_{k+1:n,\,k:k+1}^{(k)})^T$
13. ***return*** L D

5. Numerical experiments. We considered three sets of matrices that comprise a wide range of symmetric indefinite problems. The sets are described in sections 5.5 to 5.7. The tests were run on a double processor Pentium III 1.3 GHz system with 2 GB of memory under a Linux distribution based on the 2.6.9 kernel version. Only one processor was used for the testing. In order to provide realistic measurements, the timings were determined using the `gettimeofday` standard C library function, which provides a high accuracy wall clock time. To compensate for the variations, we list the best times out of three runs for each measurement.

In the comparison we concentrate on two properties of the methods, namely the amount of `fill`-in they introduce, and on the `total` time the solution takes to solve the system for one right-hand side. The fill-in is a major indicator for the method's memory usage and is a viable metric in this comparison, since all of the methods in some way rely on factorizations. The total solution time includes the computation of the matching, the fill-in reduction, the incomplete factorization, and the time for the iterative solution.

5.1. Iterative solution and termination criterion. We used the simplified symmetric QMR method (SQMR) as the standard Krylov-subspace solver (without look-ahead, as described in [17]). Since both the original system and the preconditioner are indefinite, we cannot use methods like SYMMLQ or MINRES, which require a positive definite preconditioner. We also considered GMRES(50) for the iterative solution. While GMRES provided slightly better convergence for some problems, it was generally slower and incurs a considerable memory overhead.

The iterative methods used the following . The current iterate $\hat{\boldsymbol{x}}_n$ was considered good enough if its residual referring to the preconditioned system $\hat{A}$ was less than 10^{-8}:

$$\|M^{-1}\boldsymbol{b} - M^{-1}\hat{A}\hat{\boldsymbol{x}}_n\| \leq 10^{-8}\|M^{-1}\boldsymbol{b}\|. \tag{5.1}$$

The maximum number of iterations was limited to 400. Failure of this limit is indicated by a dash ("–") in the tables that follow. Since we compare direct and iterative solvers in this paper, we also include another . The relative residuals of the results provided by the solvers were checked against the original system:

$$\|\boldsymbol{b} - A\boldsymbol{x_n}\| \leq 10^{-4}\|\boldsymbol{b}\|. \tag{5.2}$$

If they failed to be smaller than 10^{-4}, the systems were counted as incorrectly solved. In the tables, this is indicated by a dagger (†). If the residual turned out to be good enough although the iterative method did not stop before the maximum number of iterations, we counted the system as solved and indicate this by an asterisk (*).

All matrices were solved with an artificial right-hand side $\boldsymbol{b}$, with $\boldsymbol{b} = A \cdot \mathbf{1}$. The iterations were started with a zero initial guess.

5.2. Tested solution methods. In addition to results for the proposed approach, we provide comparisons with four other methods. To show the effects of the preprocessing and the pivoting, we compare the proposed method to a simple scalar incomplete $\mathrm{LDL}^{\mathrm{T}}$ factorization with a METIS reordering, and to the same factorization but with the sym2x2block reordering. Furthermore, for reference to other current methods, we provide comparisons with two additional preconditioned iterative methods and a direct solver.

We use the direct solver PARDISO [33, 34], which was shown to be competitive among current state-of-the-art direct solvers in a recent comparative study by Gould, Hu, and Scott [20]. Direct solvers are commonly used to solve general indefinite systems. In our tests we use the default setting of PARDISO, which uses supernodal Bunch–Kaufman pivoting and two steps of iterative refinement. This setting provides very fast results for symmetric indefinite systems [20]. However, some of the augmented sparse indefinite matrices used in this paper are very ill-conditioned and could not be solved using the default options.

A popular form of block preconditioners are the so-called [illegible] approaches, which are tailored for augmented systems of the form

$$\begin{pmatrix} H & B \\ B^T & 0 \end{pmatrix} \tag{5.3}$$

and need information about the dimension of the constraint block B in the system. The idea is to solve for the constraints B exactly and to use an approximation $\tilde{H}$ for block H. The simplest strategy to build M, which already has favorable properties, is to approximate H by its diagonal and to factorize M with a sparse direct solver (see, e.g., [16, 28, 30]), but various approaches are currently under active investigation (see [5, section 10.2] for a recent overview). We implemented the simple diagonal strategy and factorized M with PARDISO.

To provide comparisons with unsymmetric permutations, we include the ILUC-MPS method in our tests. See [6, 23] for earlier studies of this preconditioning method. The method is based on an incomplete LU factorization of the row-permuted and scaled matrix A,

$$M = LU = P_{\mathcal{M}} D_r A D_c - E. \tag{5.4}$$

For the tests, a Crout variant with no additional pivoting is used. The dropping criterion is the same as in the symmetric factorization. GMRES(50) is used in place of SQMR for the iterative solution, since the permuted systems are no longer symmetric.

Table 5.1 gives a short summary of the preconditioning methods and the abbreviations used. All factorization methods use the METIS nested dissection reordering by Karypis and Kumar [27].

5.3. Parameter selection. The only parameter for the incomplete factorizations is the fill-in bound γ. Unless otherwise noted, we used $\gamma = 6$, which provided good overall performance while keeping the fill-in moderate. Table 5.2 provides a comparison of the method for different values of γ for the first set of test problems.

5.4. Codes. All the preconditioned methods are implemented in the `sagg` solver library [22]. The incomplete factorizations are based on the same code base, in order to ensure meaningful comparisons. Specifically, the dropping criteria are implemented in exactly the same way in each implementation. The implementations of the iterative

TABLE 5.1
Preconditioning methods used for numerical tests, and their abbreviations.

ildlt	Incomplete LDL^T factorization
ildlt-s22m	$ILDL^T$ with sym2x2block preordering
ildlt-tp-s22m	$ILDL^T$ with sym2x2block and tridiagonal pivoting
iluc-mps	ILU variant with unsymmetric MPS reordering
pardiso	PARDISO sparse direct solver [33]
cp-direct	Constraint preconditioner with $\tilde{C} = \text{diag}(C)$

TABLE 5.2
Fill-in and number of iterations for different values of γ. Here `n` denotes the matrix dimension, `cs` denotes the number of constraint equations, and `nnz` the number of nonzeros in the matrix. The smallest fill-in factors are underlined.

Matrix				Solution with **ildlt-tp-s22b**					
				$\gamma = 1$		$\gamma = 3$		$\gamma = 6$	
Name	n	cs	nnz	fill	iters	fill	iters	fill	iters
capt12	2,443	1172	28,107	<u>1.91</u>	33	2.58	3	2.58	1
heat02	10,295	5098	90,129	2.00	–	<u>5.50</u>	80	8.84	34
lnts09	17,990	7996	109,778	<u>2.20</u>	13	4.13	4	4.14	1
mass04	8,496	46	56,864	<u>1.96</u>	12	4.68	5	6.67	3
mass05	33,410	128	241,140	<u>2.11</u>	42	4.69	30	6.88	20
mass06	33,794	512	257,732	<u>1.98</u>	83	4.61	39	6.86	17
plnt01	2,817	1362	52,891	<u>1.93</u>	9	2.49	1	2.49	1
stiff4	8,496	46	41,364	<u>1.91</u>	84	4.29	24	6.20	14
stiff5	33,410	128	177,384	<u>2.06</u>	239	4.20	80	6.12	40
traj27	17,148	7145	242,286	<u>2.03</u>	33	2.59	1	2.59	1
traj33	20,006	7145	504,090	<u>1.86</u>	6	1.96	1	1.96	1

solvers are either adapted from [4] or implemented from the original reference. The weighted matching code is based on [21] and performs comparably to the `mc64` code from the HSL library [25]. All codes were compiled using the GNU C compiler version 3.4.4, using `-O3` and processor specific optimizations.

5.5. Set 1: KKT systems. This set contains matrices that were used in recent papers by Keller, Gould, and Wathen [28] and Haws and Meyer [23]. It is comprised of a number of KKT systems from structural engineering and optimal control. As the names indicate, the `mass-` and `stiff-` matrices are mass and stiffness matrices of the form (5.3). The other matrices are from the Boeing sparse optimal control software SOCS [36]; see [23] for further details.

Looking at Table 5.3 it is of note that the proposed method is the only one which solved all of the problems. The `lnts` and `mass06` problems could not be solved with the default setting of PARDISO; hence the constraint preconditioner could not solve `lnts` either. It also only hardly solved the `stiff` problems, which have only a few constraint equations. With the same dropping threshold, the ILUC preconditioner used 50–90% of additional fill-in on average, but did not solve two systems.

Table 5.2 gives a comparison for smaller values of γ. With $\gamma = 6$, some of the incomplete factors contain more fill-in than the direct solver, because the fill-reducing reordering is not as good (see Figure 5.2 below). This is especially pronounced for the `traj*` and `lnts09` examples. These matrices contain very dense rows, but are otherwise very sparse. Thus the fill-in is increased, but few entries per column are dropped. It is remarkable, however, that the incomplete factorization is still faster than the direct solver in these cases.

TABLE 5.3
Comparison of various solvers for the KKT set. Smallest fill-in factors (within 5% of the smallest) are underlined; best times (within 5% of the fastest) are bolded; see the text for additional explanation.

Matrix	**ildlt-tp-s22b**			**iluc-mps**			**cp-direct**			**pardiso**	
Name	fill	Total	iters	fill	Total	iters	fill	Total	iters	fill	Total
capt12	2.58	**0.05**	1	5.34	0.09	2	2.05	0.12	5	2.15	0.12
heat02	8.84	1.24	34	10.37	0.88	25	7.38	0.63	4	8.01	**0.43**
lnts09	4.14	**0.34**	1	8.46	0.58	4	3.25	–	–	3.27	0.40†
mass04	6.67	**0.30**	3	9.70	**0.29**	5	0.38	0.44	9	10.88	**0.31**
mass05	6.88	2.41	20	9.68	–	–	0.42	4.34	9	14.32	**1.91**
mass06	6.86	**2.52**	17	9.51	1.72	6†	0.57	3.93	11	15.76	2.19†
plnt01	2.49	**0.10**	1	5.04	0.16	1	1.85	0.21	11	2.09	**0.09**
stiff4	6.20	0.33	14	9.09	0.66	47	0.56	4.74	271	11.60	**0.25**
stiff5	6.12	2.73	40	8.79	13.87	168	0.60	32.19	344	14.59	**1.52**
traj27	2.59	**0.54**	1	5.42	0.91	1	1.80	1.24	7	2.13	0.69
traj33	1.96	**0.97**	1	3.89	1.68	1	1.04	1.53	6	1.51	1.24
Total	**11** / 11			9 / 11			10 / 11			9 / 11	

TABLE 5.4
Comparison of various solvers for a subset of the IPOPT matrices. Meaning of bold and underlines is as in Table 5.3*. Double daggers (‡) indicate failure due to floating point exceptions.*

Matrix		**ildlt-tp-s22b**			**iluc-mps**			**pardiso**	
Name	n	fill	Total	iters	fill	Total	iters	fill	Total
c-18	2,169	2.98	**0.05**	5	6.71	0.07	5	3.29	**0.05**
c-35	6,537	4.95	0.28	4	7.02	0.37	5	4.71	**0.25**
c-41	9,769	3.77	0.69	16	7.66	8.91	–	3.90	**0.45**
c-45	13,206	1.45	0.54	1	3.39	0.90	13	1.62	**0.38**
c-54	31,793	3.34	13.25	192	‡	‡	‡	3.13	**1.97**
c-60	43,640	3.29	**2.36**	15	‡	‡	‡	8.03	**2.44**
c-65	48,066	3.46	2.56	11	‡	‡	‡	4.75	**2.26**
c-70	68,924	3.61	**5.67**	15	‡	‡	‡	11.19	8.94
c-71	76,638	4.41	**7.80**	12	‡	‡	‡	35.17	65.68
c-72	84,064	3.48	**5.87**	13	‡	‡	‡	9.55	7.80
c-73	169,422	1.55	11.98	12	4.08	29.76	50	2.73	**6.46**
c-big	345,241	3.69	**28.05**	12	‡	‡	‡	39.12	289.82
Total solved		**57** / 57			26 / 57			**57** / 57	

5.6. Set 2: Matrices from interior point optimization. As a case study, we examine a series of interior point optimization matrices[2] from IPOPT [37]. The core step of the optimization algorithm is a damped Newton iteration, where in each step a modified symmetrized linear system of the form

$$\begin{pmatrix} W + \Sigma + \delta_w I & B \\ B^T & -\delta_c I \end{pmatrix} \tag{5.5}$$

is solved. In our examples, the upper left block of (5.5) is diagonal and $\delta_c = 10^{-8}$. The whole set consists of 57 matrices of increasing size. We show a subset of 12 matrices, which was selected to include the hard-to-solve examples (specifically `c-54`) as well as to give an impression of the scalability of the methods. Therefore matrices from the whole range were picked, with a bias toward the large ones. See Table 5.4 for a

[2]The matrices are available at http://www.computational.unibas.ch/cs/scicomp/matrices.

TABLE 5.5
Details of the matrices and the solution steps of ILDL$^{\mathrm{T}}$-TP-S22B *for the IPOPT subset.* mps, metis, fact, *and* solve *are times in seconds for the weighted matching, the reordering, the incomplete factorization, and the iterative solution, respectively.* 2x2 *denotes the number of* 2×2 *pivots chosen during the factorization.*

Matrix				**ildlt-tp-s22b**					
Name	n	cs	nnz	2x2	mps	metis	fact	solve	Total
c-18	2,169	901	15,145	1	0.01	0.02	0.01	0.01	0.05
c-35	6,537	2423	62,891	1231	0.03	0.09	0.10	0.04	0.28
c-41	9,769	4320	101,745	238	0.13	0.17	0.15	0.21	0.69
c-45	13,206	6314	174,452	1	0.28	0.08	0.10	0.02	0.54
c-54	31,793	14129	391,693	251	2.48	0.91	0.62	9.07	13.25
c-60	43,640	19221	298,578	28	0.24	0.77	0.43	0.74	2.36
c-65	48,066	20408	360,528	16	0.30	0.94	0.48	0.66	2.56
c-70	68,924	29622	658,986	6	0.70	1.90	1.22	1.53	5.67
c-71	76,638	31824	859,554	40	0.72	2.26	2.69	1.70	7.80
c-72	84,064	36114	707,558	2022	0.69	2.18	1.12	1.50	5.87
c-73	169,422	83005	1,279,274	6	5.75	2.19	1.00	2.21	11.98
c-big	345,241	143364	2,341,011	95	5.56	11.60	4.09	5.31	28.05

summary of the results, and Table 5.5 for more detailed timings and statistics of the individual solver components..

Surprisingly, the ILDL$^{\mathrm{T}}$-TP-S22B method scales almost perfectly with the dimension of the matrices. This is due to the fact that the fill-in is always bounded to a factor of at most five. The only significant outliers in terms of iterations are the matrices `c-41`, `c-54`, and `c-56`, whose B^T blocks do not have full rank. Because of these rank deficiencies, small diagonal entries from the (2,2) block are matched, which gives rise to small scalar pivots during the factorization. We are currently investigating ways to handle these rank deficiencies more effectively.

The ILUC-MPS method is obviously not well suited for these problems and solved less than half of the test problems when the fill-in was strictly bounded. In many cases the factorization failed because of floating-point exceptions caused by exceeding growth in the factors. Results for the CP-DIRECT method are not shown, because the upper left block of the matrix is already diagonal, and the results are thus similar to those for PARDISO.

5.7. Set 3: Selection of problems from HSL collection. The last set of test problems is taken from an extensive survey of direct solvers for indefinite problems[3] [20]. This set is rather challenging, and several of the matrices could not be solved by direct solvers under the constraints imposed in the report. The matrices come from a variety of application areas. From the 61 original matrices, we excluded 10 `c-*` matrices which are in the separate IPOPT set. Of the 51 remaining matrices, 34 are KKT matrices. We report the results separately for both parts. The tables are slightly abridged, but with examples of all failures included.

A selection of results for the KKT subset is given in Tables 5.6 and 5.7. The augmented systems `AUG*` and the nonconvex quadratic problems `NCVX*` are singular or nearly singular, so the significance of the residual norm is at least questionable. Most direct solvers tested in [20] were unable to solve these systems either. The other failure, `olesnik0`, is highly indefinite and also caused problems for the direct solvers. Between the two preconditioners there is no clear winner for this set. The incomplete

[3]The matrices can be found at ftp://ftp.numerical.rl.ac.uk/pub/matrices/symmetric/indef.

TABLE 5.6
Short description of our subset of the KKT matrices from the HSL set.

Matrix	n	cs	nnz	Application domain
A0NSDSIL	80,016	20,004	390,042	Linear complementarity
A2NNSNSL	80,016	35,008	196,115	Linear complementarity
A5ESINDL	60,008	25,004	145,004	Linear complementarity
AUG2D	29,008	9,604	38,416	Expanded system—2D PDE
AUG3D	24,300	5,832	94,284	Expanded 2D/3D problems
BLOCKQP1	60,012	20,001	660,034	QP with block structure
BOYD1	93,279	18	1,211,249	Convex QP
BOYD2	466,316	186,531	1,780,186	Convex QP
CONT-300	180,895	90,298	1,078,493	Convex QP
DTOC	24,993	9,997	94,965	Discrete-time optimal control
K1_SAN	67,759	20,805	580,579	Mine model
NCVXQP5	62,500	12,500	437,466	Nonconvex QP
NCVXQP7	87,500	37,500	612,462	Nonconvex QP
NCVXQP9	16,554	7,500	61,540	Nonconvex QP
SIT100	10,262	3,120	64,166	Mine model
mario001	38,434	15,304	221,460	Stokes equation
stokes64	12,546	4,096	144,130	Stokes equation
stokes64s	12,546	4,096	144,130	Stokes equation
olesnik0	88,263	27,233	771,449	Mine model
tuma1	22,967	9,607	66,592	Mine model
tuma2	12,992	5,477	72,878	Mine model

TABLE 5.7
Results for KKT subset from the HSL matrices. Notation as previously.

Matrix	**ildlt-tp-s22b**			**cp-direct**			**pardiso**	
Name	fill	Total	iters	fill	Total	iters	fill	Total
A0NSDSIL	2.38	**1.37**	1	1.77	–	–	2.29	2.85
A2NNSNSL	2.38	**1.34**	1	1.74	–	–	2.17	2.65
A5ESINDL	2.39	**0.97**	1	1.45	–	–	2.14	1.26
AUG2D	6.36	–	– *	7.96	1.90	9	7.96	**0.81**
AUG3D	5.04	–	–	19.45	2.61	12	19.45	**1.25**
BLOCKQP1	1.68	**2.55**	1	1.20	3.26	2	2.60	2.93
BOYD1	1.00	**28.41**	1	1.17	32.24	1	1.17	31.75
BOYD2	1.39	**32.58**	2	2.07	43.00	1	2.07	40.65
CONT-300	7.66	55.47	205	22.65	17.93	2	22.65	**15.17**
DTOC	2.71	17.53	7	3.10	0.55	1	3.10	**0.42**
NCVXQP5	8.38	–	–	0.60	–	–	58.08	**31.79**
NCVXQP7	9.51	–	–	3.39	–	– *	70.29	**120.44**
NCVXQP9	4.72	3.99	234	6.46	0.56	2	6.46	**0.42**
SIT100	7.87	0.57	16	6.81	–	– *	14.16	**0.44**
mario001	6.14	2.11	22	3.65	3.03	12	4.65	**1.28**
olesnik0	7.71	–	–	6.13	28.59	50	11.93	**6.07**
stokes64	8.67	2.75	43	2.47	**0.43**	1	8.33	0.68
stokes64s	8.69	3.48	57	2.47	**0.45**	1	8.33	0.67
tuma1	5.66	1.18	24	3.88	2.03	18	7.50	**0.73**
tuma2	5.47	0.53	18	3.22	0.99	18	6.27	**0.35**
Total solved	27 / 34			29 / 34			**34** / 34	

factorization preconditioner seems to have advantages for the linear complementarity and the quadratic programming problems, while the constraint preconditioner works well for most of the mine models and the `stokes*` problems.

From the general indefinite matrices, only few could be solved by the iterative method (see Table 5.8). Among these were a Dixon–Maany problem, a matrix inverse

TABLE 5.8
Results for some of the general indefinite HSL matrices. Notation as previously.

Matrix			**ildlt-tp-s22b**			**pardiso**	
Name	n	nnz	fill	Total	iters	fill	Total
DIXMAANL	60,000	299,998	2.69	**1.36**	1	3.23	1.46
HELM2D03	392,257	2,741,935	7.28	–	–	17.14	**29.72**
LINVERSE	11,999	95,977	2.06	**0.22**	1	2.41	**0.23**
NCVXBQP1	50,000	349,968	6.00	–	–	14.50	**3.21**
SPARSINE	50,000	1,548,988	10.91	–	–	271.15	**4995.23**
bcsstk35	30,237	1,450,163	4.24	11.71	16	4.76	**2.85**
bcsstk39	46,772	2,089,294	6.03	25.29	14	7.97	**7.49**
bmw3_2	227,362	11,288,630	5.46	–	–	8.75	**73.94**
copter2	55,476	759,952	9.20	–	–	28.05	**16.85**
crystk02	13,965	968,583	7.48	19.35	1 †	9.51	5.56 †
crystk03	24,696	1,751,178	7.78	38.60	1 †	11.60	13.93 †
dawson5	51,537	1,010,777	7.99	–	–	8.98	**5.45**
qa8fk	66,127	1,660,579	8.87	43.85	80 †	27.92	42.64 †
vibrobox	12,328	342,828	6.51	6.43	70	15.88	**3.92**
Total solved			7 / 17			14 / 17	

TABLE 5.9
Comparison of plain incomplete LDL^T, ildlt *with* sym2x2block *preprocessing, and* ildlt *with additional tridiagonal pivoting* (ildlt-tp-s22b). *All variants use the same dropping strategy.*

Test set	# solved by **ildlt**	# solved by **ildlt-s22b**	# solved by **ildlt-tp-s22b**
Set 1, KKT	8 / 11	10 / 11	**11** / 11
Set 2, IPOPT	53 / 57	56 / 57	**57** / 57
Set 3, HSL	21 / 51	33 / 51	34 / 51
Total	82 / 119	99 / 119	102 / 119

approximation problem, a vibro-acoustic model, and the `bcsstk*` stiffness matrices. The last are diagonally dominant, so the matching had no influence on the solution. Many of these matrices were also challenging for the direct solvers [20].

5.8. Effects of preprocessing and pivoting. Table 5.9 illustrates the effects of the preprocessing step and the additional 2×2 pivoting in relation to a simple scalar incomplete LDL^T factorization. Both methods combined increase the number of solved problems by about 24%. The major part of this margin is due to the sym2x2block preprocessing. The 2×2 pivoting increases the stability by a small but critical margin. With the pivoting, all systems from the first two sets were reliably solved in less than 200 iterations. Without it, not all problems could be solved, and five problems took more than 300 iterations to converge.

5.9. Costs of preprocessing. There are two kind of costs involved in the preprocessing step. One is the time required to find a maximum weight matching for the matrix. This cost is, however, partly neutralized by the fact that the subsequent fill-reduction reordering is run on a compressed graph and is therefore of reduced complexity. Figure 5.1 relates the combined times of the matching and the fill-reducing reordering to a run of the fill-reducing reordering alone. Often the preprocessing step is even faster than METIS alone.

The other cost incurred by the preprocessing step is an increase of fill-in. This is due to the fact that the fill-reduction method works on a compressed matrix, and has

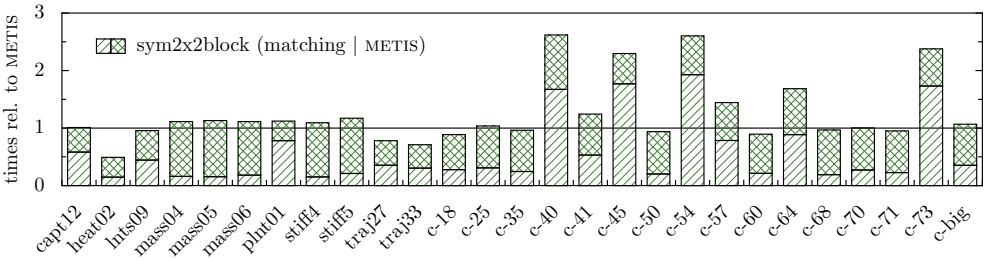


FIG. 5.1. *Times for the preprocessing step (matching + METIS) relative to METIS alone.*

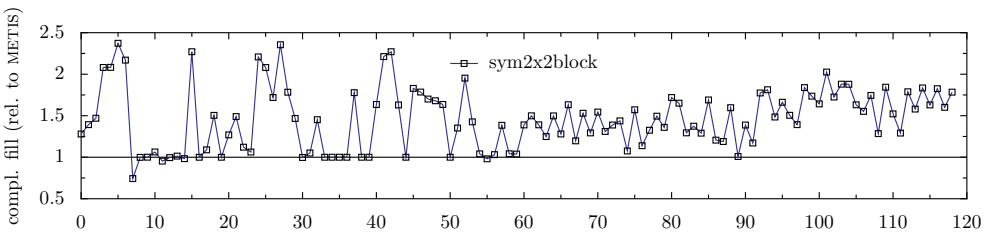


FIG. 5.2. *Fill-in for a complete symbolic factorization after the preprocessing step relative to fill-in for a factorization after METIS, for all matrices.*

thus a less accurate picture of the actual associated graph. Furthermore, the blocking often introduces additional fill-in on its own. Figure 5.2 shows the increase of fill-in for a complete symbolic factorization after the sym2x2block reordering, compared to the fill-in after a METIS reordering. The mean increase is 46%. This also illustrates why the fill-in of the incomplete factorizations can be higher than the fill-in for the direct factorization in some cases.

6. Conclusions and future work. We present a combination of a symmetric permutation and an incomplete LDL^T factorization that makes up a purely algebraic preconditioner for symmetric indefinite linear systems. It is particularly effective for sparse saddle-point problems, where the sym2x2block reordering is capable of providing a feasible pivoting order for the factorization while the cost of this preprocessing step remains relatively small. The incomplete factorization with tridiagonal pivoting is very fast, since it can be performed without pivot searching and possibly costly dynamic reorderings. By controlling the fill-in, it is possible to adjust the convergence and the memory demand over a wide range of values.

By developing a competitive implementation, we show that the relatively simple approach can compete with a fast direct solver both in terms of stability and performance. In comparison to the other preconditioners we tested, only the constraint preconditioners yielded similar results. In the simple form used here, however, our method strongly depends on the performance of the solution method for M.

The method is not well suited for many general indefinite problems like Helmholtz problems, which are highly indefinite but still have a strong diagonal. In these cases the matching yields the identity permutation, and the method falls back to a simple incomplete factorization.

Our results warrant further research into specialized, symmetric matching algorithms. Although only marginal benefits in terms of speedup are to be expected, optimal symmetric maximum weight matchings may further enhance the stability of

the method. We also experimented with approximate methods, with limited success thus far. This is, however, also a promising area of research.

Furthermore we plan to leverage the structural information that the matchings provide. In particular, information about structurally rank deficient blocks could be used to avoid numerical difficulties and might also be useful for higher level solvers like nonlinear optimization packages.

Acknowledgments. The authors thank Stefan Röllin for providing the original maximum matching implementation, Jennifer Scott for providing the HSL matrices, Andreas Wächter for providing the `c-*` matrices, and John Haws for providing the matrices from the first set. We would also like to thank the three referees for providing constructive criticism which improved both presentation and content of the paper.

REFERENCES

[1] C. Ashcraft, R. G. Grimes, and J. G. Lewis, *Accurate symmetric indefinite linear equation solvers*, SIAM J. Matrix Anal. Appl., 20 (1998), pp. 513–561.

[2] O. Axelsson and M. Neytcheva, *Preconditioning methods for linear systems arising in constrained optimization problems*, Numer. Linear Algebra Appl., 10 (2003), pp. 3–31.

[3] R. E. Bank and R. K. Smith, *The incomplete factorization multigraph algorithm*, SIAM J. Sci. Comput., 20 (1999), pp. 1349–1364.

[4] R. Barrett, M. W. Berry, T. F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. van der Vorst, *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*, SIAM, Philadelphia, 1993.

[5] M. Benzi, G. H. Golub, and J. Liesen, *Numerical solution of saddle point problems*, Acta Numer., 14 (2005), pp. 1–137.

[6] M. Benzi, J. C. Haws, and M. Tůma, *Preconditioning highly indefinite and nonsymmetric matrices*, SIAM J. Sci. Comput., 22 (2000), pp. 1333–1353.

[7] J. R. Bunch, *Partial pivoting strategies for symmetric matrices*, SIAM J. Numer. Anal., 11 (1974), pp. 521–528.

[8] J. R. Bunch and L. Kaufman, *Some stable methods for calculating inertia and solving symmetric linear systems*, Math. Comp., 31 (1977), pp. 163–179.

[9] E. Chow and Y. Saad, *Experimental study of ILU preconditioners for indefinite matrices*, J. Comput. Appl. Math., 86 (1997), pp. 387–414.

[10] I. S. Duff and J. R. Gilbert, *Maximum-weighted matching and block pivoting for symmetric indefinite systems*, in Abstract Book of Householder Symposium XV, 2002, pp. 73–75.

[11] I. S. Duff and J. Koster, *The design and use of algorithms for permuting large entries to the diagonal of sparse matrices*, SIAM J. Matrix Anal. Appl., 20 (1999), pp. 889–901.

[12] I. S. Duff and J. Koster, *On algorithms for permuting large entries to the diagonal of a sparse matrix*, SIAM J. Matrix Anal. Appl., 22 (2001), pp. 973–996.

[13] I. S. Duff and S. Pralet, *Strategies for scaling and pivoting for sparse symmetric indefinite problems*, SIAM J. Matrix Anal. Appl., 27 (2005), pp. 313–340.

[14] I. S. Duff and J. K. Reid, *The multifrontal solution of indefinite sparse symmetric linear equations*, ACM Trans. Math. Softw., 9 (1983), pp. 302–325.

[15] S. C. Eisenstat, M. H. Schultz, and A. H. Sherman, *Algorithms and data structures for sparse symmetric Gaussian elimination*, SIAM J. Sci. Comput., 2 (1981), pp. 225–237.

[16] R. E. Ewing, R. D. Lazarov, P. Lu, and P. S. Vassilevski, *Preconditioning indefinite systems arising from mixed finite element discretization of second-order elliptic problems*, in Preconditioned Conjugate Gradient Methods (Nijmegen, 1989), Lecture Notes in Math. 1457, Springer, Berlin, 1989, pp. 28–43.

[17] R. W. Freund, *Preconditioning of symmetric, but highly indefinite linear systems*, in Proceedings of the 15th IMACS World Congress on Scientific Computation, Modeling and Applied Mathematics, Vol. 2, Numerical Mathematics, Wissenschaft und Technik Verlag, 1997, pp. 551–556.

[18] R. W. Freund and F. Jarre, *A QMR-based interior-point algorithm for solving linear programs*, Math. Programming, Series B, 76 (1996), pp. 183–210.

[19] P. E. Gill, W. Murray, D. B. Ponceleón, and M. A. Saunders, *Preconditioners for indefinite systems arising in optimization*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 292–311.

[20] N. I. M. Gould, Y. Hu, and J. A. Scott, *A Numerical Evaluation of Sparse Direct Solvers for the Solution of Large Sparse, Symmetric Linear Systems of Equations*, Technical Report RAL-TR-2005-005, Rutherford Appleton Laboratory, Oxfordshire, UK, 2005.

[21] A. Gupta and L. Ying, *On Algorithms for Finding Maximum Matchings in Bipartite Graphs*, Technical Report RC 21576 (97320), IBM T. J. Watson Research Center, Yorktown Heights, NY, 1999.

[22] M. Hagemann and O. Schenk, *Sagg—Sparse à Go-Go, A Solver Package for Sparse Matrices*, University of Basel, 2004; see online resources at http://www.computational.unibas.ch/cs/scicomp/software/sagg/.

[23] J. C. Haws and C. D. Meyer, *Preconditioning KKT Systems*, Technical Report M&CT-Tech-01-021, The Boeing Company, Chicago, IL, 2003.

[24] N. J. Higham, *Stability of block* $\mathrm{LDL}^{\mathrm{T}}$ *factorization of a symmetric tridiagonal matrix*, Linear Algebra Appl., 287 (1999), pp. 181–189.

[25] *HSL Subroutine Library, A collection of Fortran codes for large-scale scientific computation*, Rutherford Appleton Laboratory, Oxfordshire, UK, 2004; http://www.cse.clrc.ac.uk/nag/hsl/hsl.shtml.

[26] M. T. Jones and P. E. Plassmann, *An improved incomplete Cholesky factorization*, ACM Trans. Math. Softw., 21 (1995), pp. 5–17.

[27] G. Karypis and V. Kumar, *A fast and high quality multilevel scheme for partitioning irregular graphs*, SIAM J. Sci. Comput., 20 (1998), pp. 359–392.

[28] C. Keller, N. I. M. Gould, and A. J. Wathen, *Constraint preconditioning for indefinite linear systems*, SIAM J. Matrix Anal. Appl., 21 (2000), pp. 1300–1317.

[29] N. Li and Y. Saad, *Crout versions of ILU factorization with pivoting for sparse symmetric matrices*, Elec. Trans. Numer. Anal., 20 (2005), pp. 75–85.

[30] L. Lukšan and J. Vlček, *Indefinitely preconditioned inexact Newton method for large sparse equality constrained non-linear programming problems*, Numer. Linear Algebra Appl., 5 (1998), pp. 219–247.

[31] M. Olschowka and A. Neumaier, *A new pivoting strategy for Gaussian elimination*, Linear Algebra Appl., 240 (1996), pp. 131–151.

[32] Y. Saad, *ILUT: A dual threshold incomplete LU factorization*, Numer. Linear Algebra Appl., 1 (1994), pp. 387–402.

[33] O. Schenk and K. Gärtner, *On Fast Factorization Pivoting Methods for Symmetric Indefinite Systems*, Technical report CS-2004-004, Computer Science Department, University of Basel, Switzerland, 2004; Electronic Transactions on Numerical Analysis, submitted.

[34] O. Schenk, K. Gärtner, and W. Fichtner, *Efficient sparse LU factorization with left-right looking strategy on shared memory multiprocessors*, BIT, 40 (2000), pp. 158–176.

[35] O. Schenk, S. Röllin, and A. Gupta, *The effects of unsymmetric matrix permutations and scalings in semiconductor device and circuit simulation*, IEEE Trans. Computer-Aided Design Integrated Circuits Systems, 23 (2004), pp. 400–411.

[36] *SOCS Sparse Optimal Control Software*, Boeing Phantom Works, The Boeing Company, Chicago, IL, 2002.

[37] A. Wächter and L. T. Biegler, *On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming*, Math. Prog., 106 (2006), pp. 25–57.

[38] W. Zulehner, *Analysis of iterative methods for saddle point problems: A unified approach*, Math. Comput., 71 (2002), pp. 479–505.

SIAM J. SCI. COMPUT.
Vol. 28, No. 2, pp. 421–438
© 2006 Society for Industrial and Applied Mathematics

FAST AND OBLIVIOUS CONVOLUTION QUADRATURE*

ACHIM SCHÄDLE†, MARÍA LÓPEZ-FERNÁNDEZ‡, AND CHRISTIAN LUBICH§

Abstract. We give an algorithm to compute N steps of a convolution quadrature approximation to a continuous temporal convolution using only $O(N \log N)$ multiplications and $O(\log N)$ active memory. The method does not require evaluations of the convolution kernel but instead uses $O(\log N)$ evaluations of its Laplace transform, which is assumed sectorial. The algorithm can be used for the stable numerical solution with quasi-optimal complexity of linear and nonlinear integral and integro-differential equations of convolution type. In a numerical example we apply it to solve a subdiffusion equation with transparent boundary conditions.

Key words. convolution, numerical integration, Runge–Kutta methods, Volterra integral equation, anomalous diffusion

AMS subject classification. 65R20

DOI. 10.1137/050623139

1. Introduction. In this paper we give a fast and memory-saving algorithm for computing the approximation of a continuous convolution (possibly matrix × vector)

$$\int_0^t f(t-\tau)\, g(\tau)\, d\tau\ , \qquad 0 \le t \le T, \tag{1.1}$$

by a convolution quadrature with a step size $h > 0$,

$$\sum_{j=0}^{n} \omega_{n-j}\, g(jh)\ , \qquad n = 1, \dots, N. \tag{1.2}$$

Here, the convolution quadrature weights ω_n are the coefficients of the generating power series (see [12])

$$\sum_{n=0}^{\infty} \omega_n \zeta^n = F\Big(\frac{\delta(\zeta)}{h}\Big), \tag{1.3}$$

where $F(s)$ is the Laplace transform of the (possibly matrix-valued) convolution kernel $f(t)$, and $\delta(\zeta) = 1 - \zeta$ or $\delta(\zeta) = (1-\zeta) + \frac{1}{2}(1-\zeta)^2$ for the methods based on the first- or second-order backward difference formula, respectively. We will also consider closely related convolution quadrature formulas that are based on implicit Runge–Kutta methods, such as the Radau IIA schemes [13].

Convolution quadratures (1.2)–(1.3) were introduced in [10, 11] and have since been used in a variety of applications. A recent review is given in [12], which contains an extensive, annotated bibliography. Attractive features of these convolution quadratures are that they work well for singular kernels $f(t)$, for kernels with multiple time

*Received by the editors January 24, 2005; accepted for publication (in revised form) October 28, 2005; published electronically March 31, 2006.
http://www.siam.org/journals/sisc/28-2/62313.html

†ZIB Berlin, Takustr. 7, D-14195 Berlin, Germany (schaedle@zib.de). This author's research was supported by the DFG Research Center Matheon, Berlin.

‡Departamento de Matemática Aplicada, Universidad de Valladolid, Valladolid, Spain (marial@mac.cie.uva.es). This author's research was supported by DGI-MCYT under project MTM 2004-07194 cofinanced by FEDER funds.

§Mathematisches Institut, Universität Tübingen, Auf der Morgenstelle 10, D–72076 Tübingen, Germany. (lubich@na.uni-tuebingen.de). This author's research was supported by DFG, SFB 382.

scales, and in the common situation where the Laplace transform $F(s)$ (the transfer function) but not the convolution kernel $f(t)$ (the impulse response) is known analytically. Perhaps most important, these convolution quadratures enjoy excellent stability properties in the discretization of integral equations or integrodifferential equations of convolution type, in a way often strikingly opposed to discretizations with more straightforward quadrature formulas. This comes about because the stability of a discrete convolution equation depends on the range of the generating function of the weights for $|\zeta| < 1$. In (1.3), the generating function is directly related to the Laplace transform $F(s)$ which determines the stability properties of the continuous convolution equation.

The direct way to compute (1.2) is to first compute and store the (possibly matrix-valued) weights $\omega_0, \dots, \omega_N$, which can be done accurately with $O(N)$ evaluations of the Laplace transform $F(s)$ [11], and then to compute the discrete convolution. Done naively, this requires $O(N^2)$ multiplications (possibly matrix $\times$ vector) and $O(N)$ active memory for the values $g(jh)$ and for the weights. Using FFT, the number of multiplications can be reduced to $O(N \log N)$ and to $O(N (\log N)^2)$ in the case of integral equations where the values of $g(t)$ are not known beforehand, but where $g(nh)$ is computed only in the nth time step [3]. However, that approach does not reduce the number of F-evaluations and the memory requirements.

Here we give an algorithm, also applicable in the case of linear and nonlinear integral equations, which computes (1.2) in a way that requires

- $O(N \log N)$ multiplications,
- $O(\log N)$ evaluations of the Laplace transform $F(s)$, and
- $O(\log N)$ active memory.

The history $g(jh)$ for $j = 0, \dots, N$ is forgotten in this algorithm, and only logarithmically few linear combinations of the g-values are kept in memory. These are obtained by solving numerically, with step size h, initial value problems of the form $y' = \lambda y + g$ with complex λ. The weights ω_n ($n \le N$) are not computed explicitly, except the first few, e.g., the first 10 weights.

The algorithm presented here takes up ideas of the fast convolution algorithm of [14], which instead of (1.2)–(1.3) makes a different approximation to the continuous convolution. The stability properties of the second-order method of [14] in integro-partial differential equations such as those of section 5 are, however, extremely difficult to analyze (cf. also [17]) and remain entirely unclear for higher-order extensions, whereas the stability of convolution quadratures (1.2)–(1.3) for such problems is well understood; see [1]. Here we show how the convolution quadratures (1.2)–(1.3) with all their known favorable properties can be implemented in an equally fast and memory-saving way as the method in [14].

Following the error analysis of [7, 8] we give exponentially convergent error bounds for the contour integral approximations that are employed in this algorithm. They ensure that the constants hidden in the O-symbols of the above work estimates depend only logarithmically on the error tolerance for these contour integral approximations.

We assume a [illegible] Laplace transform $F(s)$:

(1.4)

$F(s)$ is analytic in a sector $|\arg(s - \sigma)| < \pi - \varphi$ with $\varphi < \frac{1}{2}\pi$, and in this sector

$$|F(s)| \le M\,|s|^{-\nu} \quad \text{for some real } M \text{ and } \nu > 0.$$

The inverse Laplace transform is then given by

$$f(t) = \frac{1}{2\pi i} \int_\Gamma e^{t\lambda} F(\lambda)\, d\lambda, \qquad t > 0, \tag{1.5}$$

with Γ a contour in the sector of analyticity, going to infinity with an acute angle to the negative real half-axis and oriented with increasing imaginary part. The function $f(t)$ is analytic in $t > 0$ and satisfies

$$|f(t)| \le C\, t^{\nu-1}\, e^{ct}, \qquad t > 0, \tag{1.6}$$

and is therefore locally integrable. (The absolute values on the left-hand sides of the bounds (1.4) and (1.6) are to be interpreted as matrix norms for matrix-valued convolution kernels.) A prototypical example, which will actually be used in our numerical experiments, is the fractional-power kernel $f(t) = t^{\nu-1}/\Gamma(\nu)$ with $\nu > 0$, which has the Laplace transform $F(s) = s^{-\nu}$.

In section 2 we review convolution quadrature based on multistep and Runge–Kutta methods. We give a contour integral representation of the convolution quadrature weights whose discretization along hyperbolas or Talbot contours is discussed in section 3. The fast and oblivious convolution algorithm is formulated in section 4. Finally, in section 5 we give the results of numerical experiments with integral and integrodifferential equations originating from regular and anomalous diffusion problems.

2. Convolution quadrature. In this section we review briefly convolution quadrature and give a contour integral representation of the convolution quadrature weights on which the fast algorithm of this paper is based.

2.1. Convolution quadrature based on multistep methods. We consider the convolution quadrature (1.2) with weights (1.3). By (1.4) and Cauchy's integral formula we have, with a contour Γ as in (1.5),

$$\sum_{n=0}^{\infty} \omega_n \zeta^n = F\Big(\frac{\delta(\zeta)}{h}\Big) = \frac{1}{2\pi i} \int_\Gamma \Big(\frac{\delta(\zeta)}{h} - \lambda\Big)^{-1} F(\lambda)\, d\lambda.$$

Hence, with $e_n(z)$ defined by

$$(\delta(\zeta) - z)^{-1} = \sum_{n=0}^{\infty} e_n(z)\, \zeta^n, \tag{2.1}$$

we have the integral formula

$$\omega_n = \frac{h}{2\pi i} \int_\Gamma e_n(h\lambda)\, F(\lambda)\, d\lambda, \tag{2.2}$$

which can be viewed as the discrete analogue of (1.5). For the backward Euler discretization $\delta(\zeta) = 1 - \zeta$ we note the explicit formula

$$e_n(z) = (1 - z)^{-n-1}, \tag{2.3}$$

which is of the form $e_n(z) = q(z)r(z)^n$ with $r(z) = \frac{1}{1-z}$ and $q(z) = \frac{1}{1-z}$.

For the second-order backward differentiation formula (BDF), where $\delta(\zeta) = \sum_{k=1}^{p} \frac{1}{k}(1-\zeta)^k$ with $p = 2$, we obtain from a partial fraction decomposition of $(\delta(\zeta) - z)^{-1}$ that

$$e_n(z) = \frac{1}{\sqrt{1+2z}} \Big((2 - \sqrt{1+2z})^{-n-1} - (2 + \sqrt{1+2z})^{-n-1} \Big), \tag{2.4}$$

which is of the form $e_n(z) = q_1(z)r_1(z)^n + q_2(z)r_2(z)^n$. Connoisseurs of Cardano's formulas find analogous formulas to (2.4) also for the BDF methods of orders 3 and 4.

2.2. Convolution quadrature based on Runge–Kutta methods. We consider an implicit Runge–Kutta method with coefficients a_{ij}, b_j, c_i for $i, j = 1, \dots, m$. We denote the Runge–Kutta matrix by $\mathcal{A} = (a_{ij})$, the row vector of the weights by $b^T = (b_j)$, and the stability function by

$$r(z) = 1 + zb^T(I - z\mathcal{A})^{-1}\mathbb{1},$$

where $\mathbb{1} = (1, \dots, 1)^T$. We assume that all eigenvalues of the Runge–Kutta matrix $\mathcal{A}$ have positive real part and that the method is A-stable and the row vector of the weights equals the last line of the Runge–Kutta matrix,

$$b_j = a_{mj} \quad \text{for} \quad j = 1, \dots, m,$$

and correspondingly $c_m = 1$. Notice that this implies that the method is L-stable, i.e.,

$$|r(z)| \le 1 \text{ for } \mathrm{Re} z \le 0 \qquad \text{and} \quad r(\infty) = 0.$$

These conditions are in particular satisfied by the Radau IIA family of Runge–Kutta methods [4]. From such a Runge–Kutta method, a convolution quadrature is constructed as follows [13]: let

$$\Delta(\zeta) = \Big(\mathcal{A} + \frac{\zeta}{1-\zeta}\mathbb{1}b^T\Big)^{-1} \tag{2.5}$$

and define weight matrices W_n by

$$\sum_{n=0}^{\infty} W_n \zeta^n = F\Big(\frac{\Delta(\zeta)}{h}\Big). \tag{2.6}$$

Let $\omega_n = (\omega_n^1, \dots, \omega_n^m)$ denote the last row of W_n. Then an approximation to the convolution integral (1.1) at time $t_{n+1} = (n+1)h$ is given by

$$u_{n+1} = \sum_{j=0}^{n} \sum_{i=1}^{m} \omega_{n-j}^i \, g(t_j + c_i h) = \sum_{j=0}^{n} \omega_{n-j} \, g_j \tag{2.7}$$

with the column vector $g_j = \big(g(t_j + c_i h)\big)_{i=1}^m$. For a Runge–Kutta method of classical order p and stage order q, this approximation is known to be convergent of the order $\min(p, q+1+\nu)$ with ν of (1.4) [13, Theorem 2.2].

With the row vector $e_n(z) = (e_n^1(z), \dots, e_n^m(z))$ defined as the last row of the $m \times m$ matrix $E_n(z)$ given by

$$(\Delta(\zeta) - zI_m)^{-1} = \sum_{n=0}^{\infty} E_n(z) \, \zeta^n, \tag{2.8}$$

we obtain an integral formula like in (2.2),

$$\omega_n = \frac{h}{2\pi i} \int_\Gamma e_n(h\lambda) \otimes F(\lambda) \, d\lambda. \tag{2.9}$$

For $n \geq 0$, $e_n(z)$ is given as

$$e_n(z) = r(z)^n q(z) \tag{2.10}$$

with the row vector $q(z) = b^T(I - z\mathcal{A})^{-1}$; cf. [13, Lemma 2.4]. We note that

$$y_{n+1}^{(\lambda)} = h \sum_{j=0}^{n} e_{n-j}(h\lambda)\, g_j \tag{2.11}$$

is the Runge–Kutta approximation at time t_{n+1} of the linear initial value problem

$$y' = \lambda y + g(t), \quad y(0) = 0\,. \tag{2.12}$$

The convolution quadrature (2.7) is thus interpreted as

$$u_{n+1} = \frac{1}{2\pi i} \int_\Gamma F(\lambda)\, y_{n+1}^{(\lambda)}\, d\lambda\,;$$

see [13, Proposition 2.1].

3. Approximation of the contour integrals. The fast convolution algorithm will be based on discretizing the integrals in (2.2) and (2.9) along suitable complex contours. This approximation is discussed in the present section.

3.1. Quadrature on hyperbolas and Talbot contours. The fast algorithm approximates the quadrature weights ω_n by linear combinations of the exponential approximations $e_n(h\lambda)$, locally on a sequence of fast-growing time intervals $nh \in I_\ell$:

$$I_\ell = [B^{\ell-1}h, 2B^\ell h), \tag{3.1}$$

where the base $B > 1$ is an integer. For example, $B = 10$ was found a good choice in our numerical experiments. The approximation on I_ℓ results from applying the trapezoidal rule to a parametrization of the contour integral for the convolution quadrature weights,

$$\omega_n = \frac{h}{2\pi i} \int_{\Gamma_\ell} e_n(h\lambda) \otimes F(\lambda)\, d\lambda \approx h \sum_{k=-K}^{K} w_k^{(\ell)} e_n(h\lambda_k^{(\ell)}) \otimes F(\lambda_k^{(\ell)})\,, \qquad nh \in I_\ell, \tag{3.2}$$

with an appropriately chosen complex contour Γ_ℓ. The number of quadrature points on Γ_ℓ, $2K+1$, is chosen independent of ℓ. It is much smaller than what would be required for a uniform approximation of the contour integral on the whole interval $[0, T]$. The integration contour is chosen as a hyperbola [8] or a Talbot contour [20, 16].

The hyperbola is given by

$$\mathbb{R} \to \Gamma : \theta \mapsto \gamma(\theta) = \mu(1 - \sin(\alpha + i\theta)) + \sigma \tag{3.3}$$

with a positive scale parameter μ, a positive angle $\alpha < \pi/2 - \varphi$, and the shift σ of (1.4). The contour thus remains in the sector (1.4) of analyticity of $F(s)$. Additionally the contour is chosen such that the singularities of $e_n(hs)$ lie to the right of the contour. See the left part of Figure 3.1 for $\alpha = \pi/2 - 1/2$. The parameter μ will depend on ℓ, which yields a contour Γ_ℓ depending on the approximation interval I_ℓ. The weights and quadrature points in (3.2) are given by (omitting ℓ in the notation)

$$w_k = \frac{i\tau}{2\pi}\, \gamma'(\theta_k)\,, \quad \lambda_k = \gamma(\theta_k) \qquad \text{with} \qquad \theta_k = k\tau\,,$$

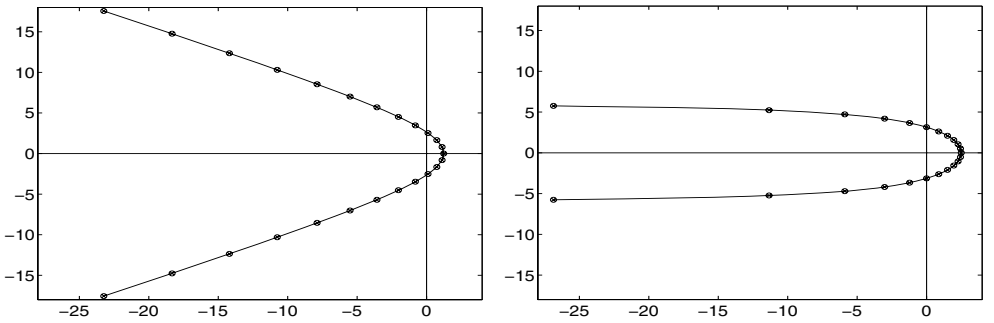


FIG. 3.1. *Hyperbola (left) and Talbot contour (right).*

where τ is a step length parameter.

Alternatively, the Talbot contour is given by

$$(-\pi,\pi) \to \Gamma : \theta \mapsto \gamma(\theta) = \mu\,(\theta\cot(\theta) + i\kappa\theta) + \sigma \tag{3.4}$$

where the parameters μ, κ, and σ are such that the singularities of $F(s)$ lie to the left of the contour and that the singularities of $e_n(hs)$ lie to the right of the contour. The parameter μ will again depend on ℓ. See the right part of Figure 3.1 for $\sigma = 0$. The weights and quadrature points in (3.2) are given by (omitting ℓ in the notation)

$$w_k = -\frac{i}{2(K+1)}\,\gamma'(\theta_k)\,, \qquad \lambda_k = \gamma(\theta_k) \qquad \text{with} \qquad \theta_k = \frac{k\pi}{K+1}\,.$$

3.2. Theoretical error bounds of the contour integral approximations. For the case of the hyperbola, we obtain with the proof of [7, Theorem 3] the following error bound, which shows exponential convergence.

THEOREM 3.1. [illegible] C, d $c_0, \dots, c_4$ [illegible] c [illegible] $t = nh \le T$ [illegible] (3.2) [illegible] (3.3) [illegible] $1 \le c\mu t \le n$ [illegible]

$$\|E(\tau,K,h,n)\| \le C\,h\,t^{\nu-1}\,(\mu t)^{1-\nu}\left(\frac{e^{c_0\mu t}}{e^{2\pi d/\tau}-1} + e^{(c_1-c_2\cosh(K\tau))\mu t} + e^{c_3\mu t}\Big(1+\frac{c_4\cosh(K\tau)\mu t}{n/2}\Big)^{-n/2}\right),$$

[illegible] ν [illegible] (1.4)

The first summand in this estimate corresponds to the error in the discretization of the integral (3.2), parametrized over the real line with an integrand holomorphic in a strip $|\mathrm{Im}\,\theta| < d$, with step size τ and with the infinite quadrature sum over all integers k [18, 19]. The other two terms result from truncating the infinite sum to the terms $-K \le k \le K$, using the strong decay of the integrand along the integration contour.

Given an error tolerance ε, the first term in the error bound becomes $O(\varepsilon\,h\,t^{\nu-1})$ if τ is chosen so small that $c_0\mu t - 2\pi d/\tau \le \log\varepsilon$, which requires an asymptotic proportionality $\frac{1}{\tau} \sim \log\frac{1}{\varepsilon} + \mu t$. For μ chosen such that $\frac{a_1}{B}\log\frac{1}{\varepsilon} \le \mu t \le a_1\log\frac{1}{\varepsilon}$ with an arbitrary positive constant a_1 and with $B > 1$, we obtain that the second term is $O(\varepsilon\,h\,t^{\nu-1})$ if $c_1 - c_2\cosh(K\tau) \le -B/a_1$, i.e., with $\cosh(K\tau) = a_2$ for a sufficiently

large constant a_2. With the above choice of τ, this yields $K \sim \log \frac{1}{\varepsilon}$. The third term then becomes smaller than $\varepsilon\, h\, t^{\nu-1}$ for $n \geq c \log \frac{1}{\varepsilon}$ with a sufficiently large constant c. In summary, this gives the following bound for the required number of quadrature points on the hyperbola.

THEOREM 3.2. (3.2) $\varepsilon\, h\, t^{\nu-1}$ $nh \in I_\ell$ $K = O(\log \frac{1}{\varepsilon})$ $n \geq c \log \frac{1}{\varepsilon}$ $c > 0$ K ℓ n h $nh \leq T$

No small error bound is, however, obtained for the first few $n < c \log \frac{1}{\varepsilon}$. The approximation is indeed poor for small n, as will be seen in the numerical experiments in section 3.4. Therefore the first few ω_n are not computed using the integral representation (3.2) but are computed using the method of section 3.5.

We expect that a similar result to Theorem 3.2 holds also for the Talbot contours, if the Laplace transform has an analytic continuation beyond the negative real axis from above and below, as is the case for the fractional powers considered below.

3.3. Choice of parameters for the hyperbolas. The estimate in Theorem 3.1 is valid for the discretization of contour integrals of the type considered in (3.2), where $e_n(h\lambda)$ is an approximation to $\exp(nh\lambda)$. Replacing $e_n(h\lambda)$ by $\exp(nh\lambda)$ in (3.2) we obtain the inverse Laplace transform at nh of F. For the numerical inversion of the Laplace transform there is a spectral order method developed in [9], where the error estimate is explicit in all constants involved. Getting an estimate explicit in all constants for the discretization of (3.2) is much more difficult and, in practice, we choose the parameters following the strategy proposed in [9] for the numerical inversion of the Laplace transform. This makes sense as for large n and small h with $nh \leq T$ the estimate in Theorem 3.1 tends to an expression of the same type as the error estimate for the approximation of the inverse Laplace transform in [9].

Given the approximation interval $[B^{\ell-1}h, (2B^\ell - 2)h]$ and K the number of nodes on the hyperbola, the strategy for choosing the parameters is as follows:

1. Choose $\alpha = d = (\pi/2 - \varphi)/2$, with φ of (1.4).
2. Minimize for $0 < \rho < 1$ the expression

$$\operatorname{eps}\, \epsilon_K(\rho)^{\rho-1} + \epsilon_K(\rho)^\rho,$$

   where

$$\epsilon_K(\rho) = \exp\Big(-\frac{2\pi d}{a(\rho)} K\Big), \qquad a(\rho) = \operatorname{acosh}\Big(\frac{2B}{(1-\rho)\sin\alpha}\Big)$$

   and eps is the machine precision.
3. Take

$$\tau = \frac{1}{K} a(\rho_{opt}), \qquad \mu = \frac{2\pi d K (1-\rho_{opt})}{(2B^\ell - 2) h\, a(\rho_{opt})}.$$

This strategy for choosing the hyperbola contours is used in the numerical experiments below.

3.4. Numerical experiments. In view of the examples of section 5 we present here numerical experiments with

$$f(t) = \frac{1}{\sqrt{\pi t}} \quad \text{for which} \quad F(s) = s^{-1/2}.$$

The error is calculated with respect to a reference solution, obtained for a discretization of the contour integral with a large number of integration points. For the

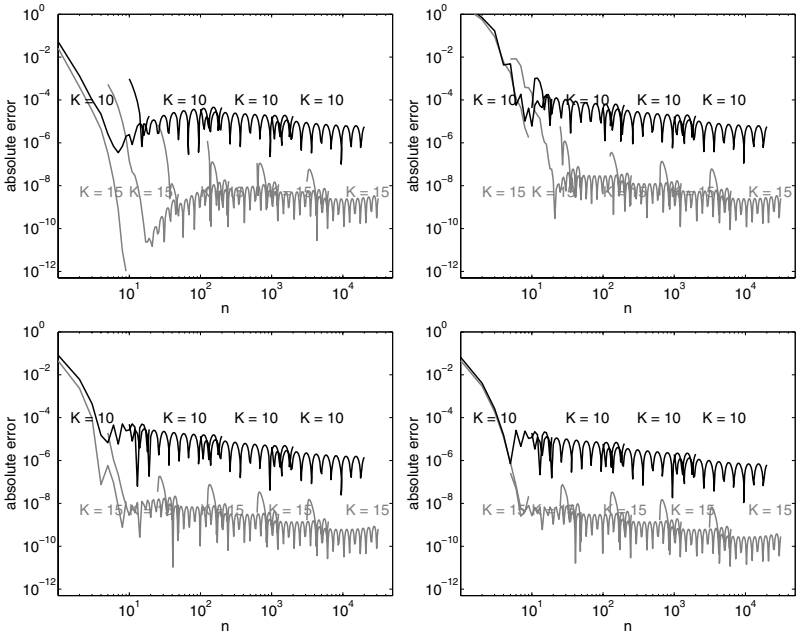


FIG. 3.2. *Hyperbola quadrature errors versus time in logarithmic scale for $K = 15$, $B = 5$ and $K = 10$, $B = 10$ for different Integrators. (Implicit Euler, BDF(2), RadauIIA(3), and RadauIIA(5) in clockwise order starting from the upper left).*

Radau IIA methods of order 3 and 5, where the ω_n are row vectors of dimension 2 and 3, respectively, we plot the error of the last entry.

Using the hyperbola contours, for $B = 10$ and $K = 10$ we obtain an absolute error of about 10^{-4} on the interval I_ℓ for $\ell \geq 2$ taking $\alpha = 1$, μ and τ as described above. For an absolute approximation error of $3 \cdot 10^{-8}$, we take $B = 5$, $K = 15$, and the other parameters as before; cf. Figure 3.2. For $n > 20$ there is no substantial difference between the quadrature errors corresponding to the different Runge–Kutta methods.

Using the Talbot contours, the following choices of parameters were experimentally found to give good results. An absolute accuracy of about 10^{-3} on the interval I_ℓ for $\ell \geq 2$ with right end-point T_ℓ is obtained with $B = 10$, $K = 10$, $\mu = 8/T_\ell$, $\kappa = 0.6$. For an absolute approximation error of 10^{-6}, take $B = 5$, $K = 15$, and the other parameters as before; cf. Figure 3.3. For $n > 20$ there is again no essential difference between the different Runge–Kutta methods.

As we commented after Theorem 3.2, the approximations to the first few convolution quadrature weights are poor and so they will not be used in the algorithm.

Figure 3.4 shows the absolute errors on the interval $[10, 20000]$ (similar for any interval $[a, 2000a]$ with $a > 10$) for the RadauIIA(3) method with $K = 10, 20, 40, 80, 160, 320, 640$. For the implicit Euler, the BDF(2), and the RadauIIA(5) methods these error plots look similar. This behavior of the errors clearly demonstrates the advantage of using local approximations. With $B = 10$, we need three approximation intervals to cover the interval $[10, 20000]$, so that for a work of $3 \cdot K$ with $K = 10$ we obtain better accuracy than with $K = 640$ over the whole interval.

In this example the maximum quadrature errors using the hyperbolas are smaller than those for the Talbot contours. Moreover, the hyperbolas allow us to choose larger

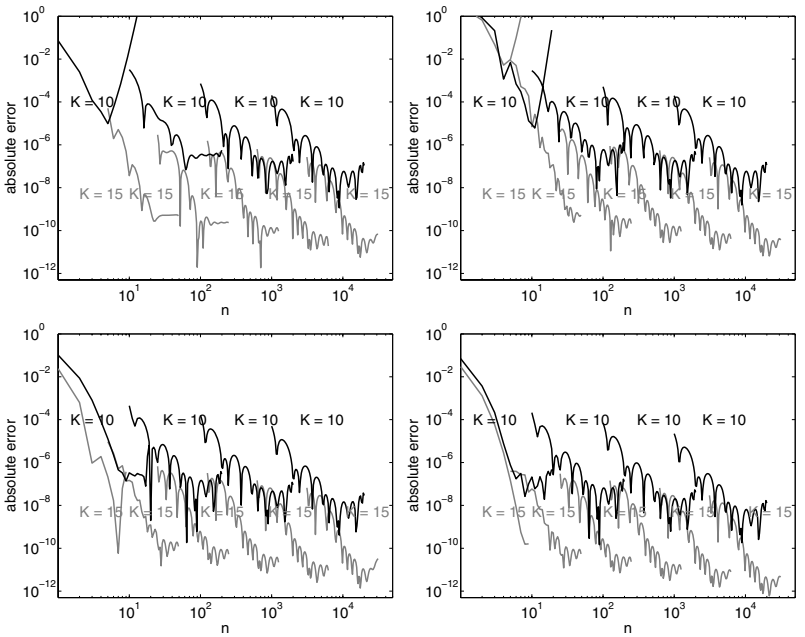


FIG. 3.3. *Talbot quadrature errors versus time in logarithmic scale for* $K = 15$, $B = 5$ *and* $K = 10$, $B = 10$ *for different Integrators. (Implicit Euler, BDF*(2)*, Radau*IIA(3)*, and Radau*IIA(5) *in clockwise order starting from the upper left).*

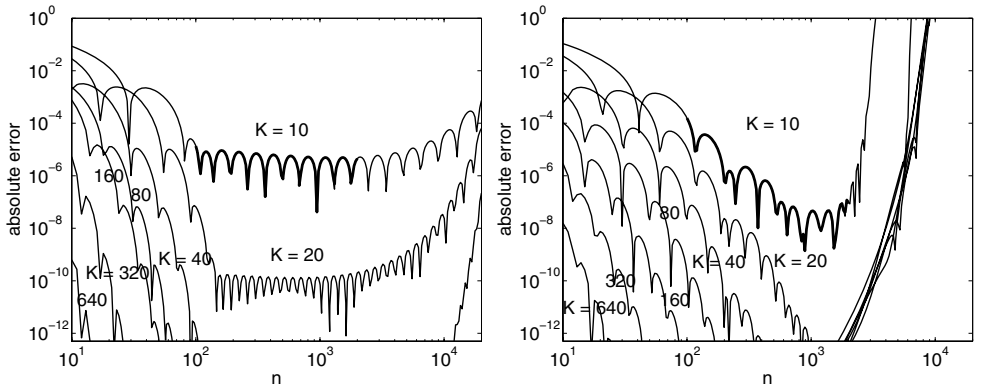


FIG. 3.4. *Hyperbola (left) and Talbot (right) quadrature error versus* n *in logarithmic scale for the Radau*IIA(3) *method with* $K = 10, 20, 40, 80, 160, 320, 640$. *The bold parts of the error curve correspond to the lower left parts of Figures* 3.3 *and* 3.2.

intervals. On the other hand, the Talbot contours turned out to be less sensitive to the choice of parameters and the Laplace transform functions than the hyperbolas.

3.5. Computation of the first few convolution quadrature weights. In view of the poor approximation properties of the discretized contour integral for the first few ω_n, the first N_0 (e.g., $N_0 = 10$ or 20, and asymptotically $N_0 = c \log \frac{1}{\varepsilon}$) convolution quadrature weights are approximated differently, using their representation

as an integral over a circle:

$$\omega_n = \text{ last row of } \frac{1}{2\pi i}\int_{|\zeta|=\rho} \zeta^{-n-1} F\Big(\frac{\Delta(\zeta)}{h}\Big)\, d\zeta, \qquad n \le N_0. \tag{3.5}$$

This is approximated by the trapezoidal rule,

$$\omega_n \approx \text{ last row of } \frac{\rho^{-n}}{J}\sum_{j=1}^{J} F\Big(\frac{\Delta(\zeta_j)}{h}\Big)\, e^{-2\pi i n j/J}, \qquad n \le N_0, \tag{3.6}$$

with $\zeta_j = \rho e^{-2\pi i j/J}$. The choice of the parameters ρ and J follows [11]: assuming that the values of F are computed with precision $\widehat{\epsilon}$, the error in ω_n $(0 \le n \le N_0)$ is $O(\widehat{\epsilon}^{1/2})$ if $J = N_0 + 1$ and $\rho^J = \widehat{\epsilon}^{1/2}$, and the error is $O(\widehat{\epsilon})$ if $J \ge N_0 \log(1/\widehat{\epsilon})$ and $\rho = e^{-\gamma h}$, with $\gamma > \sigma$ of (1.4).

4. The fast and oblivious algorithm. We now describe the convolution algorithm, concentrating on Runge–Kutta-based convolution quadrature. The algorithm differs slightly depending on whether we want to compute a convolution or to solve an integral or integrodifferential equation of convolution type.

4.1. The algorithm for computing convolutions. The algorithm presented here uses the organization scheme of the fast convolution algorithm described in a step-by-step manner in [14]. A pseudocode for the algorithm developed in [14] can be found in [5]. We now describe the fast and memory-saving algorithm for computing (2.7) for $n \le N$.

For fixed integer $n \le N$ and a given base B we split the discrete convolution (1.2) or (2.7) into L sums, where L is the smallest integer such that $n < 2B^L$:

$$u_{n+1} = \sum_{j=0}^{n} \omega_{n-j}\, g_j = u_{n+1}^{(0)} + \cdots + u_{n+1}^{(L)}$$

$$\text{with} \quad u_{n+1}^{(0)} = \omega_0\, g_n \quad \text{and} \quad u_{n+1}^{(\ell)} = \sum_{j=b_\ell}^{b_{\ell-1}-1} \omega_{n-j}\, g_j$$

for suitable $n = b_0 > b_1 > \cdots > b_{L-1} > b_L = 0$. In view of the approximation intervals (3.1), the splitting is done in such a way that for fixed ℓ in each sum from b_ℓ to $b_{\ell-1} - 1$, we have $n - j \in [B^{\ell-1}, 2B^\ell - 2]$. The $b_\ell = \mathtt{b}(\ell)$, $\ell = 1, \ldots, L-1$ are determined recursively by the following pseudocode:

```
L = 1; q = 0;
for n = 1 to N do
   if (2*B^L == n+1) then L = L+1; endif
   k = 1;
   while ((mod(n+1,B^k) == 0) & (k < L))
      q(k) = q(k)+1;
      k = k+1;
   endwhile
   for k = 1 to L-1 do b(k) = q(k)*B^k; endfor
endfor
```

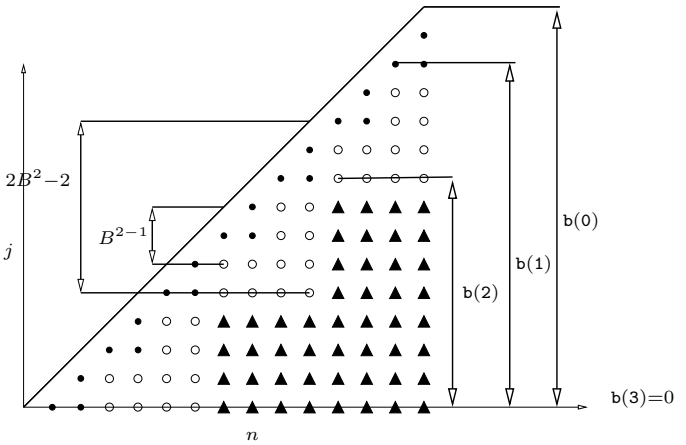


FIG. 4.1. *Visualization of the scheme for $B = 2$ to determine the values* `b(l)`. *For $n = 14$, $L = 3$ and* `b` $= [14\ 12\ 8\ 0]$. *Additionally the approximation interval for $\ell = 2$ is indicated.*

Note that for growing n, b_ℓ is augmented by B^ℓ every B^ℓ steps. The procedure is illustrated for $B = 2$ in Figure 4.1.

On inserting the integral representation (2.9) of the Runge–Kutta quadrature weights and the relation (2.10), i.e., $e_{n-j}(h\lambda) = r(h\lambda)^{n-j} q(h\lambda)$, we obtain

$$
\begin{aligned}
u_{n+1}^{(\ell)} &= \sum_{j=b_\ell}^{b_{\ell-1}-1} \omega_{n-j}\, g_j = \sum_{j=b_\ell}^{b_{\ell-1}-1} \frac{h}{2\pi i} \int_{\Gamma_\ell} e_{n-j}(h\lambda) \otimes F(\lambda)\, d\lambda\, g_j \\
&= \frac{1}{2\pi i} \int_{\Gamma_\ell} r(h\lambda)^{n-(b_{\ell-1}-1)}\, F(\lambda)\, y^{(\ell)}(h\lambda)\, d\lambda
\end{aligned}
\tag{4.1}
$$

with

$$
y^{(\ell)}(h\lambda) = h \sum_{j=b_\ell}^{b_{\ell-1}-1} e_{(b_{\ell-1}-1)-j}(h\lambda)\, g_j\,.
$$

Comparing this formula with (2.11), we see that $y^{(\ell)}(h\lambda)$ is the Runge–Kutta approximation $y(b_{\ell-1}h, b_\ell h, \lambda)$ to the solution at $t = b_{\ell-1}h$ of the linear initial-value problem

$$
y' = \lambda y + g(t), \quad y(b_\ell h) = 0, \tag{4.2}
$$

and hence $y^{(\ell)}(h\lambda)$ is computed as such, by Runge–Kutta time stepping. The integrals are discretized with the quadrature formula discussed in section 3:

$$
u_{n+1}^{(\ell)} \doteq \sum_{k=-K}^{K} w_k^{(\ell)}\, r(h\lambda_k^{(\ell)})^{n-b_{\ell-1}+1}\, F(\lambda_k^{(\ell)})\, y^{(\ell)}(h\lambda_k^{(\ell)}). \tag{4.3}
$$

In the nth time step, we thus compute $u_{n+1}^{(\ell)}$ and for subsequent use we update the Runge–Kutta solutions to the $(2K+1)L$ initial value problems (4.2) for the integration points $\lambda_k^{(\ell)}$ on the contours Γ_ℓ for $\ell = 1, \ldots, L$, doing one time step from t_n to t_{n+1} in each of these differential equations.

This algorithm does not keep the history g_j $(j = 0, \dots, n)$ in memory. Instead, for each $\ell = 1, \dots, L$, it stores the Runge–Kutta approximation `yRK(l)` to (4.2) at the current time step, the values $w_k^{(\ell)}$, $\lambda_k^{(\ell)}$, $r(h\lambda_k^{(\ell)})$, $F(\lambda_k^{(\ell)})$, the vector `y(l)` $= \big(y^{(\ell)}(h\lambda_k^{(\ell)})\big)$, and two auxiliary vectors `yA(l)` and `yM(l)` of the same dimension as `yRK(l)` and `y(l)` $((2K+1) \times (\text{size}\, g))$ needed for bookkeeping purposes.

Proceeding from t_n to t_{n+1} the Runge–Kutta solutions `yRK(l)` of (4.2) are advanced from $y(t_{b_\ell}, t_n, \lambda_k^{(\ell)})$ to $y(t_{b_\ell}, t_{n+1}, \lambda_k^{(\ell)})$ for $\ell = 2, \dots, L$ and all $k = -K \dots, K$.

The bookkeeping for the `yA` and `yM` is done using the following pseudocode:

```
l = 2;
while ((mod(n+1,Basis^(l-1)) == 0) & (l <= L))
  if (mod(n+1, Basis^l) == 0) then
    yA(l) = yRK(l);
    yM(l) = yRK(l);
    restart yRK(l);
  else
    yM(l) = yRK(l));
  endif
  l = l+1;
endwhile
```

Here "`restart yRK(l)`" means that `yRK(l)` is set to $y(t_{b_\ell}, t_{b_\ell}, \lambda_k^{(\ell)}) = 0$. This way the scheme in Figure 4.1 is build bottom up.

To retrieve the values `y(l)` $= y^{(\ell)}(h\lambda_k^{(\ell)})$ required at the current time from the `yA` and `yM` we use the `b(l)`. Again we give the pseudocode:

```
tvek = h*b;  % b is the vector of b(l); l = 0 , ... , L
for l = 2 to  length(tvek)-1 do
  y(l) = r(l) * y(l);
  if (tvek(l) == final time of yM(l)) then
    y(l) = yM(l);
    if (l == 2) then g = g(1:B-1);  endif
  endif
  if (tvek(l+1) ~= initial time of yM(l)) then
    y(l) = y(l) + yA(l)*r(l)^(B^(l-1));
  endif
endfor
```

Here `r(l)` denotes the stability function given in (2.10) and `g` is the vector containing the values $g_n, \dots, g_{b_1}$.

There are only $(2K+1)L$ evaluations of the Laplace transform $F(s)$. In the case of real functions $f(t)$ and $g(t)$ only the real parts of the above sums are used, since the imaginary ones cancel out, and hence the factor $2K+1$ can be replaced by $K+1$, since the quadrature points lie symmetric with respect to the real axis. We recall $L \le \log_B N$ and $K = O(\log \frac{1}{\varepsilon})$, where ε is the accuracy requirement in the discretization of the contour integrals.

In view of the poor approximation of the first convolution quadrature weights by the discretization of the contour integral, we evaluate $u_{n+1}^{(\ell)}$ directly for a few of the first ℓ, e.g., for $\ell = 0, 1$ with $B = 10$. For this we need to keep the $n - b_1 + 1 \le 2B$

values $g_{b_1}, \dots, g_n$ in memory, but none of the earlier history g_j for $j \le n - 2B$. We also need the few convolution quadrature weights $\omega_0, \dots, \omega_{2B-1}$, which are computed from (3.6) with $J = 2B$ evaluations of the Laplace transform $F(s)$, as described in section 3.5.

For the convolution quadrature based on the second-order BDF method a similar fast algorithm is obtained by inserting the formula (2.4) for $e_{n-j}(h\lambda)$ in (4.1).

4.2. The algorithm for solving integral equations. Consider now a Volterra integral equation

$$u(t) = a(t) + \int_0^t f(t-\tau)\, g(\tau, u(\tau))\, d\tau\,, \qquad t \ge 0, \tag{4.4}$$

with a kernel $f(t)$ whose Laplace transform satisfies the sectorial condition (1.4), with a smooth nonlinearity $g(t,u)$ and inhomogeneity $a(t)$. The adaptation of the above algorithm to such convolution equations is straightforward for the case of the convolution quadrature based on the implicit Euler method and the second-order BDF method, which use solution approximations only on the grid $t = nh$. The extension of the Runge–Kutta-based algorithm is, however, less immediate, because the integral approximation uses the internal stages of the Runge–Kutta method. Consider a Runge–Kutta-based convolution quadrature under the assumptions of section 2.2. With the column vector of internal stages $v_n = (v_{ni})_{i=1}^m$, the discretization of (4.4) reads

$$v_n = a_n + \sum_{j=0}^{n} W_{n-j}\, g_j\,, \qquad n \ge 0, \tag{4.5}$$

with $a_n = \big(a(t_n + c_i h)\big)_{i=1}^m$, with weight matrices W_n defined by (2.6), and with $g_j = \big(g(t_j + c_i h, v_{ji})\big)_{i=1}^m$ depending on the stages v_{ji}. The scheme is implicit in v_n. The solution at t_{n+1} is approximated by the last component of the stage vector v_n,

$$u_{n+1} = v_{nm}\,.$$

With the proof of [13, Theorem 4.1] we obtain that the error of this approximation over bounded time intervals is bounded by $O(h^k)$ with $k = \min(p, q+1)$, where p and q are the classical order and stage order, respectively, of the underlying Runge–Kutta method. This estimate holds under the assumption that the solution is sufficiently smooth. It gives orders 3 and 4 for the two- and three-stage Radau IIA methods, respectively. The precise approximation order for the three-stage method (of classical order 5) may become larger under appropriate conditions on the nonlinearity and the convolution kernel; cf. [13, Theorem 4.2].

The weight matrix W_n has the integral representation (cf. (2.9))

$$W_n = \frac{h}{2\pi i} \int_\Gamma E_n(h\lambda) \otimes F(\lambda)\, d\lambda\,,$$

where the $m \times m$ matrix $E_n(z)$ is defined by (2.8). By [13, Lemma 2.4], for $n \ge 1$, $E_n(z)$ is the rank-1 matrix given by

$$\begin{aligned} E_n(z) &= r(z)^{n-1} (I - z\mathcal{A})^{-1} \mathbb{1} b^T (I - z\mathcal{A})^{-1} \\ &= r(z)^{-1}\, (I - z\mathcal{A})^{-1} \mathbb{1}\, e_n(z). \end{aligned}$$

These relations permit us to proceed for the history term of (4.5) as we did for (2.7). We split the stage vector v_n as

$$v_n = a_n + v_n^{(0)} + \cdots + v_n^{(L)} \qquad \text{with} \qquad v_n^{(\ell)} = \sum_{j=b_\ell}^{b_{\ell-1}-1} W_{n-j}\, g_j$$

and obtain, like in (4.1),

$$v_n^{(\ell)} = \frac{1}{2\pi i} \int_{\Gamma_\ell} r(h\lambda)^{n-b_{\ell-1}} \, (I - h\lambda \mathcal{A})^{-1} \mathbb{1} \otimes F(\lambda)\, y^{(\ell)}(h\lambda)\, d\lambda \,,$$

where $y^{(\ell)}(h\lambda)$ is again the Runge–Kutta approximation at $t = b_{\ell-1}h$ to the initial-value problem (4.2), now for the inhomogeneity values $g_j = \big(g(t_j + c_i h, v_{ji})\big)_{i=1}^m$ in place of $g_j = \big(g(t_j + c_i h)\big)_{i=1}^m$. For $\ell \geq 2$ or 3, we thus approximate $v_n^{(\ell)}$ as

$$v_n^{(\ell)} \doteq \sum_{k=-K}^{K} w_k^{(\ell)}\, r(h\lambda_k^{(\ell)})^{n-b_{\ell-1}} \, (I - h\lambda_k^{(\ell)} \mathcal{A})^{-1} \mathbb{1} \otimes F(\lambda_k^{(\ell)})\, y^{(\ell)}(h\lambda_k^{(\ell)}) \,.$$

The algorithm stores the same values as before. The memory requirements for the algorithm are thus independent of the number of stages m and remain essentially the same as in the pure convolution case.

The effect of the (exponentially small in K) quadrature error of the contour integrals on the convolution quadrature approximation of the integral equation can be estimated using a discrete Gronwall inequality or by stability estimates of the convolution quadrature as, e.g., in [1]. Since the emphasis of the present paper is on the algorithmic aspects, we do not work out details of such estimates here.

5. Numerical experiments. We give two examples to illustrate the application and behavior of the fast convolution algorithm.

5.1. A nonlinear Volterra equation. We consider a nonlinear Volterra integral equation with weakly singular kernel from [6],

$$u(t) = -\int_0^t \frac{\big(u(\tau) - \sin(\tau)\big)^3}{\sqrt{\pi(t-\tau)}}\, d\tau \,. \tag{5.1}$$

The convolution quadrature based on the backward Euler method gives the implicit discretization

$$u_{n+1} = \sum_{j=0}^{n} \omega_{n-j} \big(u_{j+1} - \sin((j+1)h)\big)^3,$$

where ω_n is given by (1.3) with $F(s) = s^{-1/2}$ and $\delta(\zeta) = 1 - \zeta$. To solve the nonlinear equation in each time step we use Newton iterations. The history term is computed by the fast algorithm of the previous section.

We consider also the discretizations based on the backward differentiation method of order 2 (cf. section 2.1) and on the two- and three-stage Radau IIA implicit Runge–Kutta methods of orders 3 and 5, respectively; see sections 2.2 and 4.2.

In the numerical experiment we use the base $B = 5$ and the Talbot contours with $K = 15$ and $K = 30$ and the further parameters as in section 3.4. We choose

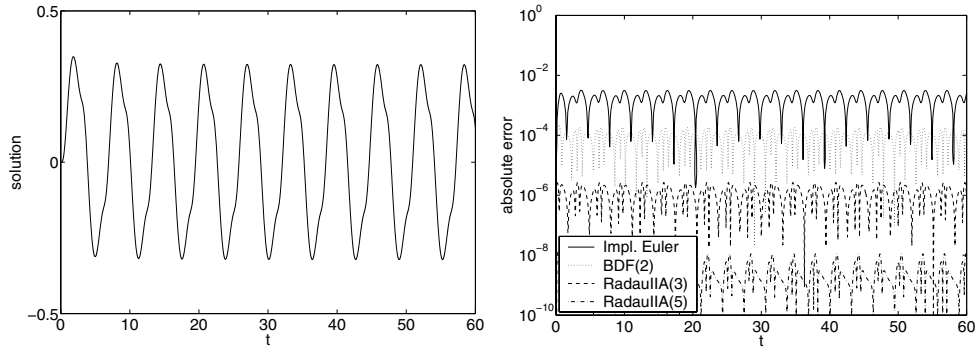


FIG. 5.1. *Evolution of the solution over the interval* $[0, 60]$ *(left) and logarithm of absolute error for different time integration methods (right) for* $h = 0.05$ *and* $K = 30$.

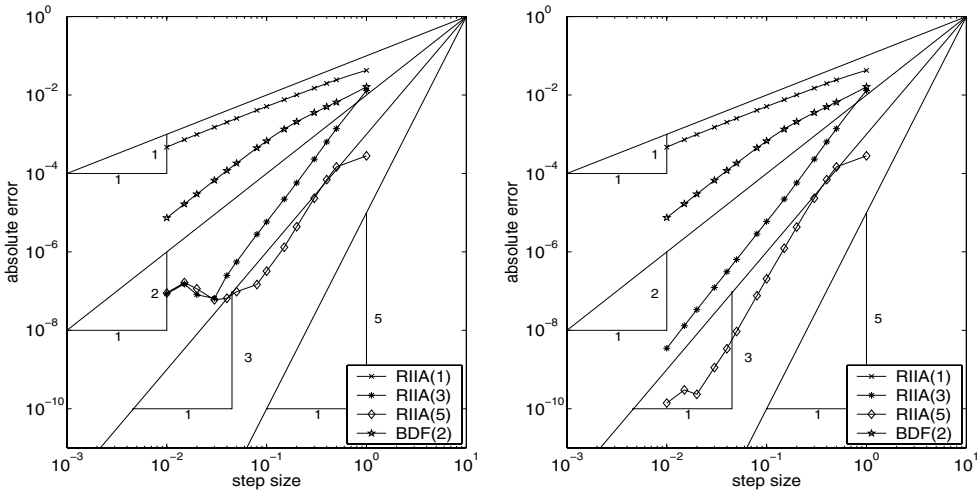


FIG. 5.2. *Absolute error versus step size* h *in logarithmic scale, for different integration methods, with* $K = 15$ *(left) and* $K = 30$ *(right).*

a tolerance of 10^{-12} in the Newton method. The error is calculated with respect to a reference solution, obtained with $h = 0.001$. Figure 5.1 shows the evolution of the absolute error and the oscillating solution u.

Figure 5.2 shows the errors $u_n - u(t_n)$ versus the step size h at time $t_n = 60$ for $K = 15$ and $K = 30$.

Figure 5.3 plots the cpu time versus the number of integration steps, up to 10^6 time steps. The near-linear growth of the computational work is clearly visible.

5.2. Fractional diffusion with transparent boundary conditions. Here we consider a fractional diffusion equation on the real line; see, e.g., [15] for applications of such equations in physics and for numerous references. The equation can be formulated as

$$u(x,t) - u_0(x) = \int_0^t \frac{(t-\tau)^{\alpha-1}}{\Gamma(\alpha)}\, \partial_{xx} u(x,\tau)\, d\tau + g(x,t) \quad \text{for } x \in \ , \quad t > 0, \tag{5.2}$$

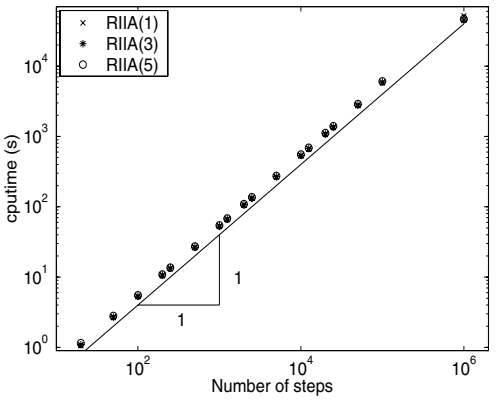


FIG. 5.3. *Cpu time in seconds versus the number of integration steps.*

with the asymptotic condition $u(x,t) \to 0$ for $x \to \pm\infty$, for an inhomogeneity g with $g(x,0) = 0$. To reduce the computation to a finite domain $x \in [-a,a]$ for initial data u_0 and inhomogeneity g with support in $[-a,a]$, we impose transparent boundary conditions at $x = \pm a$, which read

$$u(\pm a,t) = -\int_0^t \frac{(t-\tau)^{\alpha/2-1}}{\Gamma(\alpha/2)}\, \partial_\nu u(\pm a,\tau)\, d\tau, \tag{5.3}$$

with the outward derivative $\partial_\nu = \pm\partial_x$ at $x = \pm a$. These boundary conditions are derived with Laplace transform techniques in the same way as for the wave or the Schrödinger equation; see, e.g., [2]. Space discretization of (5.2) is done using second-order finite differences and a central finite difference to approximate the normal derivative. With the notation

$$\delta_{xx} u_l^n = \frac{1}{\Delta x^2}\big(u_{l-1}^n - 2u_l^n + u_{l+1}^n\big)\ , \quad \delta_\nu u_{\pm(M-1)}^n = \frac{1}{2\Delta x}\big(u_{\pm M}^n - u_{\pm(M-2)}^n\big)$$

for $a = M\Delta x$, the discrete equation approximating (5.2) is

$$\begin{aligned} u_l^{n+1} - u_l^0 &= \sum_{j=0}^{n} \omega_{n-j}^{(\alpha)}\, \delta_{xx} u_l^{j+1} + g_l^n \quad \text{for } l = -(M-1),\dots,M-1\ ;\ n>0, \\ u_{\pm(M-1)}^{n+1} &= -\sum_{j=0}^{n} \omega_{n-j}^{(\alpha/2)}\, \delta_\nu u_{\pm(M-1)}^{j+1}\ , \end{aligned} \tag{5.4}$$

where the weights $\omega_n^{(\beta)}$ are the convolution quadrature weights of the backward Euler method for the kernel $f(t) = t^{\beta-1}/\Gamma(\beta)$ with Laplace transform $F(s) = s^{-\beta}$. In this case $g_l^n = g(x_l, t_n + h)$.

In the numerical example we set $a = 5$ and $M = 450$. We consider the problem with $\alpha = 2/3$ and no inhomogeneity, i.e., $g \equiv 0$. The initial value is $u(x,0) = \exp(-x^2)$. Figure 5.4 shows the errors at $t = 2$ and $t = 10$ in dependence on the step size for the convolution quadratures based on the Radau IIA methods of orders 1, 3, 5, obtained with $B = 5$ and $K = 15$ in the fast convolution algorithm, by using the hyperbola contours. The reference solution is obtained with the convolution quadrature based on the Radau IIA method of order 5, with a small step size $h = 0.0002$

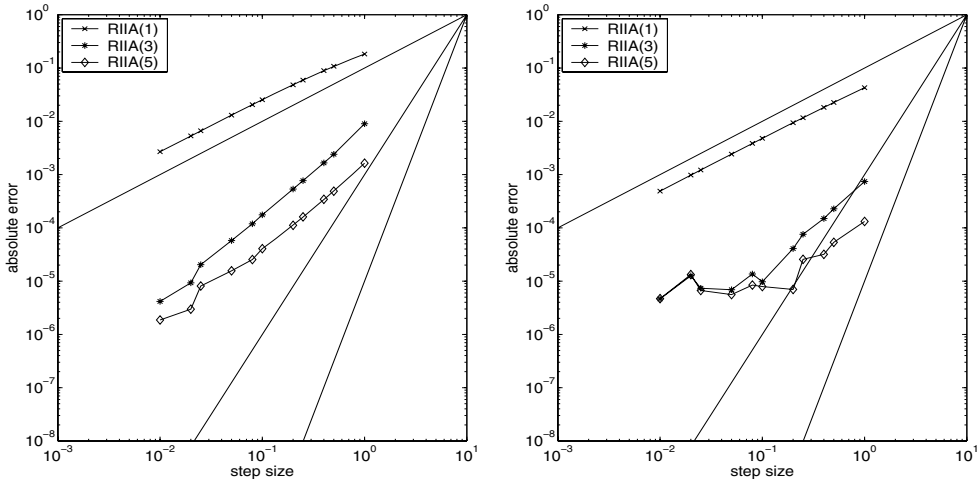


FIG. 5.4. *Absolute error versus time step in logarithmic scale, for different integration methods, with* $K = 15$. *Left* $t = 2$, *right* $t = 10$.

and large number $K = 40$ of nodes on the integration contour. We observe an order reduction for the higher-order methods, which is due to the temporal nonsmoothness of the solution at $t = 0$; cf. [1, sect. 8]. Nevertheless, the higher-order methods give much better accuracy.

The work diagram looks almost identical to Figure 5.3, showing practically linear dependence of the computational work on the number of time steps. The required memory is less than 200 entries per spatial grid point for up to $N \leq 10^4$ steps and less than 300 entries per grid point for $N \leq 10^6$ steps. These numbers are halved if we run the algorithm with $B = 10$, $K = 10$ instead of $B = 5$, $K = 15$, as is sufficient for less stringent accuracy requirements ($\sim 10^{-3}$).

Acknowledgment. We thank Cesar Palencia for helpful discussions during the preparation of this paper.

REFERENCES

[1] E. CUESTA, C. LUBICH, AND C. PALENCIA, *Convolution quadrature time discretization of fractional diffusion-wave equations*, Math. Comp., 75 (2006), pp. 673–696.

[2] T. HAGSTROM, *Radiation boundary conditions for numerical simulation of waves*, Acta Numer., 8 (1999), pp. 47–106.

[3] E. HAIRER, C. LUBICH, AND M. SCHLICHTE, *Fast numerical solution of nonlinear Volterra convolution equations*, SIAM J. Sci. Statist. Comput., 6 (1985), pp. 532–541.

[4] E. HAIRER AND G. WANNER, *Solving Ordinary Differential Equations.* II. *Stiff and Differential-Algebraic Problems*, 2nd ed., Springer, Berlin, 1996.

[5] R. HIPTMAIR AND A. SCHÄDLE, *Non-reflecting boundary conditions for Maxwell's equations*, Computing, 71 (2003), pp. 265–292.

[6] N. LEVINSON, *A nonlinear Volterra equation arising in the theory of superfluidity*, J. Math. Anal. Appl., 1 (1960), pp. 1–11.

[7] M. LÓPEZ-FERNÁNDEZ, C. LUBICH, C. PALENCIA, AND A. SCHÄDLE, *Fast Runge-Kutta approximation of inhomogeneous parabolic equations.* Numer. Math., 102 (2005), pp. 277–291.

[8] M. LÓPEZ-FERNÁNDEZ AND C. PALENCIA, *On the numerical inversion of the Laplace transform of certain holomorphic mappings*, Appl. Numer. Math., 51 (2004), pp. 289–303.

[9] M. LÓPEZ-FERNÁNDEZ, C. PALENCIA, AND A. SCHÄDLE, *A spectral order method for inverting sectorial Laplace transforms*, SIAM J. Numer. Anal., to appear.

[10] C. Lubich, *Convolution quadrature and discretized operational calculus.* I, Numer. Math., 52 (1988), pp. 129–145.
[11] C. Lubich, *Convolution quadrature and discretized operational calculus.* II, Numer. Math., 52 (1988), pp. 413–425.
[12] C. Lubich, *Convolution quadrature revisited*, BIT, 44 (2004), pp. 503–514.
[13] C. Lubich and A. Ostermann, *Runge-Kutta methods for parabolic equations and convolution quadrature*, Math. Comput., 60 (1993), pp. 105–131.
[14] C. Lubich and A. Schädle, *Fast convolution for nonreflecting boundary conditions*, SIAM J. Sci. Comp., 24 (2002), pp. 161–182.
[15] R. Metzler and J. Klafter, *The random walk's guide to anomalous diffusion: A fractional dynamics approach*. Phys. Rep., 339 (2000), pp. 1–77.
[16] M. Rizzardi, *A modification of Talbot's method for the simultaneous approximation of several values of the inverse Laplace transform*, ACM Trans. Math. Software, 21 (1995), pp. 347–371.
[17] A. Schädle, *Ein Schneller Faltungsalgorithmus für Nichtreflektierende Randbedingungen*, Doctoral Thesis, University of Tübingen, Germany, 2002.
[18] F. Stenger, *Approximations via Whitaker's cardinal function*, J. Approx. Theory, 17 (1976), pp. 222–240.
[19] F. Stenger, *Numerical methods based on Whitaker cardinal, or sinc functions*, SIAM Rev., 23 (1981), pp. 165–224.
[20] A. Talbot, *The accurate numerical inversion of Laplace transforms*. J. Inst. Math. Appl., 23 (1979), pp. 97–120.

SIAM J. SCI. COMPUT.
Vol. 28, No. 2, pp. 439–458
© 2006 Society for Industrial and Applied Mathematics

TWO-LEVEL TOEPLITZ PRECONDITIONING: APPROXIMATION RESULTS FOR MATRICES AND FUNCTIONS*

D. NOUTSOS†, S. SERRA CAPIZZANO‡, AND P. VASSALOS†

Abstract. Large 2-level Toeplitz systems arise in a variety of applications (see, e.g., [R. H. Chan and M. Ng, *SIAM Rev.*, 38 (1996), pp. 427–482]) for which efficient numerical methods for their solution are required. Some successful numerical techniques need the explicit knowledge of the generating function f of the considered system $T_n(f)\mathrm{x} = b$, an assumption that usually is not fulfilled in real applications. In this paper we analyze and complete the procedure proposed in [D. Noutsas, S. Serra Capizzano, and P. Vassalos, *Numer. Linear Algebra Appl.*, 12 (2005), pp. 231–239] for the 2-level case. In such a way, from the knowledge of the coefficients of $T_n(f)$, we determine optimal preconditioning strategies for the solution of our systems. Finally, some numerical experiments are performed and discussed in connection with our theoretical analysis.

Key words. 2-level Toeplitz matrix, conjugate gradient, preconditioning, preconditioned conjugate gradient

AMS subject classifications. 65F10, 47B25, 64T10, 15A18

DOI. 10.1137/050627046

1. Introduction.

In this paper we consider the preconditioning of 2-level Toeplitz systems of the form $T_n(f)\mathrm{x} = b$, where $n = (n_1, n_2)$ have large components, the symbol f is assumed to be defined on $\mathbf{R}$, real-valued, 2π-periodic, and continuous, $(T_n(f))_{(j,k)(p,q)} = t_{k-j,q-p}$ with $t_{r,s}$ being the Fourier coefficients of f, i.e.,

$$t_{r,s} = \frac{1}{4\pi^2}\int_{-\pi}^{\pi}\int_{-\pi}^{\pi} f(x,y)\exp(-\mathbf{i}(rx+sy))dxdy, \quad \mathbf{i}^2 = -1.$$

Here the 2-index notation $(T_n(f))_{(j,k)(p,q)}$, $n = (n_1, n_2)$, indicates that we are selecting the block (j,k) of size n_2 with $j,k \in \{1,\ldots,n_1\}$ and, in that block, we are selecting the entry (p,q), $p,q \in \{1,\ldots,n_2\}$.

Such matrices (often also called block Toeplitz with Toeplitz blocks) arise in several applications (see, e.g., [6, 11, 13]), such as Markov chains, integral equations, in the solution of certain partial differential equations (PDEs), and image restoration. In some contexts the generating function f is explicitly given or can be easily obtained, but in many others, like image processing, Markov chains, and tomography, the analytic expression of the symbol is not available and, as a consequence, we do not know crucial properties such as the presence of zeros, their localization, or their multiplicities. We recall that the considered information is essential to understand the spectral properties (extreme eigenvalues, ill-conditioning, ill-posedness) of the matrices and therefore to select the most suitable preconditioning methods (see [19, 12]). The 1-level case has been treated in [23] using band Toeplitz preconditioners and in

*Received by the editors March 17, 2005; accepted for publication (in revised form) October 28, 2005; published electronically March 31, 2006. This research was partially funded by a grant of the project PYTHAGORAS from the Greek Ministry of Education and by "Ministero dell'Università e della Ricerca" (MIUR) grant 2002014121.

http://www.siam.org/journals/sisc/28-2/62704.html

†Department of Mathematics, University of Ioannina, GR-451 10, Ioannina, Greece (dnoutsos@cc.uoi.gr, pvassal@cc.uoi.gr).

‡Dipartimento di Fisica e Matematica, Università dell'Insubria—Sede di Como, Italy (stefano.serrac@uninsubria.it, serra@mail.dm.unipi.it).

[18] using circulant-like preconditioners constructed by positive reproducing kernels. These circulant preconditioners, as well as those belonging to more general matrix algebras, possess a natural extension in the 2-level setting. However, a quite general theory shows that it is impossible to find any of these 2-level extensions preserving either the superlinear or the optimal behavior: see [31, 25, 32] for negative results regarding the notion of superlinearity and [14, 15] for negative results regarding the notion of optimality.

We recall that the spectral properties of the Toeplitz matrices are described very precisely by the symbol (see, e.g., [3, 16]). For instance, if zero belongs to the range of f, then the sequence $\{T_n(f)\}_n$ is asymptotically ill-conditioned (for $n \to \infty$, $n = (n_1, n_2)$, i.e., for $n_1, n_2 \to \infty$): more precisely, the problem will be ill-posed in a discrete sense if f has a nondefinite sign while we have invertibility but asymptotical ill-conditioning if f is nonnegative (or equivalently nonpositive). In the latter we need preconditioning and an optimal technique consists in using $T_n(g)$ as a preconditioner where g is a nonnegative trigonometric polynomial which has the same zeros as f with the same orders. Any system related to $T_n(g)$ is banded and can be solved, under suitable assumptions, by multigrid methods [9] or cyclic reduction techniques [4, 5] in linear time; moreover, the preconditioning sequence is optimal so that only a fixed number of preconditioned conjugate gradient (PCG) iterations (independent of n_1 and of n_2) has to be performed in order to reach the solution within a preassigned accuracy. In conclusion, we need algorithms for determining analytical information on the zeros of f. In [17], by extending the proposal in [23], the 2-level case is considered for generating functions only known through the matrix coefficients (i.e., the Fourier coefficients of f): the main idea is to approximate f over the grid

$$\mathcal{S}_n = \left\{ \left(-\pi + \frac{2k\pi}{n_1} \ , \ -\pi + \frac{2j\pi}{n_2} \right), k = 1, 2, \ldots, n_1 \ , \ j = 1, 2, \ldots, n_2 \right\} \tag{1.1}$$

on the square $Q = (-\pi, \pi]^2$, and then we proceed by looking for the roots of f, by estimating their multiplicities, by characterizing the problem (well-conditioned, ill-conditioned, ill-posed), by choosing the appropriate method, and finally by constructing the chosen preconditioner. Since the zeros cannot be computed exactly in general, we also need to check the robustness of the preconditioning technique with regard to numerical/approximation errors in the computation of the position of the zeros. The contribution of this paper relies in giving a theoretical support to the heuristic proposal given in [17], by using and extending new tools developed in [29], and in giving a numerical procedure for approximating the polynomial factors of the symbol in the case of curves of zeros.

The paper is organized as follows. In section 2 we report basic results from the quoted literature, and in section 3 the proposed approach is theoretically analyzed and some numerical results are given.

2. Basic theory.

2.1. Results from the literature. In the following when we write inf and sup we mean the essential infimum and the essential supremum, i.e., up to zero measure sets (with respect to the Lebesgue measure); when we write $f \sim g$ we mean that there exist positive real constants r and R such that $rg(x, y) \leq f(x, y) \leq Rg(x, y)$ for almost every (x, y) in the definition set of f and g. Moreover, when we write $a_n \sim b_n$ and $\{a_n\}$ and $\{b_n\}$ are sequences, we understand that there exist positive real constants r and R such that $rb_n \leq a_n \leq Rb_n$ for every n. (Notice that in both

the instances, the case $R = +\infty$ is excluded.) A nonnegative function f has a zero at (x_0, y_0) if for every neighborhood I of (x_0, y_0) we have $\inf_{(x,y)\in I} f(x, y) = 0$. A set $\mathcal{S}$ of zeros of $f \geq 0$ is isolated if there exists a neighborhood J of $\mathcal{S}$ such that for every neighborhood $I \subset J, I \neq J$ of $\mathcal{S}$ we have $\inf_{(x,y)\in J\setminus I} f(x, y) > 0$. Furthermore, we write that a nonnegative function f has a zero of order at least $\alpha \geq 0$ at (x_0, y_0) if and only if there exist a positive constant c such that $f(x, y) \leq c\|(x, y) - (x_0, y_0)\|^\alpha$ in a suitable neighborhood of (x_0, y_0) and for some norm $\|\cdot\|$. We write that its order is $+\infty$ if it is of order at least α for every $\alpha > 0$ or in other words if

$$\lim_{(x,y)\to(x_0,y_0)} \frac{f(x, y)}{\|(x, y) - (x_0, y_0)\|^\alpha} = 0 \quad \forall \alpha > 0.$$

For our purpose, we will consider the following general definition for a zero of order $\alpha > 0$.

DEFINITION 1. [illegible] f [illegible] $L^1(Q)$ [illegible] (x_0, y_0) [illegible] $\alpha \in (0, \infty)$ [illegible] p [illegible] $\mathcal{C}_i$ $i = 1, \ldots, p$ [illegible] $l_i(x, y) = 0$ [illegible] (x_0, y_0) [illegible] $f \sim \breve{f}$ [illegible]

$$\breve{f}(x, y) = \sum_{i=1}^{p} |l_i(x, y)|^\alpha + g(x, y),$$

[illegible] g [illegible] (x_0, y_0) [illegible] $\beta > \alpha$

The above definition is quite general and includes many different cases, such as $f(x, y) = |x - x_0|^4 + |y - y_0|^3$ ($f = \breve{f}$, isolated zero at (x_0, y_0), $p = 1$, $\alpha = 3$, $\beta = 4$, $l_1(x, y) = y - y_0$, $g(x, y) = |x - x_0|^4$); $f(x, y) = |1 - x^2 - y^2|^2 + h(x, y)\exp(-1/(1 - x^2 - y^2))$, $h(x, y) \in L^\infty(Q)$ with positive essential range ($f = \breve{f}$, isolated curve of zeros represented by the circle centered at zero with radius 1, $p = 1$, $\alpha = 2$, $\beta = \infty$, $l_1(x, y) = 1 - x^2 - y^2$, $g(x, y) = h(x, y)\exp(-1/(1 - x^2 - y^2))$); $f(x, y) = |\sin(x^2+y^4+(x-y)^2)|+(x^2+y^2)^4$ ($\breve{f}(x, y) = x^2+y^4+(x-y)^2+(x^2+y^2)^4$, isolated zero at $(0, 0)$, $p = 2$, $\alpha = 2$, $\beta = 4$, $l_1(x, y) = x$, $l_2(x, y) = x - y$, $g(x, y) = y^4 + (x^2 + y^2)^4$).

LEMMA 1 (see [21, 16]). [illegible] f [illegible] $L^1(Q)$ [illegible] $\alpha < \infty$ [illegible] 1 [illegible] $T_n(f)$ $n = (n_1, n_2)$ [illegible] $\lambda_{\min}(T_n(f))$ [illegible] $\lambda_{\min}(T_n(f)) \sim \nu^{-\alpha}$ [illegible] $n_1 \sim n_2 \sim \nu$ [illegible] $f = f_n$ [illegible] n [illegible] n [illegible] 1

For a generic n-by-n matrix X with complex entries we define

$$\|X\|_p = \begin{cases} \left(\frac{\sum_{j=1}^n \sigma_j^p}{n}\right)^{1/p} & \text{if } p \in [1, \infty), \\ \sigma_1 & \text{if } p = \infty \end{cases}$$

with $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n$ denoting the singular values of X. Furthermore, indicating by $m\{\cdot\}$ the Lebesgue measure (on Q in the present case), for $f \in L^p(Q)$ and $p \in [1, \infty)$, we define $\|f\|_p = \left[\frac{1}{m\{Q\}} \int_Q |f(x)|^p \, dx\right]^{1/p}$ and $\|f\|_\infty = \sup_{x\in Q} |f(x)|$ for $f \in L^\infty(Q)$, $p = \infty$, and sup denoting the essential supremum, i.e., up to zero measure sets.

LEMMA 2 (see [30, 26]). [illegible] f [illegible] $L^p(Q)$ [illegible] $\|T_n(f)\|_p \leq \|f\|_p$ [illegible] $\lim_{n\to\infty} \|T_n(f)\|_p = \|f\|_p \ \forall p \in [1, \infty]$

Finally, for θ Lebesgue measurable over $D \subset \mathbf{R}^k$, with k positive integer and $0 < m\{D\} < \infty$, the expression "$\{A_n\}$ is distributed as the measurable function θ" with A_n of size $N(n)$ is equivalent to writing that for any $F \in \mathcal{C}_0$ (continuous with bounded support) it holds

$$\lim_{n\to\infty} \frac{1}{N(n)} \sum_{j=1}^{N(n)} F\left(\lambda_j\left(A_n\right)\right) = \frac{1}{m\{D\}} \int_D F\left(\theta(x)\right)\, dx,$$

where $\{\lambda_j\left(A_n\right) : \ j = 1, \ldots, N(n)\}$ denotes the complete set of the eigenvalues of A_n and where n could be either a positive integer or a positive multi-index. Often the considered distribution results for the eigenvalues hold for Hermitian sequences, but there exist many noteworthy exceptions in the complex field (concerning non-Hermitian sequences): the case of Toeplitz sequences generated by complex-valued symbols whose range has nonempty interior and for which the complement of the range is a connected set in the complex field (see [33]), the case of quasi-Hermitian discretizations of finite difference differential operators (see [10]), and the case of the preconditioned sequences considered in Lemma 3. In our context we have $D = Q$, $k = 2$, and the matrices A_n are either Hermitian Toeplitz or preconditioned matrices (see Lemma 3) which can be turned into Hermitian ones by similarity transformations.

Now we recall a basic result which is the foundation of the band Toeplitz preconditioning.

LEMMA 3 (see [8, 27]). [illegible] f [illegible] g [illegible] $L^1(Q)$ [illegible] f/g [illegible]

$$\inf f/g < \lambda(T_n^{-1}(g)T_n(f)) < \sup f/g$$

[illegible] $\{T_n^{-1}(g)T_n(f)\}$ [illegible] f/g

We explain the band Toeplitz preconditioning approach with an example. Consider the function $f(x, y) = x^2 + y^2$ which has a unique zero of order two at $z_0 = (0, 0)$. The matrix $T_n(f)$ is full but can be optimally preconditioned by the simple band Toeplitz matrix

$$T_n(g) = \begin{bmatrix} B & -I & & & \\ -I & \ddots & \ddots & & \\ & \ddots & & \ddots & \\ & & \ddots & \ddots & -I \\ & & & -I & B \end{bmatrix}_{n_1 \times n_1}, \quad B = \begin{bmatrix} 4 & -1 & & & \\ -1 & \ddots & \ddots & & \\ & \ddots & & \ddots & \\ & & \ddots & \ddots & -1 \\ & & & -1 & 4 \end{bmatrix}_{n_2 \times n_2}$$

with I denoting the $n_2 \times n_2$ identity matrix and with $g(x, y) = 4 - 2\cos(x) - 2\cos(y) = 4\sin^2(x/2) + 4\sin^2(y/2)$. By Lemma 3 we deduce that the eigenvalues of the preconditioned matrix $T_n^{-1}(g)T_n(f)$ are

1. contained in the interval $(1, \pi^2/4)$, $1 = \min f/g$, $\pi^2/4 = \max f/g$, and
2. globally distributed as the function f/g.

Therefore, irrespectively of the 2-index $n = (n_1, n_2)$ and of the size $N(n) = n_1 n_2$ of the considered linear system, the number of PCG iterations for which we need to find the solution within a fixed accuracy $\epsilon > 0$ can be bounded by a universal constant independent of n (which can be estimated as a function of ϵ [1]). Moreover, for the solution of a system with matrix $T_n(g)$, we can use solvers based on fast sine transforms (see, e.g., [34]) costing $\mathcal{O}(N(n) \log N(n))$ arithmetic operations (flops),

multigrid methods [9], or just classical Poisson solvers (based on the cyclic reduction idea; see, e.g., [4, 5]) with cost of $\mathcal{O}(N(n))$ flops. In conclusion the associated PCG method is optimal (in the sense of Axelsson and Neytcheva [2]) since

1. the cost of each iteration is $\mathcal{O}(N(n)\log N(n))$ flops, $N(n) = n_1 n_2$, that is of the same order as the cost of a matrix vector product, and
2. the number of iterations does not depend on n.

We observe that the main point for deriving such a result does not rely on the regularity of the function f. Actually, if $f(x,y) = x^2 + y^2$ is replaced by $f(x,y) = (x^2+y^2)h(x,y)$, where $h(x,y)$ is any function in $L^\infty(Q)$ with positive essential range, then the same $T_n(g)$ is still an optimal preconditioner.

Therefore, the crucial points are the number of the zeros or curves of zeros of f, which should be finite, and their order, which should be even (for the case where the order of the zeros is not even, to our knowledge, the best idea is the use of multigrid strategies [9, 28]).

However, in many practical situations we possess the coefficients $\{t_{r,s}\}$, in some cases we know the positive definiteness of the matrix $T_n(f)$ or even the fact that $T_n(f)$ is asymptotically ill-conditioned, but very often we do not have an explicit expression for f. In that cases we have to recover the minimal information for devising an optimal preconditioning technique, i.e., the position and the order of the zeros of f.

In [23] an economic strategy was devised for the numerical evaluation of the order and the position of the zeros in the 1-dimensional case. The technique has been heuristically extended to two dimensions in [17]: in [29] the theoretical explanation of the goodness of the procedure for the 1-dimensional case was given. Here we complete the picture by giving theoretical results in the 2-dimensional setting and giving a numerical procedure for approximating the polynomial factors of the symbol in the case of curves of zeros. In actuality, under the assumption that the zeros are even, we can determine exactly the order of the zeros but the position of these zeros is affected by an error for which we can provide a rough a priori bound.

In conclusion, we would like to study the spectral behavior of

$$T_n^{-1}(\tilde{g})T_n(f),$$

where $f \sim g$, g is a trigonometric polynomial of fixed degrees and the zeros of $\tilde{g}$ are an approximation of those of g. A simple example is given by the situation

$$g(x,y) = \prod_{j=0}^{k}(4 - 2\cos(x - x_j) - 2\cos(y - y_j))^{\alpha_j},$$
$$\tilde{g}(x,y) = \prod_{j=0}^{k}(4 - 2\cos(x - \tilde{x}_j) - 2\cos(y - \tilde{y}_j))^{\alpha_j}$$

and $(\tilde{x}_j, \tilde{y}_j)$ is an approximation of (x_j, y_j) $\forall j = 0, \ldots, k$, even if the most challenging case is the one where the zeros form a curve: in this respect there is the problem of approximating such a curve by using the zeros of a trigonometric polynomial; since the symbol f is known only through the set of its Fourier coefficients, a direct fast solution (requiring only FFTs) can be obtained by considering the zeros of approximations of f such as the Fourier polynomial of degree n or its Cesaro sum of the same degree.

In the following subsection we furnish some mathematical tools with which to answer this question and in the subsequent subsection we analyze the spectral features of (approximate) preconditioned matrix sequences.

2.2. Theoretical results. We read the first part of Lemma 3 in the following way: if f and g are two essentially nonnegative functions (not identically zero) such that f/g is not constant and $f \sim g$, then

$$\kappa_p\left(T_n^{-1/2}(g)T_n(f)T_n^{-1/2}(g)\right) < C := (\sup f/g)(\inf f/g)^{-1} < \infty$$

for any $p \in [1,\infty]$ and with $\kappa_p(X) = \|X\|_p\|X^{-1}\|_p$ being the condition number of an invertible matrix X with respect to the (scaled) Schatten norm $\|\cdot\|_p$, $p \in [1,\infty]$. In the following we suppose to have a perturbed g, let us say g_n, such that g_n is not equivalent to f but it is close in norm to g. We try to see what happens to the condition numbers of the perturbed symmetrized preconditioned matrix

$$T_n^{-1/2}(g_n)T_n(f)T_n^{-1/2}(g_n).$$

THEOREM 1. [illegible] $f \sim g$ [illegible] $f_n \sim g_n$ [illegible] $n = (n_1, n_2)$ [illegible] $f, f_n, g, g_n \in L^p(Q)$ $p \in [1,\infty]$ [illegible] f [illegible] $\alpha < \infty$ [illegible] 1 [illegible] f_n [illegible] $\beta < \infty$ [illegible] 1 [illegible] C_1, C_2, q_1, q_2 [illegible] n [illegible]

$$\kappa_p\left(T_n^{-1/2}(g_n)T_n(f)T_n^{-1/2}(g_n)\right) \le \left[C_1 + q_1\nu^{\alpha}\|g_n - g\|_p\right]\left[C_2 + q_2\nu^{\beta}\|f_n - f\|_p\right]$$

[illegible] $n_1 \sim n_2 \sim \nu$ [illegible]. From the equivalence relationships $f_n \sim g_n$ and $f \sim g$, we infer that there exists constants C_1 and C_2, independent of n, such that

$$\sup g/f \le C_1, \quad \sup f_n/g_n \le C_2. \tag{2.1}$$

Moreover, from the nonnegativity of f and g_n and from the assumption on the maximal order of their zeros, by applying Lemma 1, we deduce that $\lambda_{\min}(T_n(f)) \sim \nu^{-\alpha}$ and that $\lambda_{\min}(T_n(g_n)) \sim \nu^{-\beta}$. Therefore, there exist absolute constants q_1 and q_2 independent of n for which we have

$$\|T_n^{-1}(f)\|_\infty \le q_1\nu^{\alpha}, \quad \|T_n^{-1}(g_n)\|_\infty \le q_2\nu^{\beta}. \tag{2.2}$$

Then, by the plain definition of the condition number, we have

$$\kappa_p\left(T_n^{-1/2}(g_n)T_n(f)T_n^{-1/2}(g_n)\right) = X_nY_n,$$

where

$$X_n = \|T_n^{-1/2}(g_n)T_n(f)T_n^{-1/2}(g_n)\|_p, \quad Y_n = \|T_n^{1/2}(g_n)T_n^{-1}(f)T_n^{1/2}(g_n)\|_p.$$

Moreover, since $\sigma_j(XY) \le \|X\|_\infty\sigma_j(Y)$, by definition of $\|\cdot\|_p$ norm, it follows that

$$\|XY\|_p \le \|X\|_\infty\|Y\|_p, \tag{2.3}$$

which corresponds to the submultiplicative property if $p = \infty$. Since $f_n \sim g_n$ with equivalence constants independent of n, by Lemma 3 we deduce that the spectrum of $T_n^{-1/2}(g_n)T_n(f_n)T_n^{-1/2}(g_n)$ is uniformly bounded away from zero and infinity (refer

to the constant C_2 in (2.1)). By invoking Lemma 2, we know that $\|T_n(f-f_n)\|_p \le \|f_n - f\|_p$ and by (2.2) we know that $\|T_n^{-1}(g_n)\|_\infty \le q_2\nu^\beta$. Therefore, by invoking the triangle inequality and (2.3) applied twice, we have

$$\begin{aligned} X_n &\le \|T_n^{-1/2}(g_n)T_n(f_n)T_n^{-1/2}(g_n)\|_p + \|T_n^{-1/2}(g_n)T_n(f-f_n)T_n^{-1/2}(g_n)\|_p \\ &\le C_2 + q_2\nu^\beta\|f_n - f\|_p. \end{aligned}$$

Moreover, since $T_n^{1/2}(g_n)T_n^{-1}(f)T_n^{1/2}(g_n)$ is positive definite, its spectral norm Y_n coincides with its maximal eigenvalue and since $T_n^{1/2}(g_n)T_n^{-1}(f)T_n^{1/2}(g_n)$ is similar to $T_n(g_n)T_n^{-1}(f)$ and the latter is similar to $T_n^{-1/2}(f)T_n(g_n)T_n^{-1/2}(f)$ which is positive definite, it follows that

$$\begin{aligned} Y_n &= \|T_n^{-1/2}(f)T_n(g_n)T_n^{-1/2}(f)\|_p \\ &\le \|T_n^{-1/2}(f)T_n(g)T_n^{-1/2}(f)\|_p + \|T_n^{-1/2}(f)T_n(g_n-g)T_n^{-1/2}(f)\|_p \\ &\le C_1 + q_1\nu^\alpha\|g_n - g\|_p, \end{aligned}$$

where in last inequality we have made recourse to Lemma 3 with the constant C_1 in (2.1), to Lemma 2, and to relations (2.2) and (2.3).

Finally we conclude that

$$\kappa_p\left(T_n^{-1/2}(g_n)T_n(f)T_n^{-1/2}(g_n)\right) \le [C_1 + q_1\nu^\alpha\|g_n - g\|_p]\left[C_2 + q_2\nu^\beta\|f_n - f\|_p\right]$$

and the proof is over. □

2.3. The robustness of the inexact band Toeplitz preconditioning. First we show that the error that we make in the evaluation of the zeros must be infinitesimal: if this is not the case, then the associated "inexact" band Toeplitz preconditioning is no longer optimal.

For the sake of simplicity, let us take a nonnegative essentially bounded function f with a unique zero of order 2α (α positive integer) at (x_0, y_0) and suppose that $g(x,y) = (2-2\cos(x-x_0))^\alpha + (2-2\cos(y-y_0))^\alpha$ and $\tilde g(x,y) = (2-2\cos(x-\tilde x_0))^\alpha + (2-2\cos(y-\tilde y_0))^\alpha$ with $(\tilde x_0, \tilde y_0) = (x_0, y_0) + (\delta_1, \delta_2)$. Here δ_1, δ_2 are fixed quantities independent of $n = (n_1, n_2)$ with $\|(\delta_1, \delta_2)\|_\infty \ne 0$. The following facts hold:

1. $\lim_{n\to\infty} \lambda_{\min}(T_n(f)) = 0$, $\lambda_{\max}(T_n(f)) \le \|f\|_\infty < \infty$;
2. $\lim_{n\to\infty} \lambda_{\min}(T_n^{-1}(\tilde g)T_n(f)) = 0$, $\lim_{n\to\infty} \lambda_{\max}(T_n^{-1}(\tilde g)T_n(f)) = \infty$,

where the claim in 1 is a consequence of Lemma 1 and the claim in 2 follows from the second part of Lemma 3. In actuality, by Lemma 3, the sequence $\{T_n^{-1}(\tilde g)T_n(f)\}$ is distributed as $h = f/\tilde g$ where h has a zero at (x_0, y_0) which is responsible for the relation $\lim_{n\to\infty} \lambda_{\min}(T_n^{-1}(\tilde g)T_n(f)) = 0$ and has a pole at $(\tilde x_0, \tilde y_0) = (x_0, y_0) + (\delta_1, \delta_2)$ which is responsible for the relation $\lim_{n\to\infty} \lambda_{\max}(T_n^{-1}(\tilde g)T_n(f)) = \infty$. In conclusion, it is self-evident that the associated PCG algorithm cannot be optimal and therefore we have to determine the root (x_0, y_0) within an error infinitesimal as n goes to infinity.

2.3.1. Isolated zeros. We proceed as follows. In full generality, let us suppose that f has $k+1$ zeros of even order $2\alpha_0, 2\alpha_1, \ldots, 2\alpha_k$ at $(x_0, y_0), (x_1, y_1), \ldots, (x_k, y_k)$ and let us take

$$g_n(x,y) = \prod_{j=0}^{k} [(2-2\cos(x-x_{j,n}))^{\alpha_j} + (2-2\cos(y-y_{j,n}))^{\alpha_j}]$$

with $\|(x_j, y_j) - (x_{j,n}, y_{j,n})\|_\infty \le \epsilon_\nu$, $n = (n_1, n_2)$, $n_1 \sim n_2 \sim \nu$. We wish to determine ϵ_ν such that

$$\kappa_\infty \left(T_n^{-1/2}(g_n) T_n(f) T_n^{-1/2}(g_n) \right) \le K, \tag{2.4}$$

where K is a universal constant independent of $n = (n_1, n_2)$.

By reading Theorem 1 with $p = \infty$ in our specific case, we obtain that

$$\begin{aligned} \kappa_\infty \left(T_n^{-1/2}(g_n) T_n(f) T_n^{-1/2}(g_n) \right) \le & \left[C_1 + q_1 \nu^{2\alpha_{\max}} \|g_n - g\|_\infty \right] \\ & \left[C_2 + q_2 \nu^{2\alpha_{\max}} \|f_n - f\|_\infty \right] \end{aligned} \tag{2.5}$$

with $n_1 \sim n_2 \sim \nu$. In the above equation the functions g and f_n have to be chosen in order to minimize $\|g_n - g\|_\infty$ and $\|f_n - f\|_\infty$ but with the constraint that $g \sim f$ and $f_n \sim g_n$. In such a way we hope to obtain a bound like the one in (2.4). An obvious proposal is represented by

$$\begin{aligned} g(x, y) &= g_n(x + (x_{j,n} - x_j), y + (y_{j,n} - y_j)), \\ f_n(x, y) &= f(x + (x_j - x_{j,n}), y + (y_j - y_{j,n})). \end{aligned} \tag{2.6}$$

It is evident that

$$g(x, y) = \prod_{j=0}^{k} [(2 - 2\cos(x - x_j))^{\alpha_j} + (2 - 2\cos(y - y_j))^{\alpha_j}],$$

$g \sim f$, and $f_n \sim g_n$; moreover a simple manipulation shows that

$$\|g - g_n\|_\infty \le \|\nabla g\|_\infty \epsilon_\nu \quad \text{and} \quad \|f - f_n\|_\infty \le L \epsilon_\nu^\gamma$$

if f is Hölder continuous with parameter $\gamma \in (0, 1]$ and constant $L > 0$. (In the regular case, e.g., if f is continuously differentiable, then f is Lipschitz continuous and therefore $L = \|\nabla f\|_\infty$ and $\gamma = 1$.) Here when we write $\|\nabla g\|_\infty$ and $\|\nabla f\|_\infty$, we mean the 1-norm of the vector composed by the infinity norms of the two components.

However, we can obtain a better estimate by choosing

$$g(x, y) = \begin{cases} g_n(x, y) & \text{if } \mathcal{A} \text{ holds true,} \\ g_n(x + (x_{j,n} - x_j), y + (y_{j,n} - y_j)) & \text{if } \mathcal{B} \text{ holds true,} \end{cases}$$

$$f_n(x, y) = \begin{cases} f(x) & \text{if } \mathcal{A} \text{ holds true,} \\ f(x + (x_j - x_{j,n}), y + (y_j - y_{j,n})) & \text{if } \mathcal{B} \text{ holds true} \end{cases} \tag{2.7}$$

with $\delta = 2\epsilon_\nu$, condition $\mathcal{A}$ being $\|(x, y) - (x_{j,n}, y_{j,n})\|_\infty > \delta \; \forall j = 0, \ldots, k$ and condition $\mathcal{B}$ being $\exists j \in \{0, \ldots, k\}$ such that $\|(x, y) - (x_{j,n}, y_{j,n})\|_\infty \le \delta$. In this way, due to the definition of order of zero, we have $\|g_n - g\|_\infty \le z_1 \epsilon_\nu^{2\alpha_{\min}}$ and $\|f_n - f\|_\infty \le z_2 \epsilon_\nu^{2\alpha_{\min}}$ for suitable absolute constants z_1 and z_2: beside the better approximation estimate, a surprising further advantage, when compared with the proposal in (2.6), is that we do not have to require any regularity to the function f. Furthermore, to apply the crucial relation (2.5) we have to verify that $f \sim g$ and $f_n \sim g_n$. This result is contained in the following theorem.

THEOREM 2. [illegible] f [illegible] $k + 1$ [illegible] $2\alpha_0, 2\alpha_1, \ldots, 2\alpha_k$ [illegible] $(x_0, y_0), (x_1, y_1), \ldots, (x_k, y_k)$ [illegible]

$$g_n(x, y) = \prod_{j=0}^{k} [(2 - 2\cos(x - x_{j,n}))^{\alpha_j} + (2 - 2\cos(y - y_{j,n}))^{\alpha_j}]$$

... $\|(x_j, y_j) - (x_{j,n}, y_{j,n})\|_\infty \le \epsilon_\nu$ $n = (n_1, n_2)$ $n_1 \sim n_2 \sim \nu$ ... f_n ... g ... (2.7) ... $f \sim g$ ... $f_n \sim g_n$ ... The proof of $f \sim g$ is equivalent to writing that $\frac{f}{g}$ and $\frac{g}{f}$ are uniformly bounded. Setting

$$g^*(x,y) = \prod_{j=0}^{k} \left[(2 - 2\cos(x - x_j))^{\alpha_j} + (2 - 2\cos(y - y_j))^{\alpha_j}\right],$$

by the assumptions on f, we have $\max\{\sup \frac{f(x,y)}{g^*(x,y)}, \sup \frac{g^*(x,y)}{f(x,y)}\} = C^* < \infty$. Therefore

$$\begin{aligned}
\sup \frac{f(x,y)}{g(x,y)} &= \max\left\{ \sup_{(x,y):\ \mathcal{A} \text{ holds}} \frac{f(x,y)}{g(x,y)}, \sup_{(x,y):\ \mathcal{B} \text{ holds}} \frac{f(x,y)}{g(x,y)} \right\} \\
&= \max\left\{ \sup_{(x,y):\ \mathcal{A} \text{ holds}} \frac{f(x,y)}{g(x,y)}, \sup_{(x,y):\ \mathcal{B} \text{ holds}} \frac{f(x,y)}{g^*(x,y)} \right\} \\
&\le \max\left\{ \sup_{(x,y):\ \mathcal{A} \text{ holds}} \frac{f(x,y)}{g(x,y)}, \sup \frac{f(x,y)}{g^*(x,y)} \right\} \\
&\le \max\left\{ \sup_{(x,y):\ \|(x,y)-(x_{j,n},y_{j,n})\|_\infty > 2\epsilon_\nu} \frac{f(x,y)}{g(x,y)}, C^* \right\}.
\end{aligned}$$

Now, by exploiting the asymptotical behavior of f and g in a neighborhood of the roots, we obtain

$$\sup \frac{f(x,y)}{g(x,y)} \le \sup_{(x,y):\ \|(x,y)-(x_{j,n},y_{j,n})\|_\infty > 2\epsilon_\nu} \hat{C} \frac{\|(x,y) - (x_j,y_j)\|_\infty^{2\alpha_{\max}}}{\|(x,y) - (x_{j,n},y_{j,n})\|_\infty^{2\alpha_{\max}}} + C^*$$

with $\hat{C}$ absolute constant independent of n. Hence by triangle inequality we deduce

$$\begin{aligned}
\sup \frac{f(x,y)}{g(x,y)} &\le \sup_{(x,y):\ \|(x,y)-(x_{j,n},y_{j,n})\|_\infty > 2\epsilon_\nu} \hat{C} \left[1 + \frac{\|(x_{j,n},y_{j,n}) - (x_j,y_j)\|_\infty}{\|(x,y) - (x_{j,n},y_{j,n})\|_\infty}\right]^{2\alpha_{\max}} + C^* \\
&\le \hat{C}(1.5)^{2\alpha_{\max}} + C^*,
\end{aligned}$$

where in the last inequality we have made recourse to the relations $\|(x_j, y_j) - (x_{j,n}, y_{j,n})\|_\infty \le \epsilon_\nu$ and $\|(x,y) - (x_{j,n}, y_{j,n})\|_\infty > 2\epsilon_\nu$.

A uniform bound (independent of n) is found along the same reasoning lines also for $\sup \frac{g(x,y)}{f(x,y)}$, $\sup \frac{f_n(x,y)}{g_n(x,y)}$ and $\sup \frac{g_n(x,y)}{f_n(x,y)}$ and therefore the theorem is proven. □

Finally, taking into account (2.5), the optimality is maintained i.e. (2.4) is satisfied if $\epsilon_\nu^{2\alpha_{\min}} \nu^{2\alpha_{\max}} = \mathcal{O}(1)$ which is equivalent to the relation $\epsilon_\nu = \mathcal{O}(\nu^{-\frac{\alpha_{\max}}{\alpha_{\min}}})$.

2.3.2. Curves of zeros. In the case of curves of zeros we suppose that f has $k+1$ regular curves of zeros of even order $2\alpha_0, 2\alpha_1, \ldots, 2\alpha_k$ at $(x_0(t), y_0(t)), (x_1(t), y_1(t)), \ldots, (x_k(t), y_k(t))$, $t \in [0,1]$, where $(x_j(t), y_j(t))$, $j = 0, \ldots, k$, is the solution of the implicit equation $l_j(x,y) = 0$. Let us take

$$g_n(x,y) = \prod_{j=0}^{k} (2 - 2\cos(l_{j,n}(x,y)))^{\alpha_j}$$

with $(x_{j,n}(t), y_{j,n}(t))$, $j = 0, \ldots, k$, being solution of the (approximate) implicit equation $l_{j,n}(x, y) = 0$, $\|(x_j(t), y_j(t)) - (x_{j,n}(t), y_{j,n}(t))\|_\infty \le \epsilon_\nu$, $n = (n_1, n_2)$, $n_1 \sim n_2 \sim \nu$. As in the case of isolated zeros the goal is to determine ϵ_ν such that

$$\kappa_\infty \left(T_n^{-1/2}(g_n) T_n(f) T_n^{-1/2}(g_n) \right) \le K, \tag{2.8}$$

where K is a universal constant independent of $n = (n_1, n_2)$. We consider Theorem 1 with $p = \infty$ in the latter context and then we have (2.5) with $n_1 \sim n_2 \sim \nu$ and where the functions g and f_n have to be chosen in order to minimize $\|g_n - g\|_\infty$ and $\|f_n - f\|_\infty$ but with the constraint that $g \sim f$ and $f_n \sim g_n$. In such a way, as in the case of the isolated zeros, we wish to obtain a bound as in (2.8). We consider the proposal described by

$$g(x, y) = \begin{cases} g_n(x, y) & \text{if } \mathcal{A} \text{ holds true,} \\ g_n(x + (x_{j,n} - x_j), y + (y_{j,n} - y_j)) & \text{if } \mathcal{B} \text{ holds true,} \end{cases} \tag{2.9}$$

$$f_n(x, y) = \begin{cases} f(x) & \text{if } \mathcal{A} \text{ holds true,} \\ f(x + (x_j - x_{j,n}), y + (y_j - y_{j,n})) & \text{if } \mathcal{B} \text{ holds true} \end{cases}$$

with $\delta = 2\epsilon_\nu$, condition $\mathcal{A}$ being $\|(x, y) - (x_{j,n}(t), y_{j,n}(t))\|_\infty > \delta \ \forall j = 0, \ldots, k$ and condition $\mathcal{B}$ being $\exists j \in \{0, \ldots, k\}$, $\exists \bar{t} \in [0, 1]$ such that $\|(x, y) - (x_{j,n}(\bar{t}), y_{j,n}(\bar{t}))\|_\infty \le \delta$. In this way, due to the definition of order of zero, we have $\|g_n - g\|_\infty \le z_1 \epsilon_\nu^{2\alpha_{\min}}$ and $\|f_n - f\|_\infty \le z_2 \epsilon_\nu^{2\alpha_{\min}}$ for suitable absolute constants z_1 and z_2: we stress that no regularity of the function f is explicitly required. Furthermore, to apply the crucial relation (2.5) we have to verify that $f \sim g$ and $f_n \sim g_n$. This result is contained in the following theorem.

THEOREM 3. [illegible] f [illegible] $k+1$ [illegible] $2\alpha_0, 2\alpha_1, \ldots, 2\alpha_k$ [illegible] $(x_0(t), y_0(t)), (x_1(t), y_1(t)), \ldots, (x_k(t), y_k(t))$ $t \in [0, 1]$ [illegible] $(x_j(t), y_j(t))$ $j = 0, \ldots, k$ [illegible] $l_j(x, y) = 0$ [illegible]

$$g_n(x, y) = \prod_{j=0}^{k} (2 - 2\cos(l_{j,n}(x, y)))^{\alpha_j}$$

[illegible] $(x_{j,n}(t), y_{j,n}(t))$ $j = 0, \ldots, k$ [illegible] $l_{j,n}(x, y) = 0$ $\|(x_j(t), y_j(t)) - (x_{j,n}(t), y_{j,n}(t))\|_\infty \le \epsilon_\nu$ $n = (n_1, n_2)$ $n_1 \sim n_2 \sim \nu$ [illegible] f_n [illegible] g [illegible] (2.9) [illegible] $f \sim g$ [illegible] $f_n \sim g_n$ [illegible] 2)

[illegible]. The proof is similar to that of Theorem 2. □

In view of (2.5), the optimality of the related PCG method is maintained, i.e., (2.4) is satisfied if $\epsilon_\nu^{2\alpha_{\min}} \nu^{2\alpha_{\max}} = \mathcal{O}(1)$ which is equivalent to the relation $\epsilon_\nu = \mathcal{O}(\nu^{-\frac{\alpha_{\max}}{\alpha_{\min}}})$. We remark that a trivial combination of Theorems 2 and 3 implies that the same conclusions hold in the case of combinations of isolated zeros and (regular) curves of zeros. Finally we have to emphasize a practical (and serious) difficulty which arises when curves of zeros are present: indeed, in that case, given the approximate curves of zeros, the proposed function $g_n(x, y)$ is not necessarily a trigonometric polynomial. Therefore the corresponding matrix $T_n(g_n)$ is dense as $T_n(f)$ and a possible but tricky solution is to approximate g_n by a polynomial of fixed degree: in some cases this idea can be followed but we believe that the answer in the general case is in the negative since, when we force ϵ_ν to be small enough, the degree of the polynomial g_n cannot be controlled in general by an absolute constant independent of $n = (n_1, n_2)$, $n_1 \sim n_2 \sim \nu$.

2.4. Comments on the use of the results. In the case where the zero is unique or in the case where all the zeros have the same order, it is sufficient to choose $\epsilon_n = \nu^{-1}$: this is a very reasonable requirement from a practical point of view. When the ratio $\frac{\alpha_{\max}}{\alpha_{\min}}$ is high (case of unbalanced zeros), the required precision on the computation of the zeros becomes very high. We can meet the requirement with an effort of $\mathcal{O}(N(n)\log(N(n)))$ flops by evaluating f as the nth Fourier sum, $n = (n_1, n_2)$. However, we must emphasize that we are guaranteed to be successful only if f is regular enough: we recall that the sup norm error of the Fourier sum is, up to a $\log \nu$ factor, $n_1 \sim n_2 \sim \nu$, of the same order of the best approximation error and the latter is $o(\nu^{-K})$ if f is K times continuously differentiable. Therefore in the case of just continuous functions and in the case where we desire that the approximation of f maintain the sign of function, then we can use the Rayleigh quotient approach considered in detail in Theorem 4.

THEOREM 4. [illegible] f [illegible] 2π [illegible] 2 [illegible] $\theta_x = \exp(\mathbf{i}x), \theta_y = \exp(\mathbf{i}y)$ [illegible]

$$\Theta_x^T = \frac{1}{\sqrt{n_1}}(1\ \theta_x\ \theta_x^2\ \cdots\ \theta_x^{n_1-1}) \qquad \Theta_y^T = \frac{1}{\sqrt{n_2}}(1\ \theta_y\ \theta_y^2\ \cdots\ \theta_y^{n_2-1}).$$

[illegible] $\Theta_{xy} = \Theta_x \otimes \Theta_y$ [illegible] $n = (n_1, n_2)$ [illegible]

$$r_n[f](x,y) := \frac{\Theta_{xy}^H T_n(f)\Theta_{xy}}{\Theta_{xy}^H\Theta_{xy}} = f(x,y) + \mathcal{O}\left(\max\left\{\frac{1}{n_1}, \frac{1}{n_2}\right\}\right). \tag{2.10}$$

[illegible] $f \in L^1(Q)$, $f \geq 0$ [illegible]

$$r_n[f](x,y) \geq 0 \qquad \forall\ (x,y) \in Q$$

[illegible] $r_n[f]$ [illegible] f [illegible]

[illegible]. Since $\Theta_{xy}^H\Theta_{xy} = 1$ by definition of Θ_{xy}, it follows that $r_n[f](x,y) = \Theta_{xy}^H T_n(f)\Theta_{xy}$. Moreover the map $r_n[\cdot]$, as an operator from $L^1(Q)$ into the space of the trigonometric bivariate polynomials, is linear and positive: indeed the linearity is evident since $T_n(\alpha g_1 + \beta g_2) = \alpha T_n(g_1) + \beta T_n(g_2)$ by the linearity of the Fourier coefficients; moreover if f is real valued, then $T_n(f)$ is Hermitian (see, e.g., [19]) and if f is nonnegative, then $T_n(f)$ is nonnegative definite (see, e.g., [19]). As a consequence, since Θ_{xy} has unit length, we infer that $r_n[f](x,y)$ is real valued if f is and is nonnegative if f is. Finally (see again [19]), if f is nonnegative and not identically zero (essential supremum of f strictly positive), then $T_n(f)$ is positive definite and $r_n[f](x,y)$ is a strictly positive function. (In the terminology of [24] the latter property means that the operator is linear and strongly positive.)

Now we prove relation (2.10). In what follows it is important to note that (x,y) is generic but fixed and (s,t) is the pair of dummy variables internal to the operator r_n. For instance, $r_n[f(s,t)](x,y)$ is the operator r_n applied to the function f and then evaluated at the point (x,y), while $r_n[f(x,y)](x,y) = f(x,y)\cdot r_n[1](x,y) = f(x,y)$ since $f(x,y)$ is a fixed constant and r_n is linear. Having this point in mind, we proceed with the proof.

From the regularity assumptions we know that

$$f(s,t) = f(x,y) + [\nabla f]^T(x,y)\begin{pmatrix} \sin(s-x) \\ \sin(t-y) \end{pmatrix} + C(s,t,x,y),$$

where $|C(s,t,x,y)| \leq K\left[\sin^2((s-x)/2)+\sin^2((t-y)/2)\right]$ with K pure constant only depending on the infinity norms of ∇f and H_f and with H_f denoting the Hessian of f. (For this kind of Taylor-style theorem in the periodic setting see [7] and references therein.) Now we perform the main step for proving the desired result which makes use (1) of the linearity and positivity of the operator, (2) of the above Taylor style representation, (3) of the simple identities (related to the Korovkin test; see subsection 4.1.1 in [22]) $r_n[1](x,y) \equiv 1$, $r_n[\sin(s)](x,y) \equiv \sin(x)+\mathcal{O}(\frac{1}{n_1})$, $r_n[\sin(t)](x,y) \equiv \sin(y)+\mathcal{O}(\frac{1}{n_2})$, $r_n[\cos(s)](x,y) \equiv \cos(x)+\mathcal{O}(\frac{1}{n_1})$, $r_n[\cos(t)](x,y) \equiv \cos(y)+\mathcal{O}(\frac{1}{n_2})$, and (4) of the boundedness of the second derivative H_f: indeed calling $E_{n,f}(x,y) = |r_n[f(s,t)](x,y)-f(x,y)|$ the approximation error, we have

$$\begin{aligned}
E_{n,f}(x,y) &=_{(3)} |r_n[f(s,t)](x,y)-r_n[f(x,y)](x,y)| \\
&=_{\text{linearity}} |r_n[f(s,t)-f(x,y)](x,y)| \\
&=_{(2)} \left| r_n\left[[\nabla f]^T(x,y)\begin{pmatrix}\sin(s-x)\\ \sin(t-y)\end{pmatrix}+C(s,t,x,y)\right](x,y)\right| \\
&=_{\text{linearity}} \left|\frac{\partial f}{\partial x}(x,y) r_n[\sin(s-x)](x,y)+\frac{\partial f}{\partial y}(x,y) r_n[\sin(t-y)](x,y)\right. \\
&\quad \left. + r_n\left[C(s,t,x,y)\right](x,y)\right| \\
&\leq_{\text{positivity}} \left|\frac{\partial f}{\partial x}(x,y)\right| |r_n[\sin(s-x)](x,y)|+\left|\frac{\partial f}{\partial y}(x,y)\right| |r_n[\sin(t-y)](x,y)| \\
&\quad + r_n\left[|C(s,t,x,y)|\right](x,y) \\
&\leq_{\text{regularity}} \|\nabla f\|_\infty \left[|r_n[\sin(s-x)](x,y)|+|r_n[\sin(t-y)](x,y)|\right] \\
&\quad + K r_n\left[\sin^2((s-x)/2)+\sin^2((t-y)/2))\right](x,y).
\end{aligned}$$

Finally the claimed thesis is equivalent to proving that

$$\begin{aligned}
&|r_n[\sin(s-x)](x,y)|+|r_n[\sin(t-y)](x,y)| \\
&\quad + r_n\left[\sin^2((s-x)/2)+\sin^2((t-y)/2))\right](x,y) = \mathcal{O}\left(\max\left\{\frac{1}{n_1},\frac{1}{n_2}\right\}\right)
\end{aligned}$$

and, taking into account the identities $\sin^2((s-x)/2) = (1-\cos(s-x))/2$, $\sin^2((t-y)/2)) = (1-\cos(t-y))/2$, the latter is proven if we prove that

$$|r_n[\sin(s-x)](x,y)| = \mathcal{O}\left(\max\left\{\frac{1}{n_1},\frac{1}{n_2}\right\}\right), \tag{2.11}$$

$$|r_n[\sin(t-y)](x,y)| = \mathcal{O}\left(\max\left\{\frac{1}{n_1},\frac{1}{n_2}\right\}\right), \tag{2.12}$$

$$r_n[1-\cos(s-x)](x,y) = \mathcal{O}\left(\max\left\{\frac{1}{n_1},\frac{1}{n_2}\right\}\right), \tag{2.13}$$

$$r_n[1-\cos(t-y)](x,y) = \mathcal{O}\left(\max\left\{\frac{1}{n_1},\frac{1}{n_2}\right\}\right). \tag{2.14}$$

By the explicit computation in the banded case (taking into account that $r_n[f]$ is the Césaro sum of f [23, 22]), it is easy to verify that relationships (2.11)–(2.14) hold.

Here, for the sake of completeness, we just give a detailed look at the last one:

$$\begin{aligned} r_n[1-\cos(t-y)](x,y) &=_{\text{trigonometric identity}} r_n[1-\cos(t)\cos(y)-\sin(t)\sin(y)](x,y)\\ &=_{\text{linearity}} r_n[1](x,y)-\cos(y)r_n[\cos(t)](x,y)\\ &\quad -\sin(y)r_n[\sin(t)](x,y)\\ &=_{(3)} 1-\cos(y)\left[\cos(y)+\mathcal{O}\left(\frac{1}{n_2}\right)\right]\\ &\quad -\sin(y)\left[\sin(y)+\mathcal{O}\left(\frac{1}{n_2}\right)\right]=\mathcal{O}\left(\frac{1}{n_2}\right). \quad \square \end{aligned}$$

Regarding the Fourier expansion $F_{n_1-1,n_2-1}(f)$ of f of degree n_1-1 and n_2-1, we observe that the latter approximation is much faster when f is very smooth (due to its Lebesgue constant of order $\log(n_1)+\log(n_2)$), but may fail to converge when f is only continuous. On the other hand, thanks to the Korovkin theory, the Rayleigh quotient approximation always converges when f is continuous and preserves the sign of f, but its order of approximation is not sensitive to the regularity of f. We point out that these two types of approximation have been the main theoretical tools for the construction of our banded preconditioners (see [17]).

3. Numerical examples and related discussions. As a preliminary step, we consider two numerical examples which give evidence of the main ideas described in Theorem 2 and Theorem 3 (whose result was essentially conjectured in Proposition 1 of [17]): a good approximation of the "exact" trigonometric polynomial $g(x,y)$ leads to a controlled number of spectral outliers laying outside the main clustering mass described by the range of $\frac{f}{g}$.

1. We consider the Toeplitz matrix $T_n(f)$ produced by the generating function $f(x,y)=(1+x^2+y^2)(2\cos(x)-\sin(x+2y))^2$. It is obvious that the trigonometric polynomial $g(x,y)=(2\cos(x)-\sin(x+2y))^2$ has an infinite number of roots which form a curve of roots. For the solution of the system $T_n(f)\mathrm{x}=b$ by a PCG iteration we use as preconditioner the 2-level band Toeplitz matrix $T_n(\tilde{g})$ instead of $T_n(g)$, where $\tilde{g}(x,y)=(2\cos(x-\epsilon)-\sin(x+2y+\epsilon))^2$ is an approximation of $g(x,y)$. In Table 3.1 we show the strict relation existing between the approximation error and the number of outlying eigenvalues. It is observed that, only when $n_1\epsilon=n_2\epsilon$ exceeds 1, there exist eigenvalues of the preconditioned matrix that lie outside the range of $\frac{f}{g}$.

2. We consider the Toeplitz matrix $T_n(f)$ generated by $f(x,y)=(3+\cos(x))(4-2\cos(x)-2\cos(y))(2-2\cos(x-\pi))^2$. It is obvious that it has a zero of order 2 at $(0,0)$ and a line of zeros at $x=\pi$ of order 4. We make a theoretical check on the use of the preconditioner $T_n(\tilde{g})$, where $\tilde{g}(x,y)=((4-2\cos(x+\epsilon)-2\cos(y+\epsilon))(2-2\cos(x-\pi-\epsilon))^2$ is an approximation of $g(x,y)$ with $g(x,y)=(4-2\cos(x)-2\cos(y))(2-2\cos(x-\pi))^2$. In Table 3.2 we report the tight relationship between the approximation error and the number of outlying

TABLE 3.1
Number of outliers for $T_n^{-1}(\tilde{g})T_n(f)$, *range*$\left(\frac{f}{g}\right)=[1\ ,\ 20.73921]$.

ϵ	$n_1=n_2$	out	$\epsilon n_1=\epsilon n_2$	$\lambda_{\min}$	$\lambda_{\max}$	ϵ	out	$\epsilon n_1=\epsilon n_2$	$\lambda_{\min}$	$\lambda_{\max}$
.05	8	0	$.4<1$	1.26	18.83	.01	0	$.08<1$	1.21	17.9
	16	0	$.8<1$	1.11	20.2		0	$.16<1$	1.03	19.3
	32	9	$1.6>1$	1.07	27.43		0	$.32<1$	1.02	19.91
	64	61	$3.2>1$	0.52	60.18		0	$.64<1$	1.01	20.5

TABLE 3.2
Number of outliers for $T_n^{-1}(\tilde{g})T_n(f)$, *range*$\left(\frac{f}{g}\right) = [2\ ,\ 4]$.

ϵ	$n_1 = n_2$	out	$\epsilon n_1^2 = \epsilon n_2^2$	$\lambda_{\min}$	$\lambda_{\max}$	ϵ	out	$\epsilon n_1^2 = \epsilon n_2^2$	$\lambda_{\min}$	$\lambda_{\max}$
	8	13	$3.2 > 1$	1.9	4.42		0	$.06 < 1$	2.28	3.96
	16	52	$12.8 > 1$	1.42	5.15		0	$.25 < 1$	2.1	3.99
.05	32	225	$51.2 > 1$	0.95	5.5	.001	8	$1.02 > 1$	2.01	4.03
	64	931	$204.8 > 1$	0.64	8.87		177	$4.09 > 1$	1.95	4.01

eigenvalues. It can be noted that, only if $n_1^2\epsilon = n_2^2\epsilon$ exceeds 1, then there exist eigenvalues of the preconditioned matrix that lie outside the range of $\frac{f}{g}$: we stress that this case falls into the one of nonbalanced zeros for which we should expect $\epsilon_\nu \sim \nu^{-2}$, $\nu = n_1 = n_2$ since $\frac{\alpha_{\max}}{\alpha_{\min}} = 2$. Therefore the suggestion from this example is that the bounds found in Theorems 2 and 3 cannot be improved in general. (Possibly, a local analysis of the Toeplitz operators could help refine the analysis in the case of special sets of symbols with nonbalanced zeros.)

In the rest of the section we follow a procedure proposed in [17] which, starting from the knowledge of the entries of $T_n(f)$, is able to determine an approximation of the generating function f (with its zeros and related orders) and consequently gives all the necessary information for determining the appropriate preconditioner. We first present and complete a sketch of this procedure and then we discuss various numerical experiments to test the effectiveness of the proposed method.

The outline of the procedure in [17] (which was based on a general scheme proposed and analyzed in [23, 29], respectively) is given in the following steps:

- Step 1: Approximate the function f from the coefficients of the matrix.
- Step 2: Search for the possible roots of f.
- Step 3: Estimate the multiplicities of each root.
- Step 4: Categorize the roots and choose the appropriate preconditioner.

We observe that these steps can be regarded as a collection of ideas that eventually have the potential to lead to a real black-box procedure for the analysis, the detection and the effective preconditioning of ill-conditioned 2-level Toeplitz systems. Here we go in this direction by describing in detail a part of Step 4 concerning the case of curves of zeros, i.e., how to construct a polynomial factor having approximately the same zeros of f (curves of zeros and isolated zeros). We stress that this subroutine (i.e., Algorithm 1 reported on the next page) is essential for the effectiveness of the whole procedure and has been ignored so far in the relevant literature. Indeed, looking at Theorem 3, we see that the approximation of the curves of the zeros is not sufficient for determining the symbol of the precondition since what we really need is the approximation $l_{j,n}(x,y)$ of every implicit function $l_j(x,y)$, $j = 0,\dots,k$, which (implicitly) defines the right curve of zeros $(x_j(t), y_j(t))$, $t \in [0,1]$, $j = 0,\dots,k$.

However, as emphasized in the remarks following Theorem 3, the approximate factor $2 - 2\cos(l_{j,n}(x,y))$ and even the theoretical one $2 - 2\cos(l_j(x,y))$ in general are not trigonometric polynomials. Therefore, that nontrigonometric symbol $2 - 2\cos(l_{j,n}(x,y))$ or $2 - 2\cos(l_j(x,y))$ would correspond to a useless preconditioner owing to the denseness of preconditioner itself (which would make the method not optimal according to Axelsson and Neytcheva [2], since the cost of every iteration would be much higher than the cost of a matrix-vector product). As a consequence, in Algorithm 1, we follow a computational approach described in the following pseudocode (written in a pseudo high-level language) for the direct computation of the (approximate) polynomial factor (if any) containing all the zeros of f. To estimate

the polynomial factor we use least-square approximation (LSA). Since we want to have good approximation on the selected roots, we choose the information of all the points of the set

$$\mathcal{G}_n = \left\{ (x_j^{(n_1)}, y_j^{(n_2)}) \in \mathcal{S}_n, \text{ computed roots of } f \text{ on } \mathcal{S}_n, \ j = 1, \ldots, \tilde{\theta}_n \right\},$$

$\mathcal{S}_n$ being a $n_1 \times n_2$ uniform gridding on $Q = (-\pi, \pi]^2$ (see (1.1)). It is well known from the above theory that the computed values of f on these points, by Rayleigh quotient, are $\mathcal{O}(\max\{\frac{1}{n_1}, \frac{1}{n_2}\})$ instead of 0. So, these values contain a significant error. We could correct those by putting 0 but then the LSA procedure would give us a polynomial which takes negative values. To avoid this, we give positive values for f of order $\mathcal{O}(\max\{\frac{1}{n_1}, \frac{1}{n_2}\})$ (the same values for all points are acceptable). In addition, it is not sufficient to apply LSA only on the points of $\mathcal{G}_n$. It is necessary to choose some points which give enough information for the reconstruction of the shape of f. For this we choose a coarse grid $\mathcal{S}_{\hat{n}}$, $\hat{n} = (\hat{n}_1, \hat{n}_2)$, where $\hat{n}_1 << n_1$ and $\hat{n}_2 << n_2$ ($\hat{n}_1 \approx \sqrt{n_1}$, $\hat{n}_2 \approx \sqrt{n_2}$ are enough), and create all the points of $\mathcal{S}_{\hat{n}}$ which are added to the points of $\mathcal{G}_n$ for the LSA method. Let θ_n be the number of the selected points ($\theta_n = |\mathcal{G}_n| + |\mathcal{S}_{\hat{n}}|$) and $\hat{f}$ be the θ_n-dimensional vector of the approximated values of f, as these are described before. Let also $q = (q_1, q_2)$, $q_1, q_2 \geq 0$, $q_1 + q_2 \geq 1$ be the pair of orders of the trigonometric polynomial that we have to approximate in each direction. Then, we have to solve the LSA problem $\min_{\mathbf{t}} \|\hat{f} - V_q \mathbf{t}\|_2^2$, where V_q is the rectangular Vandermonde matrix defined by

$$((V_q)_{j,k})_{j=1,\ldots,\theta_n, k=1,\ldots,(2q_1+1)(2q_2+1)}$$

with $(V_q)_{j,k} = \psi_k(x_j^{(n_1)}, y_j^{(n_2)})$ and $\psi_k(x, y) = \exp(\mathbf{i}\alpha x + \mathbf{i}\beta y)$. Here the relation between the index $k \in \{1, \ldots, (2q_1+1)(2q_2+1)\}$ and the 2-index $(\alpha, \beta) \in \{-q_1, \ldots, q_1\} \times \{-q_2, \ldots, q_2\}$ is defined through the bijection $k := k(\alpha, \beta, q) = (\alpha + q_1)(2q_2 + 1) + \beta + q_2 + 1$, $q = (q_1, q_2)$. To apply LSA it is necessary to have $\theta_n \geq (2q_1 + 1)(2q_2 + 1)$; otherwise we choose additional points on the fine grid, near the estimated roots, in order to have more information about the roots, until the above inequality is satisfied.

ALGORITHM 1. APPROXIMATING FACTOR.

```
b :=false;
for q_1 = 1 : q_1^max do
    for q_2 = 1 : q_2^max do
        Minimize ||f̂ − V_q t||_2^2;
        construct the polynomial p_t(x, y) = Σ_k t_k ψ_k(x, y);
        if max_{(x,y)∈G_n} |p_t(x, y)| < ε_ν then
            b :=true;
        end if
    end for
end for
if b =true then
    write: "The approximating factor is optimal";
else
    write: "The approximating factor does not insure optimality";
end if
factor: p_t(x, y);
```

Here given $q_1^{\max} > 0$, $q_2^{\max} > 0$, the procedure will compute a mask $\mathbf{t} = \mathbf{t}(q)$ of Fourier coefficients with $q = (q_1, q_2)$, $0 \le q_1 \le q_1^{\max}$, $0 \le q_2 \le q_2^{\max}$, $q_1 + q_2 \ge 1$.

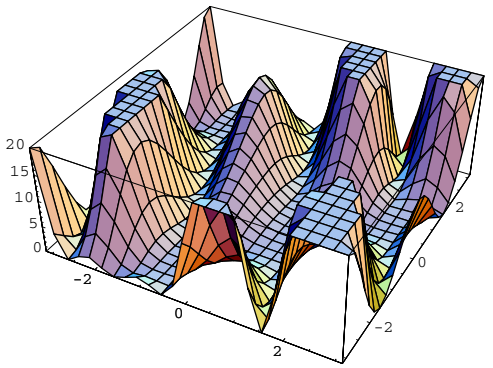
EXAMPLE 3. In order to test our procedure we solve

$$T_n(f)\mathrm{x} = b,$$

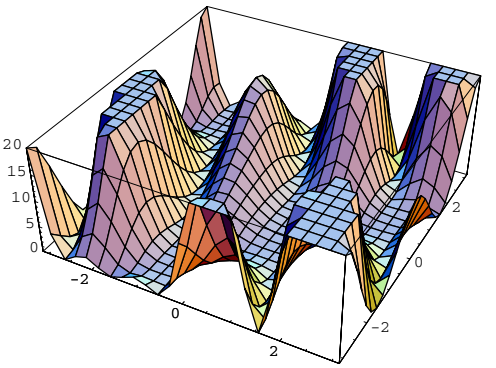
where the generating function is $f(x,y) = (x^2+y^2+1/4)(\cos(x+y)+\sin(2x))^2$ and has five curves of roots in the domain $Q = (-\pi, \pi]^2$ as is shown clearly in Figure 3.1(a). The knowledge of the generating function has been used only for the computation of the coefficients of $T_n(f)$. In what follows we calculate the grid $\mathcal{G}_n$ and we estimate the possible roots of f as well as their multiplicities. According to the reasoning in the previous sections, we set ϵ_ν as $.06, .03, .015, 7.5 * 10^{-3}, 3.75 * 10^{-3}$ for $\nu = n_1 = n_2 = 16, 32, 64, 128, 256$, respectively. For $\nu = 8$ we do not have enough information (points) to set up a reasonable good LSA scheme. We mention here that the cost of LSA is dominated by that of PCG since the number of columns of the overdetermined problem depends on the partial degrees q_1, q_2 of the desired trigonometric polynomial rather the dimensions n_1, n_2 of the original problem. The maximum permitted order of the approximated trigonometric polynomial is restricted by the dimension of the problem. More specifically, knowing that the cost of the solution of a band block Toeplitz system of size ν^2 $(n_1 = n_2 = \nu)$ and total bandwidth l can be done in $O(\nu^2 l^2)$ arithmetic operations, the total degree, which roughly speaking coincides with the total bandwidth, cannot exceed $\mathcal{O}(\log \nu)$. The stopping criterion of our algorithm is $p_t(x_i, y_i) < \epsilon_\nu$, where (x_i, y_i) are the estimated roots and $\epsilon_\nu = \mathcal{O}(\frac{1}{\nu})$ or the degree of the approximated polynomial is larger than $c \log \nu$ with suitable c. In practice, and for various examples, the second condition happens more often. This occurrence may lead to some eigenvalues which lay outside the range of $\frac{f}{g}$, where g is the trigonometric polynomial of minimal order which just raises the roots of f. On the other hand, by using LSA not only in the points that have been estimated as roots, we obtain a global approximation of the unknown function f and therefore we can gain a better clustering of the preconditioned system than that achieved by g. This observation is confirmed in Table 3.3, where we denote by I the unpreconditioned CG method, by $\hat{P}$ the PCG method derived from our procedure, and by P the optimal PCG method. Although the algorithm stopped because the second criterion was satisfied, the required iterations by the optimal preconditioner and the ones needed by our proposal are almost the same, and for some dimensions the global behavior is even better. We mention that the comparison with the optimal preconditioner has only theoretical meaning since, in the practical situation, we do not know the generating function and, as a consequence, the generating function of the exact preconditioner is not available. For the PCG method we used as an initial guess the null vector, as b the vector of all ones, and $\frac{\|r_k\|_2}{\|r_0\|_2} \le 10^{-5}$ as a stopping criterion, where r_k is the residual vector after k iterations.

TABLE 3.3
Iterations of the PCG method for $f(x,y) = (x^2 + y^2 + 1/4)(\cos(x+y) + \sin(2x))^2$.

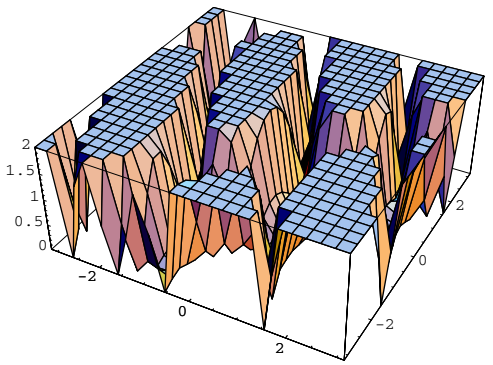
$\nu = n_1 = n_1$	I	$\hat{P}$	P
8	48	-	25
16	146	22	37
32	396	41	44
64	978	52	50
128	*	56	54
256	*	59	57



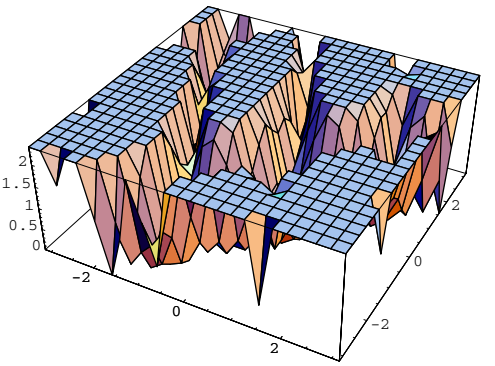
(a) The generating function in (0,20)



(b) The approximated trigonometric polynomial in (0,20)



(c) The generating function in (0,2)



(d) The approximated trigonometric polynomial in (0,2))

FIG. 3.1. *The function f and its approximated trigonometric polynomial in* $[0,2]$ *and* $[0,20]$.

In Figure 3.1 we give two snapshots of the graphs of the unknown generating function f and the trigonometric polynomial $\hat{p}$ that we have estimated using our algorithm when $\nu = 32$ and the order of $p_{\mathbf{t}}$ is $(5,5)$. More precisely, in Figures 3.1(a) and 3.1(b) we observe the global approximation features of our scheme, while in 3.1(c) and 3.1(d) we concentrate our attention on comparing the exact generating function f and the approximate polynomial $p_{\mathbf{t}}$ in the domains where they attain small values: we recall that is the good approximation in such domains, which is crucial for a good performance of the associated preconditioner. In both cases, it is clear that a very good approximation has been achieved by the procedure.

. . . . 1. For functions like $x^2 + (y-1)^4$ that need different treatment in each variable, our four-step algorithm can estimate very accurately the root $(0,1)$ as well as the multiplicities 2 in the x direction and 4 in the y direction (see Example 3 in [17]).

. . . . 2. Furthermore, it is worth commenting on the case where f has zeros of not even orders (not integer as well): in that case even the exact band Toeplitz

TABLE 3.4
Maximum, minimum eigenvalues and iterations of the four possible preconditioned schemes.

$n_1 = n_2$	$\lambda_{\max}(P_{2,4}^{-1}T)$	$\lambda_{\min}(P_{2,4}^{-1}T)$	$Iter_{2,4}$	$\lambda_{\max}(P_{2,6}^{-1}T)$	$\lambda_{\min}(P_{2,6}^{-1}T)$	$Iter_{2,6}$
8	13.9	.72	15	6.23	.8	10
16	15.9	.34	28	6.88	.38	17
32	17.4	.16	56	7.26	.18	33
$n_1 = n_2$	$\lambda_{\max}(P_{4,4}^{-1}T)$	$\lambda_{\min}(P_{4,4}^{-1}T)$	$Iter_{4,4}$	$\lambda_{\max}(P_{4,6}^{-1}T)$	$\lambda_{\min}(P_{4,6}^{-1}T)$	$Iter_{4,6}$
8	14	.98	15	3.63	.99	8
16	15.9	.8	24	4.04	.97	11
32	17.8	.5	39	4.76	.96	15

preconditioning (error free with respect to the position of the zeros) is no longer optimal. However, in the 1-dimensional setting, it has been proved that in the worst case (refer to [20]) the number of PCG iterations can grow at most as $\nu^{1/2}$, ν being the size of the 1-level Toeplitz matrix, which can be reasonably good for very ill-conditioned positive definite Toeplitz systems. It is remarked here that the proof of the 1-level case can be extended in the 2-level case. The obtained result is that in the worst case the number of PCG iterations can grow at most as $\nu^{1/2}$, $\nu \sim n_1 \sim n_2$. To show this we consider $f(x,y) = |x|^3 + |y|^5$ which has the isolated zero $(0,0)$ with multiplicities $(3,5)$. By applying our four-step algorithm we will get an isolated zero $(x_0, y_0) \approx (0,0)$ with the associated multiplicities $(k_1, k_2) \approx (3,5)$. If we approximate k_1 and k_2 by the closest even integers we could get four possible choices: $(2,4)$, $(2,6)$, $(4,4)$, or $(4,6)$. We describe in detail the first possibility $(2,4)$. The associated trigonometric polynomial is $2 - 2\cos(x) + (2 - 2\cos(y))^2$ and the condition number of the preconditioned matrix can be estimated by studying the limit of $\frac{|x|^3+|y|^5}{2-2\cos(x)+(2-2\cos(y))^2}$ as $(x,y) \to (0,0)$. In this case a simple analysis in polar coordinates implies that $\kappa(T_n^{-1}(g)T_n(f)) = \mathcal{O}(\nu)$. By using the tools introduced in this paper, we get $\kappa(T_n^{-1}(\tilde{g})T_n(f)) = \mathcal{O}(\nu)$, where $\tilde{g} = 2-2\cos(x+\epsilon_1)+(2-2\cos(y+\epsilon_2))^2$, $\epsilon_1 \sim \epsilon_2 \sim \frac{1}{\nu}$. Hence the number of PCG iterations grows as $\nu^{1/2}$. This is shown in Table 3.4, where we consider the above four choices. We can observe in this table that the choice $(4,6)$ is better than the one of $(4,4)$; then $(2,6)$ follows and the worst choice is $(2,4)$. An explanation of the aforementioned phenomenon is contained in the classical PCG convergence theory due to Axelsson and Lindskög [1]. In the overestimated choices, the eigenvalues of the preconditioned matrix are far away from zero but some of them grow to infinity since $\frac{f}{g}$ is far away from zero but unbounded while in the underestimated ones, they are uniformly bounded but some of them tend to zero since $\frac{f}{g}$ is bounded but has a zero at $(0,0)$. Now, from [1], it is well known that small eigenvalues disturb the PCG convergence more than big eigenvalues and, as a consequence, it is a general recommendation to overestimate the multiplicities of the roots rather than underestimate them when the involved numbers are not even integers.

3. As a conclusion, we remark here that the above theory could be extended in the d-dimensional case where $d \geq 3$. The difference is that we may have isolated roots, curves of roots, surfaces of roots, or hypersurfaces of roots. It is possible to state and prove spectral properties for the d-dimensional case analogous to the ones given in this paper for the 2-dimensional case. The difficulty is on the practical side. It is more complicated to construct an efficient algorithm which would recognize the roots, their nature (isolated, curves, surfaces, or hypersurfaces), and

their multiplicities. On the other hand, concerning the applications, d-level ($d \geq 3$) Toeplitz systems arise from several discretization schemes for elliptic d-dimensional PDEs. In that case, the symbol f is known (or at least the position and the order of the zeros is known) and therefore the exact trigonometric polynomial which deletes the roots can be determined analytically.

REFERENCES

[1] O. Axelsson and G. Lindskög, *The rate of convergence of the preconditioned conjugate gradient method*, Numer. Math., 52 (1986), pp. 499–523.

[2] O. Axelsson and M. Neytcheva, *The algebraic multilevel iteration methods—theory and applications*, in Proceedings of the Second International Colloquium on Numerical Analysis, Plovdiv, Bulgaria, D. Bainov, ed., 1993, pp. 13–23.

[3] A. Böttcher and B. Silbermann, *Introduction to Large Truncated Toeplitz Operators*, Springer, New York, 1999.

[4] B. Buzbee, G. Golub, and C. Nielson, *On direct methods for solving Poisson's equation*, SIAM J. Numer. Anal., 7 (1970), pp. 727–756.

[5] B. Buzbee, F. Dorr, J. George, and G. Golub, *The direct solutions of the discrete Poisson equation on irregular regions*, SIAM J. Numer. Anal., 8 (1971), pp. 722–736.

[6] R. H. Chan and M. Ng, *Conjugate gradient method for Toeplitz systems*, SIAM Rev., 38 (1996), pp. 427–482.

[7] F. Costabile, M. Gualtieri, and S. Serra Capizzano, *The approximation of continuous periodic functions by means of extrapolation techniques for the De La Vallée Poussin operator*, in Proceedings of the Ninth International Colloquium on Differential Equations, Plovdiv, Bulgaria, D. Bainov, ed., 1999, pp. 112–119.

[8] F. Di Benedetto, G. Fiorentino, and S. C. G. Serra Capizzano, *Preconditioning for Toeplitz matrices*, Comput. Math. Appl., 26 (1993), pp. 35–45.

[9] G. Fiorentino and S. Serra Capizzano, *Multigrid methods for symmetric positive definite block Toeplitz matrices with nonnegative generating functions*, SIAM J. Sci. Comput., 17 (1996), pp. 1068–1081.

[10] S. Holmgren, S. Serra-Capizzano, and P. Sundqvist, *Can one hear the composition of a drum?*, Mediterr. J. Math., to appear.

[11] X. Q. Jin, *Developments and Applications of Block Toeplitz Iterative Solvers*, Kluwer Academic Publishers, Beijing, 2003.

[12] M. Ng, *Band preconditioners for block-Toeplitz–Toeplitz-block systems*, Linear Algebra Appl., 259 (1997), pp. 307–327.

[13] M. Ng, *Iterative Methods for Toeplitz Systems*, Oxford University Press, Oxford, UK, 2004.

[14] D. Noutsos, S. Serra Capizzano, and P. Vassalos, *Spectral equivalence and matrix algebra preconditioners for multilevel Toeplitz systems: A negative result*, in Structured Matrices in Mathematics, Engineering and Computer Sciences, V. Olshevsky, ed., Contemp. Math. 323, AMS, Providence, RI, 2003, pp. 313–322.

[15] D. Noutsos, S. Serra Capizzano, and P. Vassalos, *Matrix algebra preconditioners for multilevel Toeplitz systems do not insure optimal convergence rate*, Theoret. Comput. Sci., 315 (2004), pp. 557–579.

[16] D. Noutsos, S. Serra Capizzano, and P. Vassalos, *Asymptotic behavior of the condition number of two-level Toeplitz matrix sequences*, Linear Algebra Appl., 395 (2005), pp. 121–140.

[17] D. Noutsos, S. Serra Capizzano, and P. Vassalos, *Preconditioning algorithm for ill-conditioned Hermitian block Toeplitz systems*, Numer. Linear Algebra Appl., 12 (2005), pp. 231–239.

[18] D. Potts and G. Steidl, *Preconditioners for ill-conditioned Toeplitz systems constructed from positive kernels*, SIAM J. Sci. Comput., 22 (2001), pp. 1741–1761.

[19] S. Serra Capizzano, *Preconditioning strategies for asymptotically ill-conditioned block Toeplitz systems*, BIT, 34 (1994), pp. 326–337.

[20] S. Serra Capizzano, *New PCG based algorithms for the solution of Hermitian Toeplitz systems*, Calcolo, 32 (1995), pp. 153–176.

[21] S. Serra Capizzano, *On the extreme eigenvalues of Hermitian (block) Toeplitz matrices*, Linear Algebra Appl., 270 (1997), pp. 109–129.

[22] S. Serra Capizzano, *A Korovkin-type theory for finite Toeplitz operators via matrix algebras*, Numer. Math., 82 (1999), pp. 117–142.

[23] S. Serra Capizzano, *How to choose the best iterative strategy for symmetric Toeplitz systems*, SIAM J. Numer. Anal., 36 (1999), pp. 1078–1103.
[24] S. Serra Capizzano, *Some theorems on linear positive operators and functionals and their applications*, Comput. Math. Appl., 39 (2000), pp. 139–167.
[25] S. Serra Capizzano, *Matrix algebra preconditioners for multilevel Toeplitz matrices are not superlinear*, Linear Algebra Appl., 337 (2001), pp. 37–78.
[26] S. Serra Capizzano, *Test functions, growth conditions and Toeplitz matrices*, Rend. Circ. Mat. Palermo., 68 (2002), pp. 791–795.
[27] S. Serra Capizzano, *An ergodic theorem for classes of preconditioned matrices*, Linear Algebra Appl., 282 (2002), pp. 161–183.
[28] S. Serra Capizzano, *Convergence analysis of two grid methods for elliptic Toeplitz and PDEs matrix sequences*, Numer. Math., 92 (2002), pp. 433–465.
[29] S. Serra Capizzano, *Practical band Toeplitz preconditioning and boundary layer effects*, Numer. Algorithms, 34 (2003), pp. 427–440.
[30] S. Serra Capizzano and P. Tilli, *On unitarily invariant norms of matrix valued linear positive operators*, J. Inequal. Appl., 7 (2002), pp. 309–330.
[31] S. Serra Capizzano and E. Tyrtyshnikov, *Any circulant-like preconditioner for multilevel matrices is not superlinear*, SIAM J. Matrix Anal. Appl., 21 (1999), pp. 431–439.
[32] S. Serra Capizzano and E. Tyrtyshnikov, *How to prove that a preconditioner can not be superlinear*, Math. Comput., 72 (2003), pp. 1305–1316.
[33] P. Tilli, *Some results on complex Toeplitz eigenvalues*, J. Math. Anal. Appl., 239 (1999), pp. 390–401.
[34] C. F. Van Loan, *Computational Frameworks for the Fast Fourier Transform*, SIAM, Philadelphia, 1992.

SIAM J. SCI. COMPUT.
Vol. 28, No. 2, pp. 459–484
© 2006 Society for Industrial and Applied Mathematics

CENTROIDAL VORONOI TESSELLATION-BASED REDUCED-ORDER MODELING OF COMPLEX SYSTEMS*

JOHN BURKARDT†, MAX GUNZBURGER†, AND HYUNG-CHUN LEE‡

Abstract. A reduced-order modeling methodology based on centroidal Voronoi tessellations (CVTs) is introduced. CVTs are special Voronoi tessellations for which the generators of the Voronoi diagram are also the centers of mass (means) of the corresponding Voronoi cells. For discrete data sets, CVTs are closely related to the h-means and k-means clustering techniques. A discussion of reduced-order modeling for complex systems such as fluid flows is given to provide a context for the application of reduced-order bases. Then, detailed descriptions of CVT-based reduced-order bases and how they can be constructed from snapshot sets and how they can be applied to the low-cost simulation of complex systems are given. Subsequently, some concrete incompressible flow examples are used to illustrate the construction and use of CVT-based reduced-order bases. The CVT-based reduced-order modeling methodology is shown to be effective for these examples.

Key words. reduced-order modeling, Voronoi diagrams, incompressible flows, clustering

AMS subject classifications. 65M99, 35B30, 58F39, 93C20, 93A30, 76D05

DOI. 10.1137/5106482750342221x

1. Introduction.

Even with the use of good mesh generators, discretization schemes, and solution algorithms, the computational simulation of complex, turbulent, or chaotic systems still remains a formidable endeavor. For example, typical finite element codes may require many thousands of degrees of freedom for the accurate simulation of fluid flows. The situation is even worse for optimization problems for which multiple solutions of the complex state system are usually required or in feedback control problems for which real-time solutions of the complex state system are needed. The need for so many degrees of freedom results from two causes. First, phenomena occurring over very small spatial and temporal scales have to be resolved. For example, flows at even moderate values of the Reynolds number are usually turbulent. In a direct computational simulation, very small eddies have to be accurately resolved, even if one is only interested in the large-scale features of the flow. Second, the functions or grid stencils used to effect approximation have no direct relation to the solution of the complex state system. For example, the standard, locally supported, piecewise polynomial basis functions used to define a finite element approximation of the flow solution are not specially designed for a particular flow or for flows in general. Thus, in a typical finite element calculation for a turbulent flow, many thousands of degrees of freedom are needed in order to properly resolve the small eddies.

The natural question to ask is whether or not it is really necessary to have thousands of degrees of freedom in order to produce accurate approximations to solutions of complex systems. For example, for turbulent flows, one would like to identify highly persistent spatio-temporal structures using relatively inexpensive, low-dimensional approaches instead of using expensive standard techniques in the numerical solution

*Received by the editors January 31, 2003; accepted for publication (in revised form) November 3, 2005; published electronically March 31, 2006.
http://www.siam.org/journals/sisc/28-2/42221.html

†School of Computational Science and Information Technology, Florida State University, Tallahassee FL 32306-4120 (burkardt@csit.fsu.edu, gunzburg@csit.fsu.edu). The work of the second author was supported by the National Science Foundation under grant number CCR-9988303.

‡Department of Mathematics, Ajou University, Suwon 443-749, Korea (hclee@ajou.ac.kr). The work of this author was supported by KOSEF R01-2000-000-00008-0.

of partial differential equations.[1] The feedback control of complex systems provides another example; in this setting, low-dimensional state models are needed so that actuation can be determined quickly from sensed data. As a result, there have been many studies devoted to the development, testing, and use of reduced-order models for complex systems such as unsteady fluid flows.

The types of reduced-order models that we study are those that attempt to determine accurate approximate solutions of a complex system using very few degrees of freedom. To do so, such models have to use basis functions that are in some way intimately connected to the problem being approximated. Once a very low-dimensional reduced basis has been determined, one can employ it to solve the complex system by applying, e.g., a Galerkin method. In general, reduced bases are globally supported so that the discrete systems are dense; however, if the reduced basis is of very low dimension, one does not care about the lack of sparsity in the discrete system. Of course, the first natural question to ask is: How does one determine a reduced basis of very low dimension?

There have been many reduced-order modeling techniques proposed. In the structural and fluid mechanics communities, Taylor polynomial and Lagrange polynomial based methods were studied in, e.g., [23, 31] and were shown to be effective for some specific problems. In the Lagrange polynomial approach, one determines a low-dimensional basis by solving the complex system for several values of the parameters that appear in the specification of the system or at several instants in time during the evolution of the system. In the Taylor polynomial approach, the low-dimensional basis consists of the solution of the complex system and of its derivatives with respect to the parameters evaluated at fixed values of the parameters. In both cases, one uses the reduced basis to determine the solution of the complex system at other values of the parameters and, in both cases, the reduced basis is determined by employing high-dimensional discretizations that are, of course, very costly. In simulation settings, that cost is hopefully amortized over subsequent calculations for many other parameter values using the low-cost reduced basis approach. Likewise, in control settings, the cost is amortized over the many state solves necessary to determine the controls.

Today, the most popular reduced-order modeling approach for complex systems is based on proper orthogonal decomposition (POD) analysis. POD begins with a set of $\widetilde{m}$ snapshots; these could be generated by evaluating the computational solution of a transient problem at several instants of time or by evaluating the computational solution for several values of the parameters appearing in the problem description or by a combination of the two. The snapshot set can even be determined, in principle, from experiments. The computational solutions that are used to determine the snapshot set are determined using costly, large-scale, high-fidelity, e.g., finite element, codes. The $\widetilde{d}$-dimensional POD basis is then determined from the $\widetilde{d}$ eigenvectors corresponding to the $\widetilde{d}$ most dominant eigenvalues of the correlation matrix for the snapshots.[2] The POD basis is then used, usually by applying a projection procedure,

[1] The whole field of turbulence modeling, e.g., k-ϵ and large-eddy simulation (LES) models, is an attempt at reduced-order modeling. However, these approaches still result in complex systems of partial differential equations that are usually solved by employing very high-dimensional approximation schemes. The approaches to reduced-order modeling considered here are different and result in very low-dimensional models for complex systems.

[2] Perhaps more simply, a totally equivalent definition of the $\widetilde{d}$-dimensional POD basis is the first $\widetilde{d}$ left singular vectors of the matrix whose columns are the snapshot vectors. In statistical analyses, POD is known as the Karhunen–Loève analysis or the method of empirical orthogonal eigenfunctions or principal component analysis.

to determine an approximate solution for different values of the system parameters. For example, in an optimal control or optimization setting, one would generate the POD basis from snapshots that correspond to several sets of values of the optimization or control parameters. These snapshots are determined using expensive, e.g., finite element, codes having many thousands of degrees of freedom. However, the snapshots can be generated "off-line" and their costs can be hopefully amortized over several optimization calculations. When one performs the optimization or control of the system, one uses the small-dimensional POD basis to determine the multiple state solutions and any sensitivity or adjoint equations solutions needed by the optimizer or by the controller. The POD basis possesses a well-known optimization characteristic. POD-based model reduction has been applied with some success to several problems, most notably in fluid mechanics settings. For detailed discussions, one may consult [1, 2, 3, 4, 5, 15, 17, 18, 20, 21, 22, 25, 26, 27, 28, 29, 30, 32, 33, 34, 35, 36, 37, 38, 41, 42]. A few more details about how snapshots are generated and how a POD basis is used for model reduction are also given in section 1.1.

Now that we have discussed two approaches for generating reduced bases, another natural question to ask is when can one expect a reduced-order model to "work," i.e., to produce accurate, very low-dimensional approximations to a solution of a complex system. Clearly, they can work only when the low-dimensional basis "contains" all of the information needed to produce an accurate approximation, i.e., in an interpolatory setting or in the very limited settings where extrapolation can work. In the POD setting, the snapshots used to define the POD basis must contain all the information, e.g., the dynamic behavior, of the solution one is trying to obtain. The beauty of POD, when it works, is that it can reduce a set of hundreds of snapshots to a set of just a handful of POD basis vectors that contain essentially the same information. However, when the snapshot set does not contain the needed information, one has to generate new snapshots and then generate a new POD basis, often in an adaptive manner. This can happen when one uses a reduced-basis method in an optimization or control setting. Initially, the snapshots, and therefore the POD basis, can be constructed to contain the needed information, but as the optimizer or controller changes the parameters in the problem, it is often the case that the original snapshot set does not contain enough information to accurately approximate the solution at the new values of the parameters, and thus the snapshot set has to be augmented with new snapshots that reflect the new behavior. Even worse, it may be the case that the POD basis, which included most of the information contained in the snapshot set for the parameters used to generate the snapshots, ignored information in the snapshot set that may be useful for other values of the parameters. Furthermore, redoing the POD basis when the snapshot set changes requires the solution of a new eigenvalue or singular value problem which is of the size of the cardinality of the snapshot set.

Centroidal Voronoi tessellations (CVTs) have been successfully used in several data compression settings, e.g., in image processing and the clustering of data.[3] Reduced-order modeling of complex systems is another data compression setting, i.e., replacing high-dimensional approximations with low-dimensional ones. Thus, one can

[3]CVTs are useful in many other settings, e.g., just to name some, optimal quadrature rules, optimal representation and quantization, cell division, data compression, optimal distribution of resources, territorial behavior of animals, optimal placement of sensors and actuators, grid generation in two and three dimensions and on surfaces, and meshfree methods. For further discussions about the applications of CVTs, see, e.g., [6, 7, 8, 9, 10, 11, 12, 13, 14, 19, 24]. Also, like POD, CVT is closely related to well-known statistical techniques; in fact, for discrete data sets, CVT can be identified with the h-means and k-means clustering methods.

ask if CVTs can be used for the reduced-order modeling of complex systems. In CVT-reduced order modeling, we start with a snapshot set just as is done in a POD-based setting. However, instead of determining a POD basis from the snapshot set, we apply our CVT methodologies to determine the generators of a CVT of the snapshot set; these generators constitute the reduced-order basis. We then use the CVT-based basis in just the same way as one uses a POD-based basis to determine a very low-dimensional approximation to the solution of a complex system. CVT also possesses an optimality property, although it is different from that possessed by POD bases.

The efficiency of CVT- and POD-based reduced bases depends on the their dimension, i.e., if a CVT- or POD-based basis is low-dimensional and can still approximate the state well, then approximations of the state of a complex system can be inexpensively determined. However, the ability of a CVT- or POD-based basis to approximate the state of a system is totally dependent on the information contained in the snapshot set used to generate the basis. Certainly, a CVT- or POD-based basis cannot contain more information than that contained in the snapshot set.[4] Thus, crucial to the success of both the CVT- and POD-based approaches to model reduction is the generation of "good" snapshot sets.

Unfortunately, at this time, the generation of snapshot sets is an art and not a science; it is, in fact, a rather primitive art. One recognizes that the generation of snapshot sets is an exercise in the design of experiments. For example, for stationary systems, how does one choose the sets of parameters at which the state and perhaps sensitivities are calculated using expensive, high-fidelity computations in order to generate the snapshot set? Clearly, some a priori knowledge about the types of states to be simulated or optimized using the reduced-order model is very useful in this regard. The large body of statistics literature on the design of experiments has not been used in a systematic manner. For time-dependent systems, many ad hoc measures have been invoked in the hope that they will lead to good snapshot sets. Time-dependent parameters, e.g., appearing in boundary conditions, are used to generate states that are "rich" in transients, even if the state of interest depends only on time-independent parameters. Moreover, in order to generate even "richer" dynamics, impulsive forcing is commonly used, e.g., starting the evolution impulsively with different strength impulses or introducing impulses in the midst of a simulation. In section 3, we adopt some of these strategies for the generation of snapshot sets in the concrete examples we consider. However, generating snapshot sets is not a main focus of this paper. It is clear, though, that in the future, a great deal of effort needs to be directed towards developing and justifying methodologies for generating good snapshot sets. After all, a POD or CVT basis can only be as good as the snapshot set used to generate it.

1.1. Schematic of POD- or CVT-based model reduction for simulation problems. For the sake of concreteness, we suppose that we wish to solve the problem:

$$\text{given } \theta \in \Theta, \text{ find } u(t,x) \in U \text{ such that } N(u;\theta;v) = 0 \ \forall\, v \in U, \ \text{a.e. } t \in (0,T)\,, \tag{1.1}$$

where U and Θ are given solution[5] and parameter function spaces, respectively, and

[4] Recall that the goal of invoking POD or CVT is merely to determine a low-dimensional basis that "contains" almost all the information contained in the snapshot set.

[5] In general, the spaces of test and trial functions are not the same, i.e., in (1.1) the test functions v do not, in general, belong to the same space as do the trial functions u. For the sake of simplicity of exposition, we here assume that they are the same.

$(0,T)$ is a specified time interval.[6] If (1.1) represents a time-dependent problem, then it should be supplemented with initial conditions.[7] The mapping $N : U \times \Theta \times V$ is linear in its third argument and is generally nonlinear in its first two arguments; $N(\cdot;\cdot;\cdot)$ in general involves space and time derivatives and integrals of combinations of its arguments. In (1.1), θ denotes a set of parameters that serve to specify the problem, u denotes the desired solution, and v denotes a suitable test function. Thus, (1.1) implicitly defines a solution mapping $u(\theta) : \Theta \to U$. The object of a computational simulation is to find approximations of this mapping.

For the sake of even further concreteness, we consider finite element methods for the approximation of solutions of (1.1). To this end, one chooses a finite-dimensional subspace[8] $U^h \subset U$ parameterized by a parameter h tending to zero (e.g., the grid size) and then one poses the approximate problem:

$$\begin{aligned} &\text{given } \theta \in \Theta, \text{ find } u^h(t,x) \in U^h \text{ such that} \\ &N(u^h;\theta;v^h) = 0 \quad \forall\, v^h \in U^h, \text{ a.e. } t \in (0,T)\,. \end{aligned} \tag{1.2}$$

The approximate solution is computed by choosing a basis $\{\phi_k(x)\}_{k=1}^{\widetilde{j}}$, then writing $u^h(t,x) = \sum_{k=1}^{\widetilde{j}} U_k(t)\phi_k(x)$, and then solving the problem:[9]

$$\begin{aligned} &\text{given } \theta \in \Theta, \text{ find } U_k(t),\ k = 1\ldots,\widetilde{j}, \text{ such that} \\ &N\left(\textstyle\sum_{k=1}^{\widetilde{j}} U_k\phi_k;\theta;\phi_j\right) = 0 \quad \text{for } j = 1,2,\ldots,\widetilde{j},\ t \in (0,T)\,. \end{aligned} \tag{1.3}$$

In general, for complex systems of practical interest, the dimension $\widetilde{j}$ of the approximating space U^h may be in the many thousands. As a result, it may be very expensive to solve (1.3) for a single set θ of parameter values and it may be prohibitively expensive to solve (1.3) for many sets of parameter values. The latter is exactly the situation which causes the interest in reduced-order modeling so that we assume that we want to find approximations of the solutions of (1.1) for many sets of parameter values. We now describe the steps that go into defining and using POD- or CVT-type reduced-order models for finding approximations to the solutions of (1.1).

1.1.1. Generating snapshots. We begin by choosing several parameter sets $\theta_1, \theta_2, \ldots, \theta_{\widetilde{\ell}}$ and then solving (1.3) for each set, i.e., we solve the $\widetilde{\ell}$ problems:

$$\begin{aligned} &\text{for } \ell = 1,\ldots,\widetilde{\ell},\ \text{find } U_k^\ell,\ k = 1\ldots,\widetilde{j}, \text{ such that} \\ &N\left(\textstyle\sum_{k=1}^{\widetilde{j}} U_k^\ell\phi_k;\theta_\ell;\phi_j\right) = 0 \quad \text{for } j = 1,2,\ldots,\widetilde{j}\,. \end{aligned} \tag{1.4}$$

Note that, in general, for each ℓ, (1.4) represents the solution of a large nonlinear system of ordinary differential equations, or in the stationary case, a large system of

[6] Using (1.1) as our prototype complex system prejudices our discussion towards Galerkin-finite element and Galerkin-spectral methods. However, one could easily extend our discussion to other methods, including finite difference and finite volume methods.

[7] If (1.1) represents a stationary problem, we simply suppress all dependencies on t; for the sake of concreteness, we will assume that (1.1) represents a time-dependent problem.

[8] One may also need to approximate the parameter space in which case one also has to choose an approximating parameter subspace $\Theta^h \subset \Theta$. However, for the sake of simplicity of the exposition, we assume that this is not necessary for the problem we consider. It is also possible that the mapping N has to be replaced by an approximate mapping N^h, but we ignore that possibility as well. We note that there is no difficulty extending our discussion to the more general cases.

[9] So far, we have only effected spatial discretization. If the mapping N involves time derivatives of u, then the problem (1.3) is a system of nonlinear ordinary differential equations (ordinary differential equations) for $U_k(t),\ k = 1\ldots,\widetilde{j}$. Thus, further discretization in time is usually needed to solve (1.3).

discrete nonlinear equations. Therefore, generating snapshots is an expensive proposition.

Next, we choose several time values $t_1, t_2, \ldots, t_{\widetilde{q}}$ within the interval $[0, T]$ and we evaluate the coefficients $U_k^\ell(t)$, $k = 1 \ldots, \widetilde{j}$ and $\ell = 1, \ldots, \widetilde{\ell}$, at these time values. Using this information, we define the $\widetilde{m} = \widetilde{q}\widetilde{\ell}$

$$\vec{U}_m = \begin{pmatrix} U_1^\ell(t_q) \\ U_2^\ell(t_q) \\ \vdots \\ U_{\widetilde{j}}^\ell(t_q) \end{pmatrix}, \quad m = (\ell - 1)\widetilde{q} + q, \quad q = 1 \ldots, \widetilde{q}, \text{ and } \ell = 1, \ldots, \widetilde{\ell}. \tag{1.5}$$

Note that the snapshot vectors $\vec{U}_m \in \mathbb{R}^{\widetilde{j}}$, where $\widetilde{j}$ is the dimension of the finite element space being employed which may be in the many thousands. On the other hand, the number of snapshots $\widetilde{m} = \widetilde{q}\widetilde{\ell}$ is usually taken to be of the order of hundreds.

The process just described for generating snapshots is a well defined recipe except that it requires the choosing of $\widetilde{\ell}$ parameter sets $\theta_1, \ldots, \theta_{\widetilde{\ell}}$ and $\widetilde{q}$ values of time $t_1, \ldots, t_{\widetilde{q}}$. As has already been mentioned, how these choices are made is not something that is a systematic science. In section 3, we provide some examples of how these choices are made.

There are definitions other than (1.5) that have been used to define snapshot sets. For example, one could add to the snapshots in (1.5) approximations to the time derivatives of the coefficients $U_k^\ell(t)$, i.e., in addition to (1.5), we have the snapshot vectors

$$\vec{U}_m \approx \begin{pmatrix} \left.\frac{dU_1^\ell}{dt}\right|_{t_q} \\ \vdots \\ \left.\frac{dU_{\widetilde{j}}^\ell}{dt}\right|_{t_q} \end{pmatrix}, \quad m = \widetilde{\ell}\widetilde{q} + (\ell - 1)\widetilde{q} + q, \quad q = 1 \ldots, \widetilde{q}, \text{ and } \ell = 1, \ldots, \widetilde{\ell}$$

so that now we have a set of snapshots of cardinality $\widetilde{m} = 2\widetilde{\ell}\widetilde{q}$. The approximation to the time derivatives may be effected, e.g., by using finite difference quotients.

1.1.2. Generating reduced bases from the snapshot set. Snapshot sets by themselves are not always directly useful for reduced-order modeling. First, in order to include "enough" useful information about solutions of complex systems, one usually has on the order of hundreds of snapshot vectors which is too high a number for practicality. Second, snapshot sets may be linearly dependent or may be poorly conditioned as a basis so that, again, they may not be practical. On the other hand, snapshot sets often contain much redundant information so that it should be possible to define smaller basis sets of much lower cardinality that are well conditioned and that have nearly the same information content as the snapshot set. POD and CVT analyses can be applied for this purpose.

Before applying POD or CVT analyses, the snapshot set is usually modified. For example, one usually subtracts the mean of the snapshot set before further processing, e.g., instead of using (1.5), one uses the modified snapshots in which the mean of the snapshots is subtracted from every snapshot:

$$\vec{V}_m = \vec{U}_m - \overline{U} \quad \text{for } m = 1, \ldots, \widetilde{m}, \qquad \text{where} \qquad \overline{U} = \frac{1}{\widetilde{m}} \sum_{m=1}^{\widetilde{m}} \vec{U}_m \,.$$

In addition, snapshot vectors are sometimes manipulated so as to efficiently account for inhomogeneous boundary conditions. In any case, we assume that we now have in hand a satisfactory snapshot set $\mathcal{V} = \{\vec{V}_m\}_{m=1}^{\widetilde{m}}$ from which we want to define a low-dimensional reduced basis.

Let us briefly consider POD-based reduced bases. There are many equivalent ways to define POD bases corresponding to a given snapshot set. Perhaps the simplest definition is as follows. First, we form the snapshot matrix

$$S = (\vec{V}_1, \vec{V}_2, \cdots, \vec{V}_{\widetilde{m}}),$$

i.e., the matrix whose columns are the snapshot vectors; the ordering does not matter. Note that S is a $\widetilde{j} \times \widetilde{m}$ matrix so that, in practice, it has many more rows than columns. Then, the $\widetilde{d}$-dimensional POD basis consists of the first $\widetilde{d}$ left singular vectors of S. Note that the POD basis vectors belong to $\mathbb{R}^{\widetilde{j}}$ and that the POD basis is perfectly conditioned since, by definition, it is an orthonormal basis. It has also been found in many settings that POD bases of low dimension, i.e., of the order of ten or so, contain almost all the information present in snapshot sets consisting of hundreds of vectors.

Instead of POD, in this paper we consider using CVT analyses to generate a low-dimensional reduced basis from a given set of snapshots. This process is described in detail in section 2.

1.1.3. Using a reduced basis to solve a simulation problem. Suppose now that we have in hand a $\widetilde{d}$-dimensional reduced basis $\mathcal{Z} = \{\vec{Z}_i\}_{i=1}^{\widetilde{d}}$, be it POD-based or CVT-based. The basis vectors $\vec{Z}_i$ belong to $\mathbb{R}^{\widetilde{j}}$ and consist of the coefficients of particular (globally supported) functions belonging to the finite element trial space U^h. In fact, if we write

$$\vec{Z}_i = \begin{pmatrix} Z_{i,1} \\ Z_{i,2} \\ \vdots \\ Z_{i,\widetilde{j}} \end{pmatrix} \qquad \text{for } i = 1, \ldots, \widetilde{d},$$

the corresponding finite element functions are given by

$$z_i(x) = \sum_{j=1}^{\widetilde{j}} Z_{i,j}\phi_j(x) \qquad \text{for } i = 1, \ldots, \widetilde{d}, \tag{1.6}$$

where $\{\phi_j(x)\}_{j=1}^{\widetilde{j}}$ is the basis for the finite element space U^h, and the $\widetilde{d}$-dimensional reduced basis can be written in the form $\{z_i(x)\}_{i=1}^{\widetilde{d}}$. The reduced-order space $Z^h = \text{span}\,\{z_1(x), \ldots, z_{\widetilde{d}}(s)\} \subset U^h$. Note that $\dim Z^h = \widetilde{d}$ which is very small compared to the dimension $\widetilde{j}$ of the finite element space used to define the discrete problem (1.2).

Having obtained a reduced basis, we now want to use it to determine an approximate solution of (1.1) for new parameter sets θ. We use exactly the same process that led to (1.2) and (1.3). Since we now seek an approximate solution $u_{ro}^h(t,x)$ belonging to the reduced-order space Z^h for all $t \in (0,T)$, we solve the problem:

$$\begin{aligned} &\text{given } \theta \in \Theta, \text{ find } u_{ro}^h(t,x) \in Z^h \text{ such that} \\ &N(u_{ro}^h; \theta; v^h) = 0 \quad \forall\, v^h \in Z^h, \text{ a.e. } t \in (0,T). \end{aligned} \tag{1.7}$$

Since $u^h_{ro}(t,x) \in Z^h$, we can write $u^h_{ro}(t,x) = \sum_{i=1}^{\widetilde{d}} c_i(t) z_i(x)$ for some time-dependent coefficients c_i, $i = 1, \ldots, \widetilde{d}$. Then, (1.7) is equivalent to the problem of finding the coefficients $c_i(t)$, $i = 1, \ldots, \widetilde{d}$, from the nonlinear system of ordinary differential equations:

$$\begin{array}{l} \text{given } \theta \in \Theta, \text{ find } c_i(t),\ i = 1 \ldots, \widetilde{d}, \text{ such that} \\ N\left(\sum_{i=1}^{\widetilde{d}} c_i z_i; \theta; z_j\right) = 0 \quad \text{for } j = 1, 2, \ldots, \widetilde{d},\ t \in (0,T)\,. \end{array} \tag{1.8}$$

The reduction in size between (1.3) and (1.8) is now explicitly evident. The former is a system of $\widetilde{j}$ nonlinear ordinary differential equations for the $\widetilde{j}$ unknowns $U_k(t)$, $k = 1, \ldots, \widetilde{j}$, while (1.8) is a system of $\widetilde{d}$ ordinary differential equations for the $\widetilde{d}$ unknowns $c_i(t)$, $i = 1, \ldots, \widetilde{d}$. Thus, having produced the reduced basis $\{z_i(x)\}_{i=1}^{\widetilde{d}}$, we can now solve (1.8) for any given parameter set θ much more cheaply than if we resorted to solving (1.3).

Using (1.6), we see that the solution of (1.7) or, equivalently, of (1.8) can be written as

$$u^h_{ro}(t,x) = \sum_{k=1}^{\widetilde{j}} \left(\sum_{i=1}^{\widetilde{d}} c_i(t) Z_{i,k} \right) \phi_k(x) \tag{1.9}$$

so that clearly $u^h_{ro} \in U^h$, i.e., the solution of (1.7) lives in the finite element space. However, only $\widetilde{d}$ degrees of freedom (the c_i's) completely determine $u^h_{ro} \in U^h$. If we substitute (1.9) into (1.7) or, equivalently, if we substitute (1.6) into (1.8), we arrive at another equivalent form for the reduced-order problem:

$$\begin{array}{l} \text{given } \theta \in \Theta, \text{ find } c_i(t),\ i = 1 \ldots, \widetilde{d}, \text{ such that} \\ N\left(\sum_{k=1}^{\widetilde{j}} \left(\sum_{i=1}^{\widetilde{d}} c_i Z_{i,k}\right) \phi_k(x); \theta; \sum_{j=1}^{\widetilde{j}} Z_{m,j} \phi_j(x)\right) = 0 \\ \text{for } m = 1, 2, \ldots, \widetilde{d},\ t \in (0,T)\,. \end{array} \tag{1.10}$$

This form of the reduced-order problem is written in terms of the standard finite element functions ϕ_j and the components of the reduced basis vectors $Z_{i,j}$; it is the form that is actually used in practice to assemble the $\widetilde{d}$ ordinary differential equations corresponding to the reduced-order problem.

To complete the definition of the reduced-order problem, one must choose a solution method for the system of ordinary differential equations (1.10).

We have now given an almost complete recipe for how a reduced basis is determined and used to find low-order approximations of complex systems; the missing step is how a reduced basis is determined from the snapshot set. We briefly described how this step is carried out in the POD setting, and in the next section, we describe how it is done in the CVT setting.

2. Centroidal Voronoi tessellation-based reduced bases. The generation of a CVT-based reduced-order basis is the intermediate step of the model reduction algorithm. It follows the generation of the snapshot set, creating a set of basis vectors which are used in the subsequent step to determine approximations of the solution of the complex system for new physical states, i.e., new sets of parameters. We now provide a definition of a CVT-based reduced basis and then discuss how they are constructed.

2.1. Definition of centroidal Voronoi tessellations for discrete data sets. The definition of centroidal Voronoi tessellations (CVTs) for discrete data sets begins with a set $\mathcal{V} = \{\vec{V}_m\}_{m=1}^{\widetilde{m}}$ consisting of $\widetilde{m}$ vectors belonging to $\mathbb{R}^{\widetilde{j}}$. Of course, $\mathcal{V}$ can also be viewed as a set of $\widetilde{m}$ points in $\mathbb{R}^{\widetilde{j}}$. The definition also requires us to introduce the notion of clustering or tessellation, which in our context is defined to be a subdivision of the set $\mathcal{V}$ into $\widetilde{d} \le \widetilde{m}$ subsets $\mathcal{V}_i$, $i = 1, \dots \widetilde{d}$, such that $\mathcal{V}_i \cap \mathcal{V}_j = \emptyset$ for $i \neq j$ and $\cup_{i=1}^{\widetilde{d}} \mathcal{V}_i = \mathcal{V}$. The set of subsets $\{\mathcal{V}_i\}_{i=1}^{\widetilde{d}}$ is called a [illegible] or a [illegible] of the set $\mathcal{V}$. One way to determine a clustering is to first choose a set $\mathcal{Z} = \{\vec{Z}_i\}_{i=1}^{\widetilde{d}}$ consisting of $\widetilde{d} \le \widetilde{m}$ distinct vectors,[10] also belonging to $\mathbb{R}^{\widetilde{j}}$, and then choose a function $C(\vec{V}) = \vec{Z}$ that assigns each $\vec{V}_m \in \mathcal{V}$ to a particular $\vec{Z}_i \in \mathcal{Z}$, and then letting $\mathcal{V}_i = \{\vec{V}_m \in \mathcal{V} \text{ such that } C(\vec{V}_m) = \vec{Z}_i\}$ for $i = 1, \dots, \widetilde{d}$. We call the collection $(\mathcal{V}, C, \mathcal{Z})$ a [illegible]. The vectors $\vec{Z}_i$ are called [illegible] and the subsets $\mathcal{V}_i$ are called [illegible]. The cardinality of cluster $\mathcal{V}_i$ is denoted by $\overline{m}_i$; clearly, $\sum_{i=1}^{\widetilde{d}} \overline{m}_i = \widetilde{m}$, the cardinality of the set $\mathcal{V}$. The vectors

$$\overline{Z}_i = \frac{1}{\overline{m}_i} \sum_{\vec{V}_m \in \mathcal{V}_i} \vec{V}_m \qquad \text{for } i = 1, \dots, \widetilde{d}$$

are called the [illegible] or [illegible].

For a given set $\mathcal{V}$, a [illegible] or [illegible] of $\mathcal{V}$ is a particular type of cluster configuration for which the cluster generators $\{\vec{Z}_i\}_{i=1}^{\widetilde{d}}$ may be chosen arbitrarily and for which the clustering rule C assigns each $\vec{V}_m \in \mathcal{V}$ to the nearest[11] $\vec{Z}_i \in \mathcal{Z}$:

$$C(\vec{V}_m) = \vec{Z}_i \quad \Longrightarrow \quad |\vec{V}_m - \vec{Z}_i| = \min_{\vec{Z}_j \in \mathcal{Z}} |\vec{V}_m - \vec{Z}_j| \,. \tag{2.1}$$

Given a given set $\mathcal{V}$ of $\widetilde{m}$ vectors in $\mathbb{R}^{\widetilde{j}}$ and a positive integer $\widetilde{d} \le \widetilde{m}$, a [illegible] (CVT) or [illegible] of $\mathcal{V}$ is a special Voronoi tessellation satisfying

$$\vec{Z}_i = \overline{Z}_i \,, \tag{2.2}$$

i.e., the generators of the Voronoi tessellation coincide with the centroids of the corresponding Voronoi clusters. It is important to note that general Voronoi tessellations do not satisfy the CVT property (2.2) so that, for given a set $\mathcal{V}$ and positive integer $\widetilde{d}$, a CVT must be constructed. Algorithms for this purpose are discussed in section 2.2.

CVTs possess an optimization property that can be used as a basis for an alternate definition. Given a given set $\mathcal{V}$ of $\widetilde{m}$ vectors in $\mathbb{R}^{\widetilde{j}}$, a positive integer $\widetilde{d} \le \widetilde{j}$, and a clustering configuration $(\mathcal{V}, C, \mathcal{Z})$, let

$$E(\mathcal{V}, C, \mathcal{Z}) \equiv \sum_{m=1}^{\widetilde{m}} |\vec{V}_m - C(\vec{V}_m)|^2 = \sum_{m=1}^{\widetilde{d}} \sum_{\vec{V}_m \in \mathcal{V}_i} |\vec{V}_m - \vec{Z}_i|^2 \tag{2.3}$$

[10] The vectors in the data set $\mathcal{V}$ need not be distinct.

[11] Of course, there may be more than one vector in $\mathcal{Z}$ that is nearest a $\vec{V}_m$ so that a tie-breaking rule has to be appended to (2.1). For example, among all vectors in $\mathcal{Z}$ that satisfy (2.1), we could choose the one with the smallest index value.

denote [illegible]" Then, it can be shown (see [6]) that among all possible assignment rules $C(\cdot)$ and all possible cluster generators $\{\vec{Z}_i\}_{i=1}^{\widetilde{d}}$, i.e., among all possible cluster configurations, cluster energy minimizers are CVT cluster configurations.

One easily recognizes that (2.3) defines the variance of the clustering so that, as described here, CVTs are minimum variance clusterings of discrete point sets. Thus, in the context discussed here, CVTs are well known in the statistical literature under names like k-means clustering. However, CVTs can be generalized in many ways, e.g., to continuous sets, to other metrics, to weighted energy functionals, etc. Since the latter generalization can be useful in the model reduction setting, we will briefly discuss it in section 4.

The connection between CVTs and reduced bases is now easily made. The set $\mathcal{V}$ is obviously the set of (possibly modified) snapshots. Then, the CVT reduced basis set is the set of generators $\mathcal{Z} = \{\vec{Z}_i\}_{i=1}^{\widetilde{d}}$ of a CVT of $\mathcal{V}$.

2.2. Algorithms for constructing discrete CVTs. Even for moderate values of $\widetilde{m}$ and $\widetilde{d}$, the number of possible clusterings of the data is enormous, meaning that an exhaustive search for minimizers of the cluster energy cannot be carried out. Fortunately, there do exist good algorithms for determining CVTs of discrete point sets. We use the h-means and k-means methods (see, e.g., [16, 39, 40]) which are described in section 2.2.1. The h-means method is cheaper to use than k-means, but does not minimize the cluster energy function directly, and so it is not completely reliable. Since the two methods are compatible, an efficient hybrid technique consists of applying h-means until it can achieve no further energy reduction, followed by (presumably few) steps of k-means to drive the energy down.

The h-means and k-means algorithms have the attractive property of being [illegible] with respect to the cluster energy. Given any cluster configuration $(\mathcal{V}, C, \mathcal{Z})$, both algorithms consider a series of modifications to the cluster assignment function C. However, a modification is only made if it is guaranteed to reduce the cluster energy $E(\mathcal{V}, C, \mathcal{Z})$. Thus, an algorithm based on taking h-means or k-means steps cannot diverge or produce a final clustering having higher energy than the initial clustering. Also, since the set $\mathcal{V}$ is finite dimensional, the algorithms are guaranteed to terminate in finitely many steps. By virtue of the nearest neighbor and centroidal generator features, the resulting clusters will be discretely convex and will tend to correspond to natural clusterings of the data.

The single step approach of the h-means and k-means methods, i.e., the fact that the methods consider one member of the data set $\mathcal{V}$ at a time, has the disadvantage of making the methods vulnerable to "local minima."[12] The k-means algorithm will never allow a one-point modification that increases the cluster energy, and so it is guaranteed to halt if it reaches a local minimum. However, for a given problem, there can be many local minima that are not global minimizers of the cluster energy. To deal with the problem of local minima, one can simply apply the CVT construction algorithm several times, using different initial configurations, and then take the final configuration having the lowest cluster energy to be the best estimate of the optimal configuration. One should also note that in other applications of CVT it has been found the local minima determined by, say, the k-means method yield very good

[12]Given a nearest-neighbor, i.e., a Voronoi, cluster configuration $(\mathcal{V}, C, \mathcal{Z})$, define a *one point change* as a change, for a single data point $\vec{V}_m$, in the value of the cluster assignment function from $C(\vec{V}_m) = \vec{Z}_i$ to $C(\vec{V}_m) = \vec{Z}_j$, followed by the replacement of $\vec{Z}_i$ and $\vec{Z}_j$ by the centroids of their new associated clusters. A *local minimum* is then a cluster configuration $(\mathcal{V}, C, \mathcal{Z})$ with the property that every one-point change results in an increase in the cluster energy.

results; this is borne out by the computational results of this paper which show that CVT reduced-order modeling can provide effective low-dimensional approximations.

2.2.1. The h-means and k-means algorithms. The h-means algorithm begins with an initial cluster configuration $(\mathcal{V}, C, \mathcal{Z})$. Then, for each point $\vec{V}_m \in \mathcal{V}$, the algorithm finds the nearest cluster generator $\vec{Z}_i \in \mathcal{Z}$; if that is not the cluster to which the data point is currently assigned, the data point is "transferred" to the cluster associated with its nearest generator. When all data points have been considered, the cluster generators $\vec{Z}$ are replaced by the centroids of the newly formed clusters. If at least one point was transferred during this process, a new sweep of all the points is carried out, and this continues until a sweep is made in which no point is transferred.

Recall that:

$\mathcal{V} = \{\vec{V}_m\}_{m=1}^{\widetilde{m}}$ denotes the fixed set of points to be clustered into a CVT consisting of $\widetilde{d}$ clusters, where $\widetilde{d}$ is also fixed;

$\mathcal{Z} = \{\vec{Z}_i\}_{i=1}^{\widetilde{d}}$ denotes the cluster generators which can possibly change at each step of the algorithm;

$\{\mathcal{V}_i\}_{i=1}^{\widetilde{d}}$ denotes the clustering of $\mathcal{V}$ which can also possibly change at each step of the algorithm; and

$\overline{d}_i$ denotes the cardinality of the cluster $\mathcal{V}_i$ which can also possibly change at each step of the algorithm.

Then, the h-means algorithm may be formalized as follows:

```
do
    for each V_m ∈ V
        determine Z_i ∈ Z that minimizes |V_m − Z_j|
        if V_m ∉ V_i, then
            reassign V_m to V_i
        end if
    end for
    if no transfers occurred, exit the loop
    for each i = 1, ..., d̃
        Z_i = (1/d̄_i) Σ_{V_m ∈ Z_i} V_m
    end for
end do
```

The k-means algorithm is a sophisticated version of h-means. The h-means transfer test is based on the distance from a point to a cluster center. However, if a point is transferred to a new cluster, the cluster center, when it is recomputed, will move closer to the point, something that h-means misses until the step is complete, i.e., until all points in $\mathcal{V}$ are treated. Within each iteration, the k-means algorithm works with a transfer test that directly measures the change in energy that occurs when a point moves from one cluster to another and then immediately redetermines the cluster centroids before moving to the next point in $\mathcal{V}$. A single k-means sweep is computationally more expensive than a single h-means sweep. This added expense is the reason why the h-means algorithm is employed first to perform the "easy" transfers.[13] The k-means algorithm may be formalized as follows:

[13] How much more expensive is k-mean compared to h-mean depends on exactly how one carries out the steps in the two methods, e.g., how one calculates nearest neighbors. Our limited experience has indicated that it is more efficient to use k-means, especially near the end of the CVT construction process.

```
do
    for each V_m ∈ 𝒱
        evaluate the cluster energy for all possible transfers of V_m from its
          current cluster 𝒱_i to any of the other clusters 𝒱_j, j = 1, ..., d̃,
          j ≠ i
        if moving V_m from its current cluster 𝒱_i to the cluster 𝒱_k will most
          reduce the cluster energy, then
            reassign V_m to cluster 𝒱_k
            replace the cluster generators Z_i and Z_k by the centroids of
            the newly modified clusters 𝒱_i and 𝒱_k, respectively
        end if
    end for
    if no transfers occurred, exit the loop
end do
```

The main lines of a hybrid algorithm can now be made clear. Starting with a randomly initialized cluster configuration, one applies the h-means iteration (presumably many times) and the k-means iteration (hopefully a few times) to produce a locally optimal cluster configuration. One repeats the process several times, each time starting with a different random initial cluster configuration. The generators of the locally optimal configuration having the lowest cluster energy are then output as the approximation to the desired CVT reduced-order basis.

We note that there are several ancillary processing issues that arise in the implementation of CVT construction algorithms. For example, one may wish to thin, shift, or scale the snapshots so that clustering patterns are more readily exposed. One may also want to use, instead of the Euclidean one, metrics that do not differentiate between snapshot vectors which point in nearly the same direction but which have different magnitudes. Additionally, the choice of the initial set of generators can affect the efficiency of a CVT construction algorithm.

3. Computational experiments. To demonstrate the use, efficiency, and accuracy of the CVT-based reduced-order modeling technique, two examples are considered, which we denote the [illegible] and [illegible] problems. Both are incompressible flow problems having boundary conditions that include an inflow shape function containing a multiplicative parameter α that controls the strength of the inflow.

3.1. In/Out problem. The governing equations of two-dimensional incompressible viscous flow in a square region $\Omega = (0,1) \times (0,1)$ are the Navier–Stokes system

$$\frac{\partial \mathbf{u}}{\partial t} - \nu \, \Delta \mathbf{u} + \mathbf{u} \cdot \nabla \mathbf{u} + \nabla p = \mathbf{0} \quad \text{in} \quad (0,T) \times \Omega, \tag{3.1}$$

$$\nabla \cdot \mathbf{u} = 0 \quad \text{in} \quad (0,T) \times \Omega \tag{3.2}$$

along with, for the specific problem we considered here, the initial and boundary conditions

$$\mathbf{u}(0,\mathbf{x}) = \mathbf{u}_0(\mathbf{x}) \quad \text{in} \quad \Omega, \tag{3.3}$$

$$\mathbf{u} = 100 \, \alpha(t) \, y \, (0.2 - y) \quad \text{on} \quad (0,T) \times \Gamma_i, \tag{3.4}$$

$$(\mathbf{n} \cdot \nabla) \, \mathbf{u} = 0 \quad \text{on} \quad (0,T) \times \Gamma_o, \tag{3.5}$$

$$\mathbf{u} = 0 \quad \text{on} \quad (0,T) \times \Gamma_d, \tag{3.6}$$

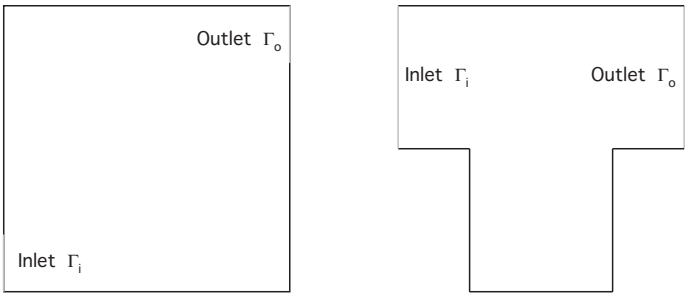


FIG. 3.1. *The flow regions for the In/Out (left) and T-cell (right) examples.*

where $\mathbf{x} = (x, y)$, $\Gamma_i = \{x = 0\,;\, 0 < y < 0.2\}$, and $\Gamma_o = \{x = 1\,;\, 0.8 < y < 1\}$ are the inflow and outflow parts of the boundary, respectively, and $\Gamma_d = \partial\Omega\backslash(\overline{\Gamma}_i \cup \overline{\Gamma}_o)$. See Figure 3.1. In (3.4), $\alpha(t)$ is a parameter that determines the strength of the parabolic inflow velocity profile. We choose $\nu = 1$ and $T = 0.08$.

Accurate Galerkin-mixed method finite element approximations of the solutions of (3.1)–(3.6) are obtained using the $P2$–$P1$ Taylor–Hood finite element pair on a grid of 1681 nodes. Finite element solutions are used for the generation of snapshots and later for comparison with CVT-based reduced-order solutions.

For the generation of snapshots, we will also solve, by the finite element method, stationary versions of (3.1)–(3.6) for which the time derivative term in (3.1) and the initial condition (3.3) are omitted and α in (3.4) is chosen independent of t.

3.1.1. Generating snapshots. We use the procedure given in section 1.1.1 to determine a set of snapshot vectors. First, the finite element approximation to the stationary version of (3.1)–(3.6) with $\alpha = 1$ is obtained. Using that steady-state solution as the initial data $\mathbf{u}_0$ in (3.3) and using $\alpha = 5$ for $0 < t < T/2 = 0.04$ and $\alpha = 1$ for $0.04 < t < T = 0.08$ in the boundary condition (3.4), we then determine a finite element approximation $\sum_{k=1}^{\widetilde{j}} U_k(t)\phi_k(\mathbf{x})$ of the solution of (3.1)–(3.6), where $\widetilde{j}$ denotes the dimension of the finite element space used for the velocity and ϕ_x denotes the basis functions for that space. This is the flow we use to generate the snapshots; it can be viewed as one for which the steady-state solution for $\alpha = 1$ is suddenly jolted, at $t = 0$, by increasing the value of α to five, and jolted again at $t = T/2 = 0.04$ by decreasing the value of α back to one. The 800 snapshot vectors

$$\vec{U}_m = \begin{pmatrix} U_1(t_m) \\ U_2(t_m) \\ \vdots \\ U_{\widetilde{j}}(t_m) \end{pmatrix}, \quad m = 1, \ldots, 800$$

are then determined by evaluating the solution of this impulsively started problem at 800 equally spaced time values t_m, $m = 1, \ldots, 800$, ranging from $t = 0$ to $t = T = 0.08$.

For subsequent use, it is convenient to modify the 800 snapshots so that they satisfy homogeneous boundary conditions. To this end, we first obtain the reference finite element approximation $\mathbf{v}(\mathbf{x}) = \sum_{k=1}^{\widetilde{j}} V_k\phi_k(\mathbf{x})$ of the stationary version of (3.1)–(3.6) with $\alpha = 3$. We then modify the first 400 snapshots by

$$\vec{U}_m \leftarrow \left(\vec{U}_m - \frac{5}{3}\vec{V}\right) \quad \text{for } m = 1, \ldots, 400$$

and, in the same way, the second 250 snapshots are modified

$$\vec{U}_m \leftarrow \left(\vec{U}_m - \frac{1}{3}\vec{V} \right) \quad \text{for } m = 401, \ldots, 800,$$

where

$$\vec{V} = \begin{pmatrix} V_1 \\ V_2 \\ \vdots \\ V_{\widetilde{j}} \end{pmatrix}.$$

In this way, all the snapshots satisfy homogeneous boundary conditions.

The snapshots we have generated by the above processes we will refer to as the [illegible] This is to contrast them with a second set of snapshots we will use, referred to as the [illegible] which are obtained from the nonnormalized snapshots by dividing by their respective length, i.e., the normalized snapshots are given by

$$\vec{U}_m \leftarrow \frac{\vec{U}_m}{|\vec{U}_m|}.$$

We will also used a [illegible] that is determined from the nonnormalized snapshots by deleting every other snapshot from the latter, i.e., the 250 reduced snapshots are given by

$$\vec{U}_m \quad \text{for } m = 2, 4, 6, \ldots, 800.$$

3.1.2. Generating CVT reduced basis. We next apply the algorithms of section 2.2 to determine the generators of a CVT of each of the snapshot sets; a set of generators is to be used as a reduced basis. CVTs having $\widetilde{d} = 4$, 8, and 16 generators are determined. Figure 3.2 displays, for the 4-generator case, the CVT basis computed from the normalized snapshots. Note that since the original boundary conditions have been "subtracted away," each basis function satisfies a zero Dirichlet boundary condition at the inlet, and a zero Neumann condition at the outlet. In the interior of the region, each basis function satisfies (discretized) continuity equation.

Figure 3.3 displays the system of 8 CVT basis functions computed from the same data. It is important to note that the set of size 8 is not built by augmenting the set of size 4; most of the elements of the larger set seem significantly different from any of those of the smaller set.

Each generator $\vec{Z}_i \in \mathbb{R}^{\widetilde{j}}$ of a CVT defines a finite element function, i.e., if

$$\vec{Z}_i = \begin{pmatrix} Z_1^i \\ Z_2^i \\ \vdots \\ Z_{\widetilde{j}}^i \end{pmatrix} \quad \text{for } i = 1, \ldots, \widetilde{d},$$

we then have the corresponding finite element functions

$$\mathbf{z}_i = \sum_{k=1}^{\widetilde{j}} Z_1^i \phi_j(\mathbf{x}) \quad \text{for } i = 1, \ldots, \widetilde{d}.$$

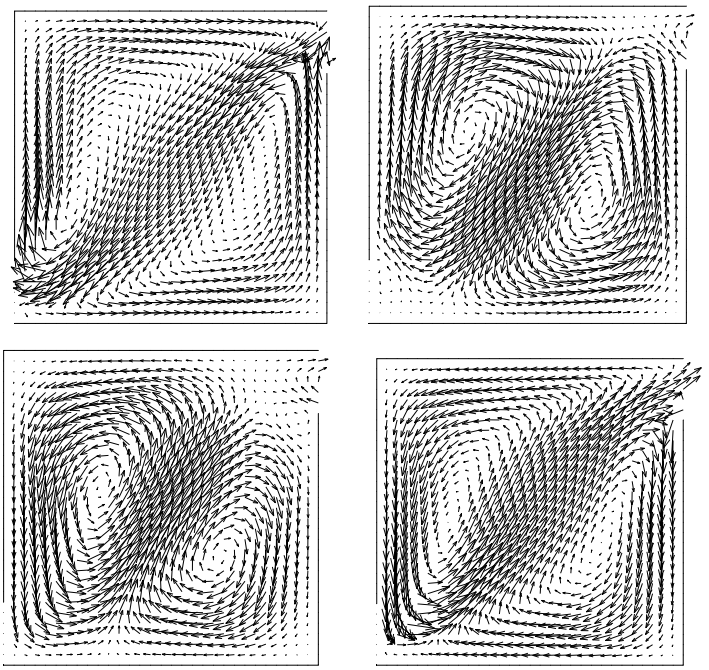


FIG. 3.2. *The CVT basis of dimension* 4 *determined from normalized snapshots.*

3.1.3. Determining CVT-based reduced-order approximations. We now use the Galerkin procedure discussed in section 1.1.3 to determine a CVT-based reduced-order approximation of the velocity field. The approximate velocity field $\mathbf{u}(\mathbf{x}, t)$ is represented as a linear combination of the CVT basis functions as follows:

$$\mathbf{u}_{cvt} = \beta(t)\mathbf{v} + \sum_{i=1}^{\widetilde{d}} a_i(t)\mathbf{z}_i(\mathbf{x}), \tag{3.7}$$

where $\mathbf{z}_i$ denotes the ith CVT basis function, $a_i(t)$ is the corresponding coefficient, $\mathbf{v}$ is the reference solution defined in section 3.1.1, $\beta(t) \equiv \alpha(t)/\alpha_r$, $\alpha_r = 3$, and $\widetilde{d}$ is the total number of CVT basis functions. In our computations, $\widetilde{d} = 4$, 8, or 16. The first term in (3.7) is included so that $\mathbf{u}$ satisfies the boundary conditions in (3.1)–(3.6).

Applying the Galerkin principle which forces the residual to be orthogonal to each of the basis functions, we obtain

$$\begin{aligned} \int_\Omega \frac{\partial}{\partial t}\mathbf{u}_{cvt} \cdot \mathbf{z}_j \, d\Omega + \nu \int_\Omega \nabla \mathbf{u}_{cvt} : \nabla \mathbf{z}_j \, d\Omega \\ + \int_\Omega (\mathbf{u}_{cvt} \cdot \nabla)\mathbf{u}_{cvt} \cdot \mathbf{z}_j \, d\Omega = 0 \qquad \text{for } j = 1, \ldots, \widetilde{d}. \end{aligned} \tag{3.8}$$

Note that due to the fact that the CVT basis functions $\mathbf{z}_j$, $j = 1, \ldots, \widetilde{d}$, are discretely solenoidal, the pressure does not appear in this system. Using (3.7), it is easy to see

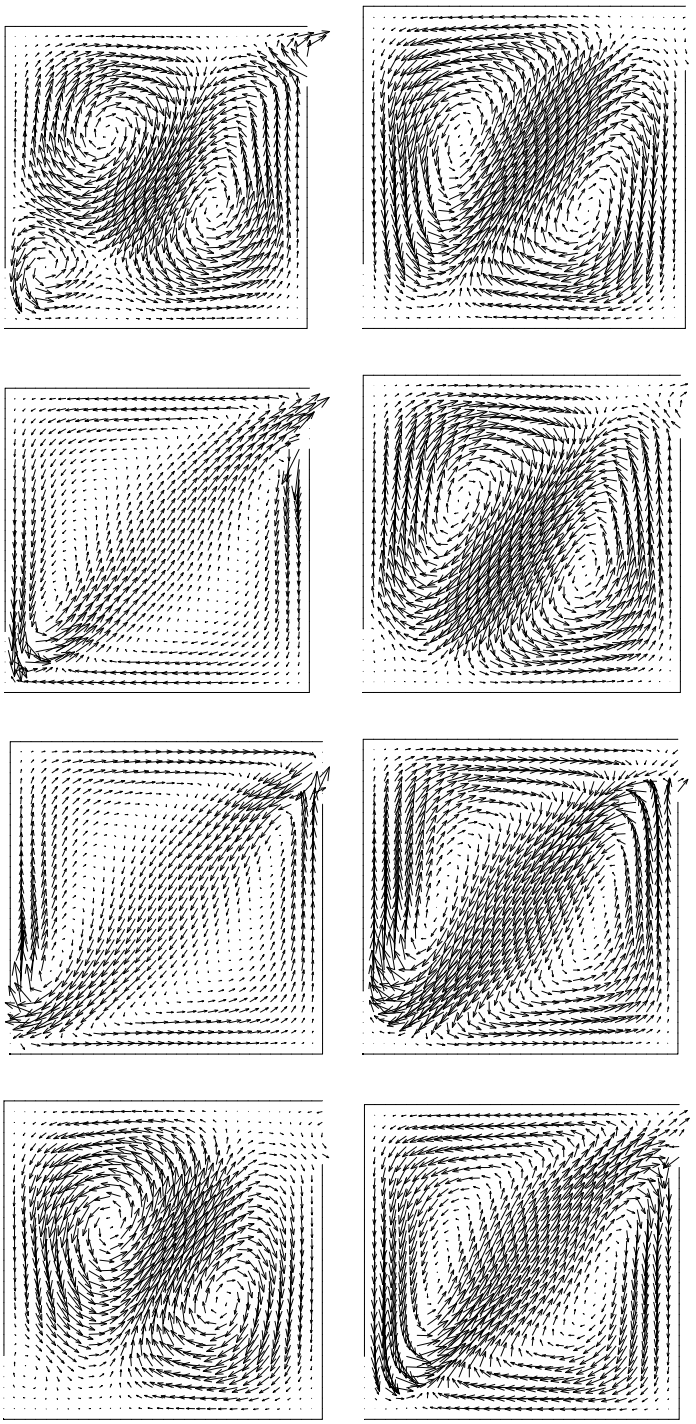


FIG. 3.3. *The CVT basis of dimension 8 function determined from normalized snapshots.*

that (3.8) is equivalent to the system of nonlinear ordinary differential equations

$$
\begin{aligned}
&\sum_{i=1}^{\widetilde{d}} \frac{d}{dt} a_i(t)\Big(\mathbf{z}_i, \mathbf{z}_j\Big) + \nu \sum_{i=1}^{\widetilde{d}} a_i(t)\Big(\nabla \mathbf{z}_i, \nabla \mathbf{z}_j\Big) \\
&+ \left(\sum_{i=1}^{\widetilde{d}} a_i(t)\mathbf{z}_i \cdot \nabla \sum_{k=1}^{\widetilde{d}} a_k(t)\mathbf{z}_k, \mathbf{z}_j\right) + \beta(t) \sum_{i=1}^{\widetilde{d}} a_i(t)\Big(\mathbf{z}_i \cdot \nabla \mathbf{v} + \mathbf{v} \cdot \nabla \mathbf{z}_i, \mathbf{z}_j\Big) \\
&+ \frac{d}{dt}\beta(t)\Big(\mathbf{v}, \mathbf{z}_j\Big) + \beta(t)\Big(\beta(t) - 1\Big)\Big(\mathbf{v} \cdot \nabla \mathbf{v}, \mathbf{z}_j\Big) = 0, \quad j = 1, \ldots, \widetilde{d},
\end{aligned}
\tag{3.9}
$$

along with the initial conditions

$$
\sum_{i=1}^{\widetilde{d}} a_i(0)\big(\mathbf{z}_i, \mathbf{z}_j\big) = \big(\mathbf{u}_0 - \beta(0)\mathbf{v}, \mathbf{z}_j\big), \tag{3.10}
$$

where $(\cdot, \cdot)$ denotes the $L^2(\Omega)$ inner product. The set of ordinary differential equations (3.9)–(3.10) is solved by using a 4th order Runge–Kutta method.

To demonstrate robustness, the low-dimensional, CVT-based dynamic model was solved using several shapes for the inlet velocity factor $\alpha(t)$ and several different choices for the CVT basis set. The two choices for the inlet velocity function for which we present results are given by:

- [illegible] 1: a hat function with respect to time (see Figure 3.4);
- [illegible] 2: a sinusoid with respect to time (see Figure 3.5).

Nine different CVT bases were used: for $\widetilde{d} = 4$, 8, and 16,

- a set of $\widetilde{d}$ CVT basis functions derived from the nonnormalized snapshot set;
- a set of $\widetilde{d}$ CVT basis functions derived from the normalized snapshot set;
- a set of $\widetilde{d}$ CVT basis functions derived from the reduced snapshot set.

In addition, for both Cases 1 and 2, solutions of the system (3.1)–(3.6) were also approximated using the full finite element discretization employing thousands of unknowns.

Computing the CVT-based reduced-order solution for basis function sets of size $\widetilde{d} = 4$, 8, and 16 allowed for some study of the approximation error behavior. The CVT-based solutions are compared to the corresponding solutions of the full finite element system. The relative L^2 error was computed at each time step, based on the difference between the full finite element and CVT-based reduced-order solutions. The plots of these L^2 errors versus time are displayed in Figures 3.4 and 3.5 for Cases 1 and 2, respectively. Note that, for each case and for every time step, the overall L^2 error decreases as the size of the CVT basis set is increased.

Further comparisons are provided in Figures 3.6 and 3.7 where we display, side by side, the full finite element and CVT-based reduced-order solutions using 8 basis functions at selected time steps. Visually, the two solutions seem indistinguishable; however, the CVT-based solution was constructed from a system of 8 ordinary differential equations, while the corresponding finite element solution is derived by solving 1681 ordinary differential equations. The total running time was reduced by a factor of several thousand.

3.2. The T-cell problem. The CVT-based reduced-order modeling methodology was also applied to a second example of flow in a t-shaped region; see Figure 3.1. The flow enters from the left and exits on the right. We choose $\nu = 1$ and $T = 0.05$. A

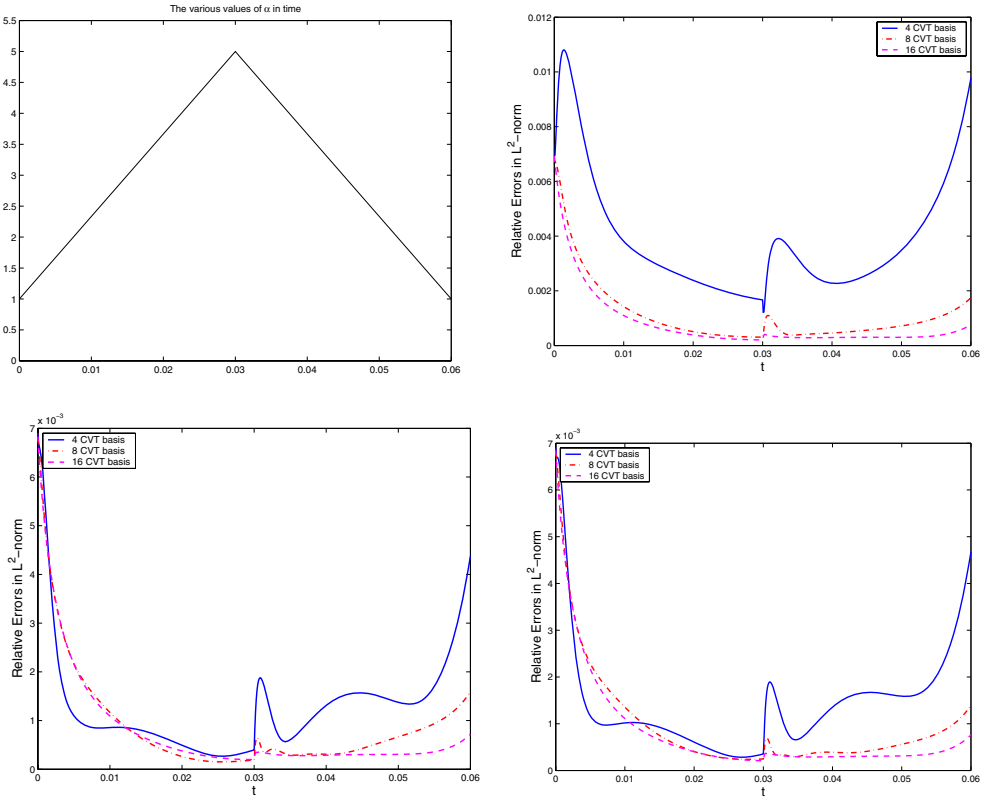


FIG. 3.4. *Multiplicative inflow strength parameter α for Case* 1 *(top left) and L^2-norm of difference between CVT-based reduced-order solutions and a full finite element simulation vs. time for the In/Out problem based on the normalized snapshot set (top right), the nonnormalized snapshot set (bottom left), and the reduced snapshot set (bottom right).*

grid of 4961 nodes was used for all finite element computations. Again, a multiplicative parameter α controls the strength of the inflow. For $0 < t < 0.025$, $\alpha = 5$ was used which represents an abrupt change at $t = 0$ from the value 1 used to generate the steady-state initial condition. For $0.025 < t < 0.05$, $\alpha = 1$ was used which represents another abrupt change at $t = 0.025$. The abrupt changes in the value of α cause sharp transient effects. This single finite element solution of (3.1)–(3.6) was sampled at 500 equally spaced time intervals to generate 500 snapshot vectors.

As for the In/Out problem, computations were performed for CVT bases of size 4 (displayed in Figure 3.8), 8 (displayed in Figure 3.9), and 16. Also, nonnormalized, normalized, and reduced snapshot sets were considered. The nine basis sets were then used to approximate the solution of a time-dependent flow in the T-shaped region. The multiplicative flow strength factor $\alpha(t)$ was chosen to be

$$\alpha(t) = \begin{cases} 160t + 1 & (0 \le t \le 0.025) \\ -160(t - 0.05) + 1 & (0.025 \le t \le 0.05). \end{cases}$$

A finite element solution of the same problem (using 4961 nodes) was also computed for comparison purposes. In Figure 3.10, the time histories of the differences between the CVT-based reduced-order solutions and the full finite element solutions, measured in the $L^2(\Omega)$-norm, are provided for the nine basis sets.

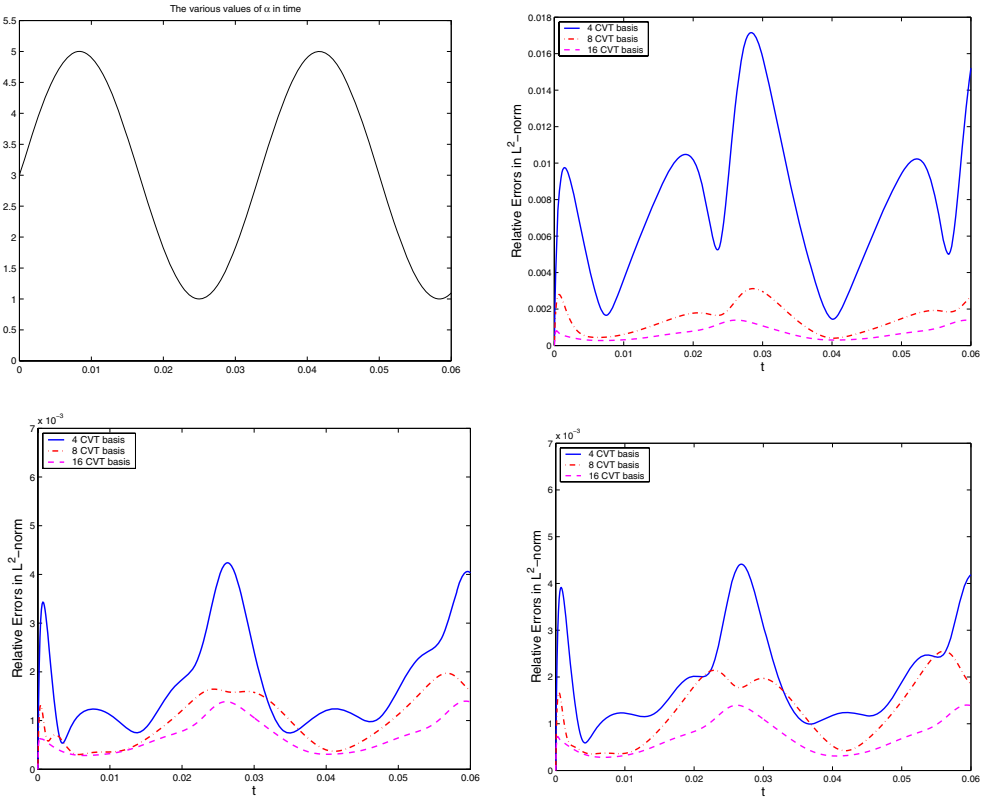


FIG. 3.5. *Multiplicative inflow strength parameter α for Case 2 (top left) and L^2-norm of difference between CVT-based reduced-order solutions and a full finite element simulation vs. time for the In/Out problem based on the normalized snapshot set (top right), the nonnormalized snapshot set (bottom left), and the reduced snapshot set (bottom right).*

3.3. Some information about CVT bases. It is of interest to mention some details about the CVT basis generation process; this discussion will consider the case of 8 CVT generators, based on nonnormalized snapshots, for the T-cell problem.

The basis generation algorithm preprocessed the data by subtracting an appropriate multiple of a steady-state solution that satisfies the boundary conditions and which had been computed with the parameter value $\alpha = 1$. The overall computation was repeated with 15 random initial generator configurations. The inner h-means and k-means algorithms were limited to a maximum of 30 iterations but always concluded before that limit was reached. The initial cluster energy of the random configurations was generally about 50,000; the final cluster energy of the 15 cases varied from 2,879.5 to 3,212.5.

For a "typical" random initial configuration, Table 3.1 shows how the cluster energy drops. Of course, the huge energy at the initial random configuration is slightly misleading. It drops tremendously after the first h-means step, as each data point is assigned to its nearest cluster center.

There was an interesting feature observed about the final cluster generators. For each cluster, the cluster elements tended to comprise all the solutions in a single time interval. In particular, for the case discussed here, the 8 cluster generators had components as listed in Table 3.2. In the interest of clarity, we've renumbered the cluster indices.

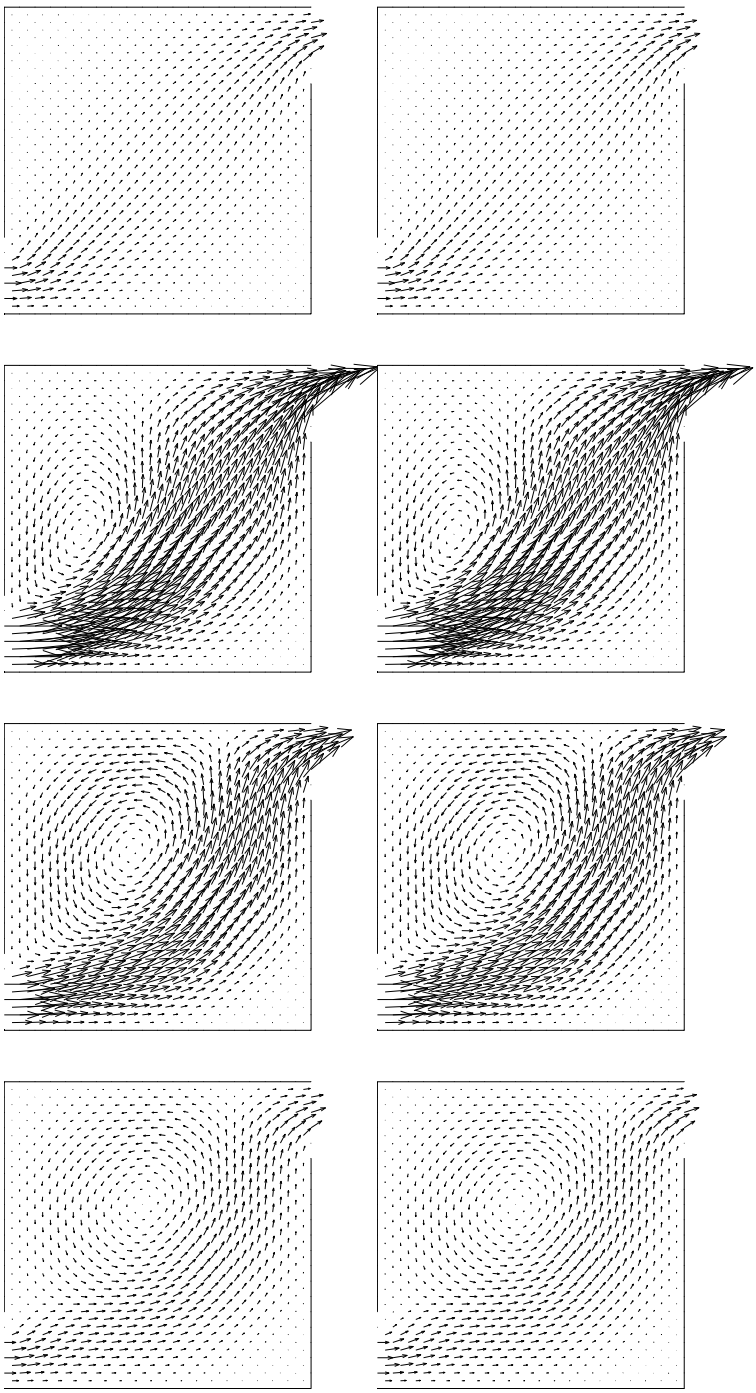


FIG. 3.6. *Full finite element approximation (left) and CVT-based reduced-order approximation of size 8 based on nonnormalized snapshots (right) for* $t = 0.1$ *(top),* $t = 1.0$ *(second),* $t = 1.5$ *(third),* $t = 2.0$ *(bottom).*

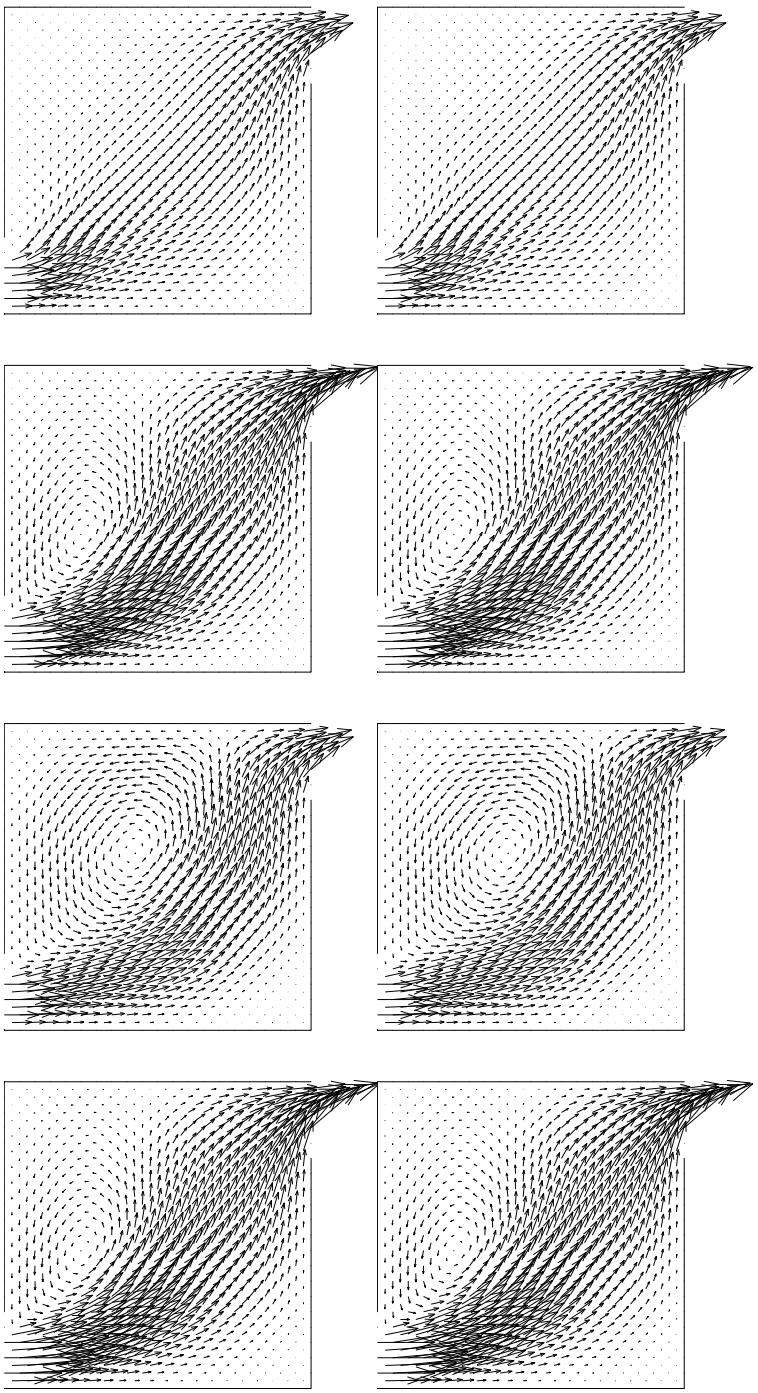


FIG. 3.7. *Full finite element approximation (left) and CVT-based reduced-order approximation of size 8 based on nonnormalized snapshots (right) for* $t = 2.5$ *(top),* $t = 3.0$ *(second),* $t = 3.5$ *(third),* $t = 5.0$ *(bottom).*

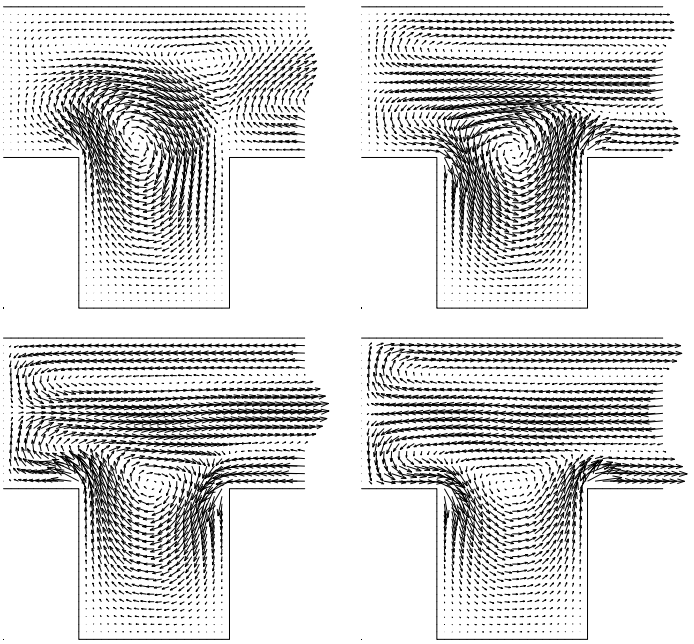


FIG. 3.8. *The CVT basis of dimension 4 determined from nonnormalized snapshots.*

Note that, except for cluster 5, the clusters are formed exactly from a sequence of data points at neighboring times. The two smallest clusters, number 1 and number 6, correspond to the initial transient and the "shock" to the system that occurs after step 250, when the system parameter α is changed. Since the solution is changing rapidly at those times, it makes sense that the clusters are small, and yet still comprise a sizable portion of the total cluster energy. Also, the one exceptional cluster, number 5, corresponds to the relatively quiescent "tails" of the two evolution processes, since it contains the solutions for steps [75, 250] and [308, 500]. This near "sequential" arrangement of snapshots within the clusters may be useful in some settings, e.g., for adapting a CVT basis.

4. Concluding remarks. We have introduced and discussed a centroidal Voronoi tessellation based reduced-order modeling methodology and have shown that it is effective, i.e., accurate and inexpensive, on some example problems. The CVT-based approach is based on data clustering concepts. Compared to POD-based reduced-order modeling approaches, which are the ones in most common use today, the CVT-based approach is potentially less expensive, i.e., for the same number of snapshots and the same number of basis vectors, the CVT-based approach is usually less costly than POD-based approaches which require the solution of a dense eigenvalue of singular value decomposition problem. Although no direct comparisons of POD- and CVT-based results were made, the computational results provided in this paper indicate that the latter approach provides accurate results for the same type of problems for which the former approach works.

There are several avenues for the further development of CVT-based reduced-order modeling methodologies which we are currently considering. First, in [9], a discussion was given of how the CVT- and POD-based reduced-order modeling approaches can be combined to produce a hybrid method (called CVOD in that paper)

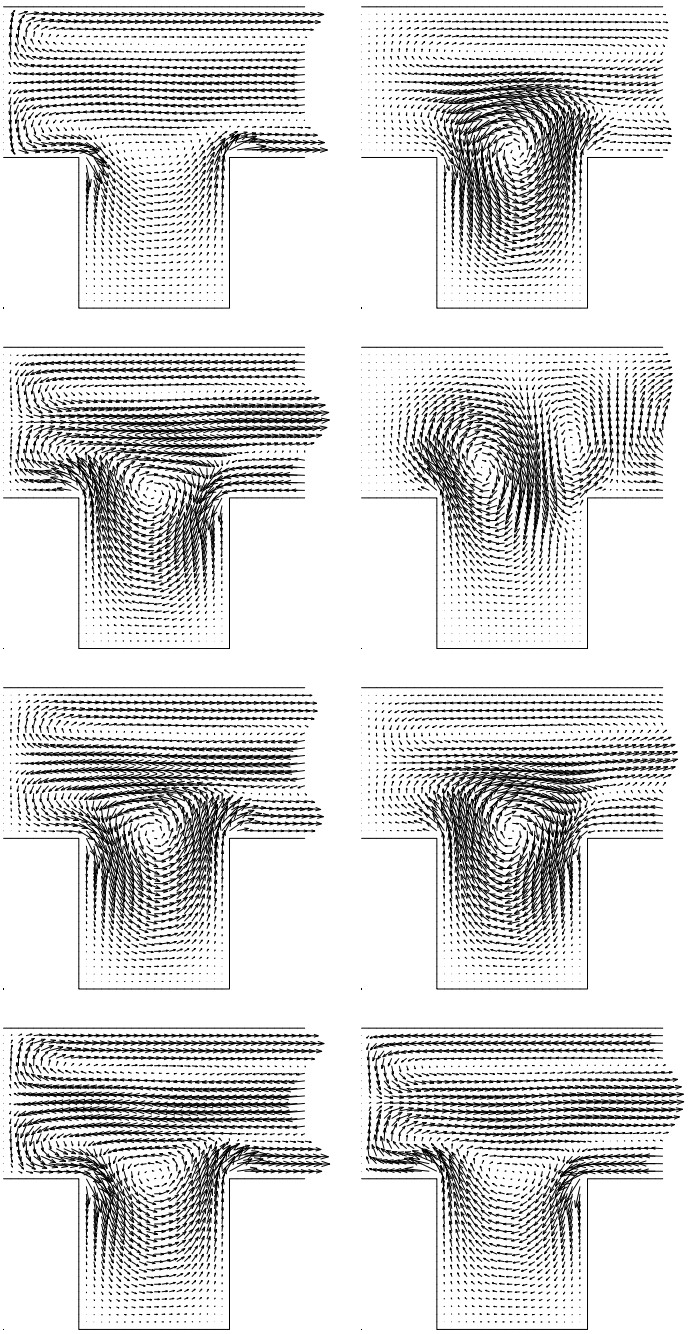


FIG. 3.9. *The CVT basis of dimension 8 determined from nonnormalized snapshots.*

that possesses the good features of both. The implementation of CVOD and experimentation with the various hybridization possibilities is a subject that is certainly deserving of attention.

Another important avenue we are pursuing is the application of CVT- and CVOD-based reduced-order modeling methodologies to control and optimization problems for

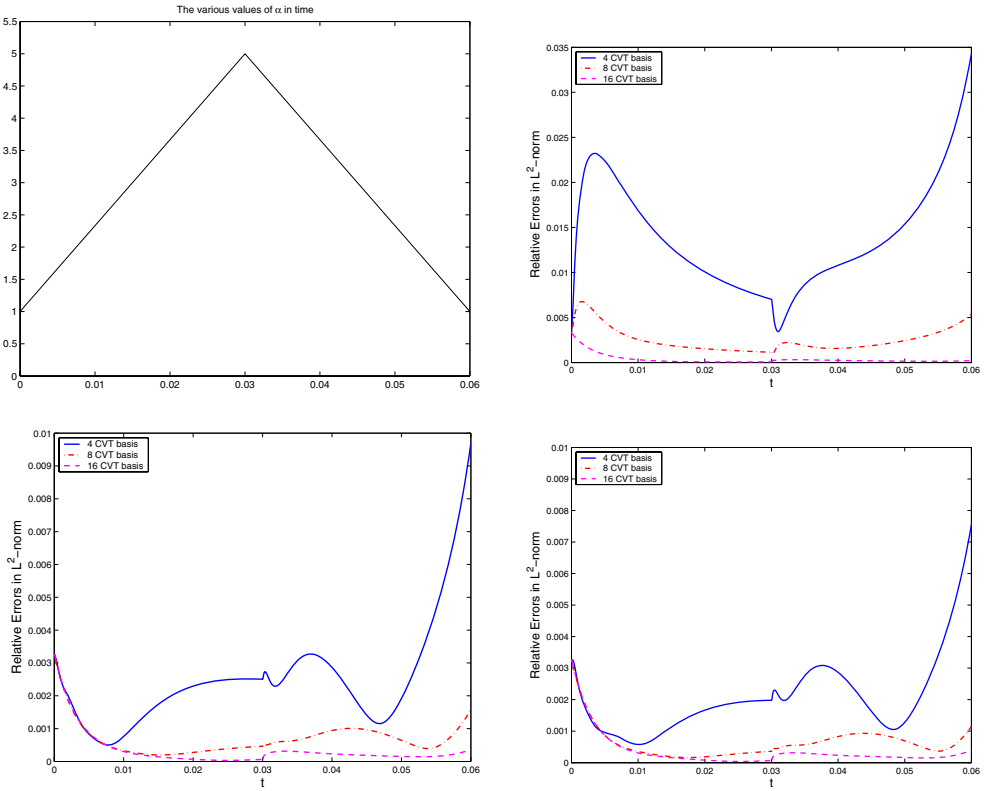


FIG. 3.10. *Multiplicative inflow strength parameter α (top left) and L^2-norm of difference between CVT-based reduced-order solutions and a full finite element simulation vs. time for the T-cell problem based on the normalized snapshot set (top right), the nonnormalized snapshot set (bottom left), and the reduced snapshot set (bottom right).*

TABLE 3.1
Energy and iteration statistics for the CVT-basis generation.

Process	Final Energy	Iterations
Random initial	49,860.5	1
h-means	3,283.4	22
k-means	3,212.5	4

TABLE 3.2
Cluster statistics for computed CVT generators.

Cluster	Cluster Energy	Population	Range
1	281.1	5	[1, 5]
2	247.5	10	[6, 15]
3	236.4	18	[16, 33]
4	215.3	41	[34, 74]
5	532.1	369	[75, 250] + [308, 500]
6	523.3	7	[251, 257]
7	428.3	15	[258, 272]
8	420.1	35	[273, 307]
Total	3,212.5	500	[1, 500]

complex systems. Many approaches to such problems require multiple or real-time state simulations so that reducing the cost of those simulations is a necessity. Indeed, much of the current interest in the reduced-order modeling of complex systems is motivated by control and optimization problems. One issue that is sure to enter these considerations is the adaptation of reduced-order bases. Adaptation is likely to be needed since, during a control or optimization process, the snapshot set that was used to generate the reduced basis my not remain adequate to represent the dynamic behavior of the system. The "sequential" aspect of CVT clustering of the snapshots may turn out to be a useful feature in this context.

A third avenue is to use the flexibility provided by CVT methodologies to advantage in the reduced-order modeling setting. For example, it is often the case that one wants a certain snapshot or a group of snapshots to have greater influence on the reduced basis than do the others. This can be easily accomplished by simply scaling the important snapshots, or equivalently, by introducing a density function defined over the snapshot set; important snapshots would have high associated densities. CVT construction algorithms would then automatically prejudice some of the generators towards the important snapshots. Furthermore, constrained CVTs can be applied which enable one to include important snapshot vectors in the reduced basis.

REFERENCES

[1] N. Aubry, W. Lian, and E. Titi, *Preserving symmetries in the proper orthogonal decomposition*, SIAM J. Sci. Comput., 14 (1993) pp. 483–505.

[2] G. Berkooz, P. Holmes, and J. Lumley, *The proper orthogonal decomposition in the analysis of turbulent flows*, Ann. Rev. Fluid. Mech., 25 (1993), pp. 539–575.

[3] G. Berkooz and E. Titi, *Galerkin projections and the proper orthogonal decomposition for equivariant equations*, Phys. Lett. A, 174 (1993), pp. 94–102.

[4] E. Christensen, M. Brons, and J. Sorensen, *Evaluation of proper orthogonal decomposition-based decomposition techniques applied to parameter-dependent nonturbulent flows*, SIAM J. Sci. Comput., 21 (2000), pp. 1419–1434.

[5] A. Deane, I. Kevrekidis, G. Karniadakis, and S. Orszag, *Low-dimensional models for complex geometry flows: Applications to grooved channels and circular cylinders*, Phys. Fluids A, 3 (1991), pp. 2337–2354.

[6] Q. Du, V. Faber, and M. Gunzburger, *Centroidal Voronoi tessellations: Applications and algorithms*, SIAM Rev., 41 (1999), pp. 637–676.

[7] Q. Du and M. Gunzburger, *Model reduction by proper orthogonal decomposition coupled with centroidal Voronoi tessellation*, in Proceedings of the Fluids Engineering Division Summer Meeting, FEDSM2002-31051, ASME, 2002.

[8] Q. Du and M. Gunzburger, *Grid generation and optimization based on centroidal Voronoi tessellations*, Appl. Math. Comput., 133 (2002), pp 591–607.

[9] Q. Du and M. Gunzburger, *Centroidal Voronoi tessellation based proper orthogonal decomposition analysis*, in Proceedings of the 8th Conference on Control of Distributed Parameter Systems, Birkhauser, Basel, Switzerland, 2002, pp. 137–150.

[10] Q. Du, M. Gunzburger, and L. Ju, *Meshfree, probabilistic determination of points, sets and support regions for meshless computing*, Comput. Methods Appl. Mech. Engrg., 191 (2002), pp. 1349–1366.

[11] Q. Du, M. Gunzburger, and L. Ju, *Constrained centroidal Voronoi tessellations for surfaces*, SIAM J. Sci. Comput., 24 (2003), pp. 1488–1506.

[12] Q. Du, M. Gunzburger, and L. Ju, *Voronoi-based finite volume methods, optimal Voronoi meshes, and PDEs on the sphere*, Comput. Methods Appl. Mech. Engrg., 192 (2003), pp. 3933–3957.

[13] A. Faulds, *Centroidal Voronoi Decompositions, Algorithms and Applications*, M.S. thesis, Department of Mathematics, Penn State University, State College, PA 2002.

[14] A. Faulds and B. King, *Sensor location in feedback control of partial differential equation systems*, in Proceedings of the 2000 IEEE CCA/CACSD, IEEE, Washington, 2000, pp. 536–541.

[15] M. Graham and I. Kevrekidis, *Pattern analysis and model reduction: Some alternative approaches to the Karhunen-Lóeve decomposition*, Comput. Chem. Engrg., 20 (1996), pp. 495–506.
[16] J. Hartigan and M. Wong, *Algorithm AS 136: A K-means clustering algorithm*, Appl. Statist., 28 (1979), pp. 100–108.
[17] P. Holmes, J. Lumley, and G. Berkooz, *Turbulence, Coherent Structures, Dynamical Systems and Symmetry*, Cambridge University Press, Cambridge, 1996.
[18] P. Holmes, J. Lumley, G. Berkooz, J. Mattingly, and R. Wittenberg, *Low-dimensional models of coherent structures in turbulence*, Phys. Rep., 287 (1997), pp. 337–384.
[19] L. Ju, Q. Du, and M. Gunzburger, *Probablistic methods for centroidal Voronoi tessellations and their parallel implementations*, J. Parallel Comput., 28 (2002), pp. 1477–1500.
[20] K. Kunisch and S. Volkwein, *Control of Burger's equation by a reduced order approach using proper orthogonal decomposition*, JOTA, 102 (1999), pp. 345–371.
[21] K. Kunisch and S. Volkwein, *Galerkin proper orthogonal decomposition methods for parabolic problems*, Spezialforschungsbereich F003 Optimierung und Kontrolle, Projektbereich Kontinuierliche Optimierung und Kontrolle, Bericht Nr. 153, Graz, 1999.
[22] J. Lumley, *Stochastic Tools in Turbulence*, Academic Press, New York, 1971.
[23] A. Noor, *Recent advances in reduction methods for nonlinear problems*, Comput. & Structures, 13 (1981), pp. 31–44.
[24] A. Okabe, B. Boots, K. Sugihara, and S. Chiu, *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*, 2nd ed., Wiley, New York, 2000.
[25] H. Park and D. Cho, *Low dimensional modeling of flow reactors*, Int. J. Heat Mass Transf., 39 (1996), pp. 3311–3323.
[26] H. M. Park and J. S. Chung, *A sequential method of solving inverse natural convection problems*, Inverse Problems, 18 (2002), pp. 529–546.
[27] H. M. Park and Y. D. Jang, *Control of Burger's equation by means of mode reduction*, Internat. J. Engrg. Sci., 38 (2000), pp. 785–805.
[28] H. M. Park and J. H. Lee, *Solution of an inverse heat transfer problem by means of empirical reduction of modes*, Z. Angew. Math. Phys., 51 (2000), pp. 17–38.
[29] H. M. Park and M. W. Lee, *An efficient method of solving the Navier-Stokes equations for flow control*, Internat. J. Numer. Methods Engrg., 41 (1998), pp. 1133–1151.
[30] H. M. Park and W. J. Lee, *A new numerical method for the boundary optimal control problems of the heat conduction equation*, Internat. J. Numer. Methods Engrg., 53 (2002), pp. 1593–1613.
[31] J. Peterson, *The reduced basis method for incompressible flow calculations*, SIAM J. Sci. Statist. Comput., 10 (1989), pp. 777–786.
[32] M. Rathinam and L. Petzold, *A new look at proper orthogonal decomposition*, to appear.
[33] M. Rathinam and L. Petzold, *Dynamic iteration using reduced order models: A method for simulation of large scale modular systems*, to appear.
[34] S. Ravindran, *Proper orthogonal decomposition in optimal control of fluids*, Int. J. Numer. Methods Fluids, 34 (2000), pp. 425–448.
[35] S. Ravindran, *Reduced-order adaptive controllers for fluid flows using POD*, J. Sci. Comput., 15 (2000), pp. 457–478.
[36] J. Rodríguez and L. Sirovich, *Low-dimensional dynamics for the complex Ginzburg-Landau equations*, Phys. D, 43 (1990), pp. 77–86.
[37] L. Sirovich, *Turbulence and the dynamics of coherent structures*, I-III, Quart. Appl. Math., 45 (1987), pp. 561–590.
[38] N. Smaoui and D. Armbruster, *Symmetry and the Karhunen-Loève analysis*, SIAM J. Sci. Comput., 18 (1997), pp. 1526–1532.
[39] H. Späth, *Cluster Dissection and Analysis, Theory, FORTRAN Programs, Examples*, Ellis Horwood, Chichester, 1985.
[40] D. Sparks, *Algorithm AS 58: Euclidean cluster analysis*, Appl. Statist., 22 (1973), pp. 126–130.
[41] S. Volkwein, *Optimal control of a phase field model using the proper orthogonal decomposition*, ZAMM Z. Angew. Math Mech., 81 (2001), pp. 83–97.
[42] S. Volkwein, *Proper orthogonal decomposition and singular value decomposition*, Spezialforschungsbereich F003 Optimierung und Kontrolle, Projektbereich Kontinuierliche Optimierung und Kontrolle, Bericht Nr. 153, Graz, 1999.

SIAM J. SCI. COMPUT.
Vol. 28, No. 2, pp. 485–497
© 2006 Society for Industrial and Applied Mathematics

RAPID PROLATE PSEUDOSPECTRAL DIFFERENTIATION AND INTERPOLATION WITH THE FAST MULTIPOLE METHOD*

NARAYAN KOVVALI†, WENBIN LIN†, ZHIQIN ZHAO†, LUISE COUCHMAN‡, AND LAWRENCE CARIN§

Abstract. Pseudospectral methods utilizing prolate spheroidal wave functions as basis functions have been shown to possess advantages over the conventional pseudospectral methods based on trigonometric and orthogonal polynomials. However, the spectral differentiation and interpolation steps of the prolate pseudospectral method involve matrix-vector products, which, if evaluated directly, entail $O(N^2)$ memory requirement and computational complexity (where N is the number of unknowns utilized for discretization and interpolation). In this work we show that the fast multipole method (FMM) can be used to reduce the memory requirement and computational complexity of the prolate pseudospectral method to $O(N)$. Example simulation results demonstrate the speed and accuracy of the resulting fast prolate pseudospectral solver.

Key words. prolate spheroidal wave functions, pseudospectral method, differentiation, interpolation, fast multipole method

AMS subject classifications. 65M70, 65N35, 65Y20

DOI. 10.1137/050635961

1. Introduction. Pseudospectral methods [5, 10, 11, 26, 27] are some of the most powerful numerical approaches known for solving ordinary and partial differential equations. The strength of pseudospectral methods is derived from the approximation theory of the basis functions used in the discretization procedure. Traditionally, the basis functions have been chosen from families of trigonometric and orthogonal polynomials and related functions. When the functions involved in the problem are sufficiently smooth, pseudospectral methods require very few unknowns to yield solutions with , , ... accuracy, and have consequently found widespread application in many areas, such as fluid dynamics, quantum mechanics, and electromagnetics.

Recently, prolate spheroidal wave functions [3, 18, 24, 25, 29] of order zero have also been used as basis functions in pseudospectral methods [2, 6, 7, 8, 17]. It has been shown that the prolate basis possesses significant advantages over its polynomial counterparts. The prolate expansion, which is the natural tool for effectively representing bandlimited functions on an interval, seems to be more suitable than the orthogonal polynomials for describing wave phenomena encountered in real-world physics and engineering applications. The prolate pseudospectral method yields spectrally accurate results using even fewer unknowns than conventional pseudospectral methods. In addition, the prolate pseudospectral method utilizes a more uniform spatial grid, which has the advantages of (a) achieving better resolution in the center of the domain and (b) allowing for a larger stable time step in the solution of time-dependent problems. It is also easy to modify an existing pseudospectral code to use the prolate spheroidal wave functions.

*Received by the editors July 13, 2005; accepted for publication (in revised form) September 29, 2005; published electronically March 31, 2006.
http://www.siam.org/journals/sisc/28-2/63596.html
†Department of Electrical and Computer Engineering, Duke University, Durham, NC 27708-0291 (narayan@ee.duke.edu, wl@ee.duke.edu, zzhao@ee.duke.edu).
‡Naval Research Laboratory, Washington, DC 20375 (couchman@nrl.navy.mil).
§Corresponding author. Department of Electrical and Computer Engineering, Duke University, Durham, NC 27708-0291 (lcarin@ee.duke.edu).

Regardless of the basis in use, the spectral differentiation and interpolation steps of pseudospectral methods involve matrix-vector products, which, if evaluated directly, entail $O(N^2)$ memory requirement and computational complexity (where N is the number of unknowns utilized for discretization and interpolation). For the pseudospectral methods based on Fourier and Chebyshev polynomial series, the fast Fourier transform (FFT) and the fast cosine/sine transforms, respectively, can be employed to reduce the aforementioned costs to $O(N \log N)$, provided that all computations are restricted to standard grid positions. When the grid is nonstandard, or the pseudospectral methods are based on Legendre and other orthogonal polynomials and related functions, it has been shown [4, 5, 9] that the fast multipole method (FMM) [12, 13, 15] can be used to reduce the costs to $O(N)$.[1] In this work we show that as for the conventional pseudospectral methods, the FMM can also be used to reduce the memory requirement and computational complexity of the prolate pseudospectral method to $O(N)$. The resulting fast prolate pseudospectral solver is ideally suited for the large-scale problems of physics and engineering applications.

The remainder of the paper is organized as follows. Section 2 reviews the pseudospectral method based on prolate spheroidal wave functions. Section 3 describes the fast prolate pseudospectral differentiation and interpolation with the FMM. Example simulation results are presented in section 4, which is followed by conclusion in section 5.

2. Pseudospectral method based on prolate spheroidal wave functions. In this section, we briefly review the framework of the pseudospectral method based on prolate spheroidal wave functions. For a more detailed discussion, the reader is referred to [6, 7, 8, 17].

2.1. Prolate spheroidal wave functions. Prolate spheroidal wave functions [3, 18, 24, 25, 29] of order zero are the eigenfunctions $\psi_j(x)$ of the Helmholtz equation in prolate spheroidal coordinates

$$\left[(x^2-1)\frac{d^2}{dx^2} + 2x\frac{d}{dx} + c^2x^2\right]\psi_j(x) = \chi_j\psi_j(x), \tag{2.1}$$

where $j = 0, 1, \ldots$ denotes the mode number, χ_j are the associated eigenvalues, and the constant c is known as the "bandwidth parameter."

For any positive real c, the prolate spheroidal wave functions are orthonormal on $[-1, 1]$ with respect to the unit weight function. They form a complete basis of $\mathbf{L^2}[-1, 1]$ and a Chebyshev system [29] on this interval. The wave function $\psi_j(x)$ is symmetric for even j and antisymmetric for odd j, and has exactly j zeros in $[-1, 1]$. When $c = 0$, the prolate spheroidal wave functions are the normalized Legendre functions, i.e., $\psi_j(x)|_{c=0} = \overline{P}_j(x)$. The most efficient procedure known for computing the prolate spheroidal wave functions is a Legendre–Galerkin method [3, 29].

Prolate spheroidal wave functions are the natural tool for representing bandlimited functions on the interval. In [6, 8] it was shown that in the representation of a bandlimited function $f(x)$ of bandwidth W (Fourier transform identically zero for frequency $|\omega| > W$), the expansion coefficients decay exponentially as $j \to \infty$, for $j > W$ when a Chebyshev or Legendre basis is used, and for $j > 2W/\pi$ when the

[1] Optionally, the fast eigenfunction transform of [21] can be used, but this is restricted to the case of basis functions which satisfy three-term recurrence relations (like the orthogonal polynomials, for example).

prolate basis with $c = W$ is used.[2] This means that the error in the representation of $f(x)$ using $N > O(W)$ terms of a truncated series of Chebyshev, Legendre, or prolate functions (with parameter c chosen as the bandwidth W) falls [illegible] with increasing N. More important, the prolate expansion is [illegible] to Chebyshev and Legendre polynomial series in that it requires a factor $\pi/2$ fewer unknowns to achieve spectral accuracy.

The parameter c plays an important role in the approximation theory of the prolate spheroidal wave functions. For a bandlimited function of bandwidth W, the value $c = W$ usually seems to be the best choice for the parameter. For functions which are not bandlimited, the optimum value of c is in general not fixed but can be a function of N. Regardless of whether the function being represented is bandlimited or not, all the useful choices of c must satisfy [6]

$$0 \le c < c_*(N) = (\pi/2)(N + 1/2). \tag{2.2}$$

If c does not satisfy (2.2), all the prolate functions in the truncated series vanish near the end points of the interval and are therefore incapable of representing arbitrary functions. Empirical studies for representative prototype functions in [6] show that c can be chosen about one-half to two-thirds of $c_*(N)$ for moderate N, approaching the limit as N increases. Guidelines on the suitable choice of c can also be found in [2]. A simple procedure for obtaining a good approximation to the optimum value of c in practical applications of the prolate pseudospectral method has been given in [20].

2.2. Prolate pseudospectral method. Owing to their promising approximation properties, prolate spheroidal wave functions of order zero have recently been applied as basis functions in the pseudospectral method [6, 7, 8, 17].

Consider the representative equation

$$Lf(x) = g(x) \tag{2.3}$$

on the domain $x \in [-1, 1]$, where L is a linear differential operator, g is a known function, and f is the unknown function to be determined.[3] Boundary conditions accompany the original equation, and it is assumed that the problem is well-posed.

In order to solve (2.3) numerically, the prolate pseudospectral method effects the following discretization procedure. First, the function f is approximated using the semidiscrete representation

$$f(x) = \sum_{j=0}^{N-1} f_j \, C_j(x), \tag{2.4}$$

where $\{f_j, j = 0, \ldots, N-1\}$ are the expansion coefficients and $\{C_j, j = 0, \ldots, N-1\}$ are "cardinal " [5] functions, which are simply a linear combination of the zero-order prolate spheroidal wave functions $\{\psi_j, j = 0, \ldots, N-1\}$. The cardinal functions are designed to satisfy the interpolation property

$$C_j(x_k) = \delta_{j,k}, \quad j, k = 0, \ldots, N-1, \tag{2.5}$$

[2] Even if the function being represented is not bandlimited, under conditions of sufficient smoothness, spectral accuracy can be achieved in the representation.

[3] For simplicity, we have considered functions of only a single variable. The discussion, however, also applies to functions of many variables.

where δ denotes the Kronecker delta symbol and $\{x_k, k = 0, \ldots, N-1\}$ are appropriately chosen "collocation" [5] positions. The advantage of using the cardinal basis is that the expansion coefficients directly provide us the samples of the function f at the collocation positions, i.e.,

$$f_j = f(x_j), \quad j = 0, \ldots, N-1. \tag{2.6}$$

The spatial grid positions used for collocation in pseudospectral methods are usually chosen as the nodes of a Gaussian quadrature associated with the basis functions being employed. In the prolate pseudospectral method, two important choices for the collocation grid points are the prolate Gauss–Lobatto grid and the prolate-Lobatto grid (sometimes also called the "roots" grid) [6]. Both grids include the end points $x_0 = -1$ and $x_{N-1} = 1$ and have been reported to yield similar performance when used in the pseudospectral method. The prolate pseudospectral solver of this paper utilizes the prolate-Lobatto grid defined as follows.

DEFINITION 2.1 (prolate-Lobatto positions). Let ψ_{N-1} be the $N-1$ [illegible] $x_0, \ldots, x_{N-1} \in [-1, 1]$ [illegible] N [illegible] $s(x) = (1-x^2)\psi'_{N-1}(x)$ [illegible] $x_0 = -1$ $x_{N-1} = 1$

We refer the reader to [17] for details on the efficient computation of this grid.

Substituting the expansion (2.4) into (2.3), we get

$$\sum_{j=0}^{N-1} f_j \, LC_j(x) = g(x). \tag{2.7}$$

This is followed by pseudospectral collocation, leading to the prolate pseudospectral discretization of the original equation (2.3) given by

$$\sum_{j=0}^{N-1} f_j \, LC_j(x_k) = g(x_k), \quad k = 0, \ldots, N-1. \tag{2.8}$$

Writing (2.8) concisely in matrix form, we have

$$\mathbf{L}\mathbf{f} = \mathbf{g}, \tag{2.9}$$

where the vector $\mathbf{f}$ comprises the expansion coefficients $\{f_j, j = 0, \ldots, N-1\}$, the vector $\mathbf{g}$ comprises $\{g(x_k), k = 0, \ldots, N-1\}$, and the matrix $\mathbf{L}$ is a representation of the operator L in the prolate basis with elements given by

$$\mathbf{L}(i, j) = LC_j(x_i), \quad i, j = 0, \ldots, N-1. \tag{2.10}$$

The linear system of equations in (2.9) is solved (in conjunction with any boundary conditions) for the expansion coefficients, and the function f is known using (2.4).

3. Fast prolate pseudospectral differentiation and interpolation with the FMM. In the model problem of the previous section, the storage requirement for the dense matrix $\mathbf{L}$ is $O(N^2)$. Moreover, the direct solution of (2.9) requires computational effort proportional to $O(N^3)$, which quickly becomes prohibitive for large N. It is therefore common to use an iterative solver, which only requires the repeated evaluation of matrix-vector products involving $\mathbf{L}$. Unfortunately, the direct computation of these matrix-vector products is still $O(N^2)$ and, given the fact that

the matrix-vector products must be computed repeatedly, even the cost involved in the iterative solution strategy remains challenging.

The situation is even worse for eigenvalue and time-dependent problems. At each iteration of an eigenvalue solver, a linear system similar in form to the one above must be solved. In time-dependent problems, matrix-vector products involving the dense matrix $\mathbf{L}$ have to be evaluated at each of a usually very large number of time steps.

In [4, 5, 9] it has been shown that, when the expansions are based on orthogonal polynomials and related functions, the structure of the pseudospectral discretization matrix is such that the one-dimensional FMM can be used to accelerate computation of the aforementioned matrix-vector products. In this section we show that the structure of the pseudospectral discretization matrix $\mathbf{L}$ can similarly be exploited to evaluate the matrix-vector products efficiently in $O(N)$ operations via the FMM in one dimension. Without loss of generality, we assume in the following that $\mathbf{L}$ is the matrix representation of a first-derivative operator L in the prolate basis.

The matrix-vector products which appear in the prolate pseudospectral method are of two kinds: (a) differentiation on the prolate-Lobatto grid and (b) interpolation onto a different grid. The prolate pseudospectral differentiation on the prolate-Lobatto grid is given by the matrix-vector product

$$f'(x_k) = \sum_{j=0}^{N-1} f_j \, \mathbf{L}(k,j) = \sum_{j=0}^{N-1} f(x_j) \, C_j'(x_k), \quad k = 0, \ldots, N-1, \tag{3.1}$$

and the prolate pseudospectral interpolation onto a different grid $y_0, \ldots, y_{M-1} \in [-1, 1]$ is represented by the matrix-vector product

$$f(y_i) = \sum_{j=0}^{N-1} f(x_j) \, C_j(y_i), \quad i = 0, \ldots, M-1. \tag{3.2}$$

The cost of evaluating (3.1) and (3.2) using direct matrix-vector multiplication is $O(N^2)$ and $O(MN)$, respectively.

The following set of theorems, which can be thought of as prolate analogues of the observations in [4, 5, 9], provide the analytical framework for accelerating the prolate pseudospectral differentiation and interpolation steps to $O(N)$ and $O(M+N)$, respectively, using the FMM in one dimension.

LEMMA 3.1 (prolate cardinal functions). [illegible] $\{C_j, j = 0, \ldots, N-1\}$ [illegible] (2.5) [illegible] $\{x_k, k = 0, \ldots, N-1\}$ [illegible]

$$C_j(x) = \frac{s(x)}{s'(x_j)(x - x_j)}, \quad j = 0, \ldots, N-1, \tag{3.3}$$

[illegible] $s(x) = (1-x^2)\psi'_{N-1}(x)$ [illegible]. First, $C_j(x_k) = 0$ for $j, k = 0, \ldots, N-1$ and $k \neq j$ because by definition the prolate-Lobatto collocation positions satisfy $\{s(x_k) = 0, k = 0, \ldots, N-1\}$. Moreover, $C_j(x_j) = 1$ for $j = 0, \ldots, N-1$ because $\lim_{x \to x_j} s(x)/(x - x_j) = s'(x_j)$. □

[illegible] 3.2. The cardinal functions in (3.3) are [illegible] linear combinations of the prolate spheroidal wave functions; however, numerical experiments [19] indicate that they yield representations for bandlimited functions which are just as good.

THEOREM 3.3 (prolate pseudospectral differentiation). [illegible] (3.1) [illegible] $\{x_k, k = 0, \ldots,$

$N-1\}$

$$f'(x_k) = \frac{f(x_k)s''(x_k)}{2s'(x_k)} + s'(x_k)\sum_{\substack{j=0\\ j\neq k}}^{N-1} \frac{f(x_j)}{s'(x_j)(x_k-x_j)}, \quad k = 0,\ldots,N-1, \tag{3.4}$$

$s(x) = (1-x^2)\psi'_{N-1}(x)$

. Differentiating (3.3), we note that

$$C_j'(x) = -\frac{C_j(x)}{(x-x_j)} + \frac{s'(x)}{s'(x_j)(x-x_j)},$$

which, combined with the fact that $C_j(x_k) = 0$ for $k \neq j$, readily gives us the second term on the right-hand side of (3.4) upon substitution into (3.1) and pulling $s'(x_k)$ out of the summation over j. For the first term on the right-hand side of (3.4), rewrite the differentiation of (3.3) in the form

$$C_j'(x) = \frac{1}{s'(x_j)}\left[\frac{s'(x)(x-x_j) - s(x)}{(x-x_j)^2}\right],$$

take the limit $x \to x_j$, and finally substitute into (3.1). □

THEOREM 3.4 (prolate pseudospectral interpolation).

(3.2)

$y_0,\ldots,y_{M-1} \in [-1,1]$

$\{x_k, k = 0,\ldots,N-1\}$

$$f(y_i) = s(y_i)\sum_{j=0}^{N-1} \frac{f(x_j)}{s'(x_j)(y_i-x_j)}, \quad i = 0,\ldots,M-1, \tag{3.5}$$

$s(x) = (1-x^2)\psi'_{N-1}(x)$

. Use (3.3) for the cardinal functions in (3.2) and pull $s(y_i)$ out of the summation over j. □

3.5. It is well known [9] that numerical pseudospectral differentiation is an ill-conditioned procedure. In fact, for the pseudospectral method based on Chebyshev polynomials, the condition number has been shown [14] to be proportional to N^2 (where N is the size of the differentiation matrix). The prolate pseudospectral method may well suffer from the same problem; however, preliminary experiments [2, 7, 19, 29] seem to indicate that the conditioning may be better than that of Chebyshev differentiation. While this is an important issue and needs further examination, a careful analysis is beyond the scope of this article.

The fast algorithms of this section exploit the form of the expressions in (3.4) and (3.5) for evaluating them efficiently using the FMM in one dimension. A formal description of the algorithms for fast prolate pseudospectral differentiation and interpolation now follows. It is assumed below that the vectors $\{x_k, s'(x_k), s''(x_k), \alpha_k = s''(x_k)/(2s'(x_k)), k = 0,\ldots,N-1\}$ and $\{s(y_i), i = 0,\ldots,M-1\}$ have been precomputed and stored.

ALGORITHM 3.1 (fast prolate pseudospectral differentiation with the FMM).

Input: Function values at prolate-Lobatto points $\{f(x_k), k = 0,\ldots,N-1\}$.
Output: Function derivative at prolate-Lobatto points $\{f'(x_k), k = 0,\ldots,N-1\}$.

Step 1

Comment[Initialization.]
Initialize FMM (all calculations independent of $f(x_k)$ performed here).
Complexity: $O(N)$

Step 2

Comment[Compute charge strengths for the FMM.]
do $j = 0, \ldots, N-1$
  Compute $\beta_j = f(x_j)/s'(x_j)$.
end do
Complexity: $O(N)$

Step 3

Comment[Compute contributions from off-diagonal entries of the differentiation matrix efficiently using the FMM in one dimension.]
Compute $\gamma_k = \sum_{j=0, j\neq k}^{N-1} \frac{\beta_j}{(x_k - x_j)}$, $k = 0, \ldots, N-1$, using the FMM.
Complexity: $O(N)$

Step 4

Comment[Add contribution from the diagonal entries of the differentiation matrix to finish the calculation.]
do $k = 0, \ldots, N-1$
  Compute $f'(x_k) = f(x_k) \cdot \alpha_k + s'(x_k) \cdot \gamma_k$.
end do
Complexity: $O(N)$

ALGORITHM 3.2 (fast prolate pseudospectral interpolation with the FMM).

Input: Function values at prolate-Lobatto points $\{f(x_k),\ k = 0, \ldots, N-1\}$.
Output: Prolate interpolant $\{f(y_i),\ i = 0, \ldots, M-1\}$.

Step 1

Comment[Initialization.]
Initialize FMM (all calculations independent of $f(x_k)$ performed here).
Complexity: $O(M+N)$

Step 2

Comment[Compute charge strengths for the FMM.]
do $j = 0, \ldots, N-1$
  Compute $\beta_j = f(x_j)/s'(x_j)$.
end do
Complexity: $O(N)$

Step 3

Comment[Compute the interpolation matrix contributions efficiently using the FMM in one dimension.]
Compute $\gamma_i = \sum_{j=0}^{N-1} \frac{\beta_j}{(y_i - x_j)}$, $i = 0, \ldots, M-1$, using the FMM.
Complexity: $O(M+N)$

Step 4
Comment[Scale contributions to finish the calculation.]
do $i = 0, \ldots, M-1$
Compute $f(y_i) = s(y_i) \cdot \gamma_i$.
end do
Complexity: $O(M)$

The running times (and storage requirements) of Algorithms 3.1 and 3.2 are $O(N)$ and $O(M+N)$, respectively. The reader will note that the differentiation and interpolation matrices were never explicitly constructed; instead, the interactions have been evaluated efficiently using the FMM. Also, when the prolate pseudospectral differentiation or interpolation is required for many different sets of function values $\{f(x_j),\, j = 0, \ldots, N-1\}$ on the prolate-Lobatto grid positions $\{x_j,\, j = 0, \ldots, N-1\}$, the initialization steps have to be performed only once.

4. Simulation results. In this section, we present numerical simulation results. We first show results of experiments with the FMM and follow this with the solution of an example problem using the FMM-accelerated prolate pseudospectral method. All calculations were carried out in double-precision on a 3.06 GHz Intel Xeon processor.

4.1. Numerical experiments with the FMM. We consider the problem of computing the summations

$$f(x_j) = \sum_{\substack{k=1 \\ k \neq j}}^{N} \frac{q_k}{x_j - x_k}, \quad j = 1, \ldots, N,$$

with $\{x_k, k = 1, \ldots, N\}$ chosen as the Chebyshev Gauss–Lobatto positions

$$x_k = -\cos\left(\frac{k-1}{N-1} \cdot \pi\right), \quad k = 1, \ldots, N,$$

and $\{q_k, k = 1, \ldots, N\}$ being uniform random numbers in $[0, 1]$. Tables 4.1 and 4.2 show comparisons of the CPU-time expended in the computation by the FMM of Algorithm 3.1 in [9] and the direct calculation, for different values of N. The timing for the FMM has been split into two parts: the initialization step and the evaluation step.[4] The relative error in the results (in the l^∞ and l^2 norms) is also listed in the tables. We see that the CPU-time for the FMM computation grows only linearly, as opposed to the quadratic growth of that for the direct calculation. Moreover, if we consider only the CPU-time for the evaluation step, the FMM is almost always more efficient than the direct calculation. We also note that $p = 16$ terms in the Chebyshev expansions of the FMM seem to be sufficient for achieving near double-precision accuracy in the result, with $p = 7$ yielding near single-precision accuracy.

4.1. While the FMM and direct matrix-vector multiplication are, respectively, $O(N)$ and $O(N^2)$ in complexity, the proportionality constants of these costs are somewhat implementation-dependent. It is possible, for example, to decrease the constants (for both methods) by exploiting in the computations involved [5].

[4]The reason for doing this is that in the applications of interest here the summations must usually be computed many times, with $\{x_k, k = 1, \ldots, N\}$ remaining the same but $\{q_k, k = 1, \ldots, N\}$ changing between successive computations. The computations which depend only on $\{x_k, k = 1, \ldots, N\}$ have been collected into the initialization step which therefore needs to be carried out only *once*, while the evaluation step has to be repeated for every different $\{q_k, k = 1, \ldots, N\}$. The point is that it is often the evaluation step of the FMM which governs the total time complexity.

TABLE 4.1
Numerical experiment with the FMM (Algorithm 3.1 *of* [9]*, with* $p = 16$ *and* $s = 32$*).*

N	Relative errors		Timings (milliseconds)		
	l^∞	l^2	FMM init.	FMM eval.	Direct calc.
128	1.3E-14	4.6E-14	1.07	0.29	0.21
256	1.1E-14	6.0E-14	1.68	0.73	0.83
512	4.0E-15	2.8E-14	2.86	1.66	3.30
1024	2.9E-15	4.5E-14	5.26	3.60	13.13
2048	1.1E-14	1.4E-13	10.10	7.77	52.42
4096	3.2E-15	7.8E-14	19.70	16.40	209.50

TABLE 4.2
Numerical experiment with the FMM (Algorithm 3.1 *of* [9]*, with* $p = 7$ *and* $s = 14$*).*

N	Relative errors		Timings (milliseconds)		
	l^∞	l^2	FMM init.	FMM eval.	Direct calc.
128	7.6E-07	3.8E-06	0.20	0.18	0.21
256	5.6E-07	4.5E-06	0.35	0.40	0.83
512	2.6E-07	3.0E-06	0.66	0.85	3.30
1024	1.9E-07	2.9E-06	1.29	1.81	13.13
2048	4.8E-07	5.3E-06	2.55	3.79	52.42
4096	1.2E-07	2.7E-06	5.21	8.20	209.50

4.2. FMM-accelerated prolate pseudospectral method for the scalar wave equation. Consider the first-order hyperbolic initial boundary value problem

$$\begin{aligned} &u_t = c_0 u_x, && x \in [-1,1], \quad t > 0, \\ &u(x,0) = f(x), && x \in [-1,1], \\ &u(1,t) = g(t), && t \geq 0. \end{aligned} \tag{4.1}$$

The inflow function $g(t)$ is chosen as $f(1 + c_0 t)$, and the exact solution is given by $u(x,t) = f(x + c_0 t)$. We choose $f(x) = \cos^2(\frac{251\pi}{2}x)$ and the wave velocity as $c_0 = 0.05$. Note the high value of the wave number, selected to mimic the situation in large problems.

We solve (4.1) numerically using prolate pseudospectral differentiation in space and the third-order Adams–Bashforth (AB-3) formula [26] in time:

$$\mathbf{v}^{n+1} = \mathbf{v}^n + \frac{1}{12} c_0 \Delta t D_N^P (23\mathbf{v}^n - 16\mathbf{v}^{n-1} + 5\mathbf{v}^{n-2}), \tag{4.2}$$

where $\mathbf{v}^n$ denotes the numerical solution at the nth time step, Δt is the size of the time step, and D_N^P represents the prolate pseudospectral differentiation matrix of size N. The exact solution is used for the three initial AB-3 time steps. For the prolate differentiation matrix, we use bandwidth parameter $c = 790$, which is close to the bandwidth 251π of the bandlimited function $f(x)$, and size $N = 575$ to achieve a relative error of about 10^{-11} in the spatial derivatives. The inflow boundary condition at $x = 1$ is imposed strongly. For the FMM-accelerated prolate pseudospectral method, $p = 16$ terms were used in the Chebyshev expansions of the FMM to retain near double-precision accuracy.

Tables 4.3 and 4.4 show the number of time steps N_{tstep}, running times, and errors in the computed solutions at the grid positions at final time $t_f = 2$ from the direct and FMM-accelerated prolate pseudospectral methods, as a function of Δt. For each row, Δt is decreased by a factor of 5. The following observations can be made:

TABLE 4.3
Direct prolate pseudospectral method for the scalar wave equation (4.1).

Δt	N_{tstep}	Direct Prolate		
		Timings (secs)	Relative errors	
			l^∞	l^2
5.0E-03	400	1.75	1.0E+120	7.6E+118
1.0E-03	2000	8.77	9.1E-04	1.0E-03
2.0E-04	10000	43.6	7.3E-06	8.1E-06
4.0E-05	50000	218.6	5.8E-08	6.5E-08
8.0E-06	250000	1092.7	4.6E-10	5.2E-10
1.6E-06	1250000	5442.1	4.0E-12	4.1E-12

TABLE 4.4
FMM-accelerated prolate pseudospectral method (with $p = 16$) for the scalar wave equation (4.1).

Δt	N_{tstep}	FMM Prolate		
		Timings (secs)	Relative errors	
			l^∞	l^2
5.0E-03	400	0.69	8.0E+119	7.7E+118
1.0E-03	2000	3.45	9.1E-04	1.0E-03
2.0E-04	10000	17.3	7.3E-06	8.1E-06
4.0E-05	50000	86.5	5.8E-08	6.5E-08
8.0E-06	250000	431.3	4.7E-10	5.2E-10
1.6E-06	1250000	2167.0	1.0E-11	5.4E-12

- Since N_{tstep} increases by the same factor, the running times (for both methods) also increase fivefold along each row.
- The extremely large errors in the first row are explained by the fact that the discretization (4.2) is unstable for $\Delta t = 5 \times 10^{-3}$. Figure 4.1 confirms this with plots of the stability region for AB-3 time stepping and eigenvalues of the modified prolate differentiation matrix (scaled by $c_0 \Delta t$) for pseudospectral collocation with the strongly imposed boundary condition.[5] As seen from Figure 4.1, for the choice $\Delta t = 5 \times 10^{-3}$ some of the eigenvalues lie outside the stability region and the numerical methods are consequently unstable. Even though the differentiation matrix is not explicitly constructed in the FMM-accelerated method, the "eigenvalues" have their effect. The situation is remedied by choosing $\Delta t = 1 \times 10^{-3}$ in the second row for which the eigenvalues lie inside the stability region and the methods are now stable.[6] For the smaller values of Δt in the remaining rows, stability is retained.
- The errors decrease by factors of 125 along the rows, which is attributed to the third-order accuracy of the AB-3 time stepping (reduction in Δt by a factor of 5 reduces the errors $5^3 = 125$ times). This indicates that it is the error due to temporal discretization which is dominating in these results; the spatial derivatives are very accurate. This pattern is followed until $\Delta t = 1.6 \times 10^{-6}$ in the last row for which the time discretization error is now of the same order as the error in the spatial derivatives. For the FMM-accelerated prolate pseudospectral method, the additional errors introduced by the FMM start becoming important.

[5] It is worth noting here that the eigenvalues of the modified prolate differentiation matrix are known to have negative real parts [8].

[6] It has been shown that eigenvalue analysis only provides necessary conditions for Lax-stability; the necessary and sufficient conditions are obtained using ε-pseudoeigenvalues [16, 22, 23].

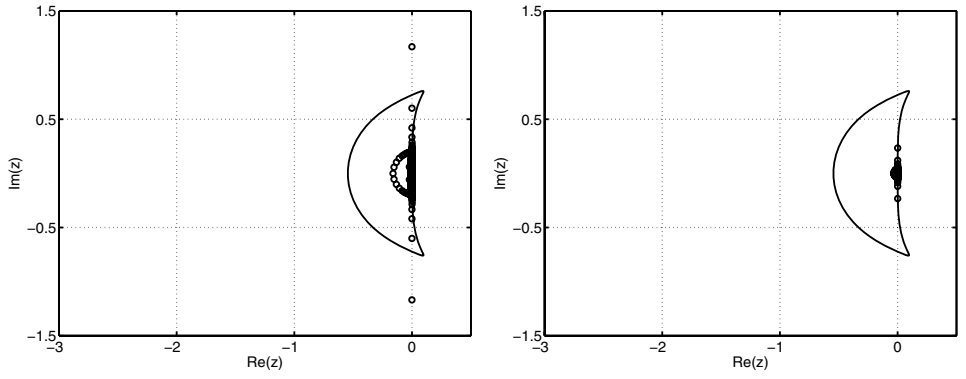


FIG. 4.1. *Stability region for AB-3 time stepping and eigenvalues of the modified prolate differentiation matrix* D_N^P *in* (4.2) *(scaled by* $c_0\Delta t$*) for pseudospectral collocation with a strongly imposed boundary condition.* (a) $\Delta t = 5 \times 10^{-3}$ *(unstable);* (b) $\Delta t = 1 \times 10^{-3}$ *(stable).*

- Finally, the point of this exercise. The FMM-accelerated prolate pseudospectral solver is nearly 3 times faster than the direct prolate pseudospectral method. This figure is consistent with the savings afforded by FMM over direct matrix-vector multiplications for the choice of $N = 575$ (see Table 4.1). Importantly, as opposed to the $O(N^2)$ direct prolate pseudospectral method, the FMM-accelerated prolate pseudospectral solver is an $O(N)$ method. The factor of 3 savings in running time is for our example problem with $N = 575$ unknowns. [illegible] N [illegible]

In the results shown in Table 4.4 for the FMM-accelerated prolate pseudospectral method, the number of terms used in the Chebyshev expansions of the FMM was kept fixed at $p = 16$, ensuring approximately double-precision accuracy. However, as seen from the values of the errors in Tables 4.3 and 4.4, the large errors due to temporal discretization are dominating the accuracy of the results. It therefore seems to be unnecessary to carry out the FMM computations to such high precision. Table 4.5 shows the results of repeating the above experiment for the FMM-based prolate pseudospectral method with p now being varied as a function of the time step Δt. In each case, the value of p is selected according to the accuracy allowed by Δt.

From the results in Table 4.5 we see that, as expected, the errors in the solutions are only slightly affected by this decrease in the precision of the FMM. Note that if p were to be decreased further, the error in the FMM would dominate. The increase in running times along the rows is now more than fivefold. This is because even though N_{tstep} increases only by factors of 5, p is also increasing along the rows. However, with number of terms p used in the FMM expansions changing with Δt, the FMM-accelerated prolate pseudospectral solver is now nearly 3 to 5 times faster than the direct prolate pseudospectral method, with the savings being highest for larger Δt (smaller p). This speedup will increase with the number of unknowns.

[illegible] 4.2. A note on the stability of the AB-3 time stepping in the FMM-accelerated prolate pseudospectral solver for the choice $p = 7$. It is well known that the

TABLE 4.5
FMM-accelerated prolate pseudospectral method (p changing with Δt) for the scalar wave equation (4.1).

		FMM Prolate			
Δt	N_{tstep}	p	Timings (secs)	Relative errors	
				l^∞	l^2
5.0E-03	400	7	0.36	8.0E+119	7.7E+118
1.0E-03	2000	7	1.79	9.4E-04	1.0E-03
2.0E-04	10000	9	9.80	8.8E-06	8.2E-06
4.0E-05	50000	11	58.1	9.0E-08	7.0E-08
8.0E-06	250000	13	349.9	1.6E-09	9.0E-10
1.6E-06	1250000	15	2053.9	4.0E-11	1.8E-11

stability limit for explicit time stepping in pseudospectral solvers based on Chebyshev and Legendre polynomials is sensitive to the precision with which the computations are carried out [28].[7] The relevant question now is: How are the eigenvalues of the FMM-based (implicit) prolate pseudospectral differentiation matrix D_N^P affected by single-precision accuracy in its representation? The answer is that the eigenvalues are undoubtedly affected by the lower accuracy of the differentiation matrix, but not significantly enough to change the trend in Figure 4.1. This has been verified by examination of its spectrum (not shown here) and is confirmed by the results in Table 4.5.

5. Conclusion. In this paper we have presented fast algorithms for differentiation and interpolation in the pseudospectral method based on prolate spheroidal wave functions. Our approach is motivated by [4, 5, 9], where similar schemes have been developed for conventional pseudospectral methods based on orthogonal polynomials and related functions. Numerical results show that, as compared to the direct prolate pseudospectral method, the FMM-accelerated prolate pseudospectral solver shows significant improvement in running times. For large-scale problems where the number of unknowns is high, the savings are expected to be significant.

Several open questions remain. First, what is the conditioning of prolate pseudospectral differentiation? Our experiments indicate that the conditioning may be better than that of Chebyshev pseudospectral differentiation (which has a condition number proportional to N^2, where N is the size of the differentiation matrix). Second, the FFT is known to improve the numerical stability of Chebyshev differentiation [1]; it is not yet clear what role the FMM-based fast algorithm has to play in this context. A related observation is that, unlike the eigenvalues of the Chebyshev and Legendre differentiation matrices, those of the prolate differentiation matrix do not seem to be highly precision-sensitive. These issues are currently under investigation and results will be reported at a later date.

Acknowledgments. The authors thank Professor Leslie Greengard for several pointers to the FMM literature and Dr. Qian-Yong Chen for useful discussions. The authors are also grateful to the anonymous reviewers for their valuable comments.

[7] The differentiation matrices involved in these methods tend to be ill-conditioned and have eigenvalues which are strongly affected by the working precision. This has led to the concept of "precision-dependent stability" in explicit time stepping schemes used with these methods (see [28] for more details).

REFERENCES

[1] R. Baltensperger and M. R. Trummer, *Spectral differencing with a twist*, SIAM J. Sci. Comput., 24 (2003), pp. 1465–1487.

[2] G. Beylkin and K. Sandberg, *Wave propagation using bases for bandlimited functions*, Wave Motion, 41 (2005), pp. 263–291.

[3] C. J. Bouwkamp, *On spheroidal wave functions of order zero*, J. Math. Phys., 26 (1947), pp. 79–92.

[4] J. P. Boyd, *Multipole expansions and pseudospectral cardinal functions: A new generalization of the fast Fourier transform*, J. Comput. Phys., 103 (1992), pp. 184–186.

[5] J. P. Boyd, *Chebyshev and Fourier Spectral Methods*, 2nd ed., Dover, New York, 2001.

[6] J. P. Boyd, *Prolate spheroidal wavefunctions as an alternative to Chebyshev and Legendre polynomials for spectral element and pseudospectral algorithms*, J. Comput. Phys., 199 (2004), pp. 688–716.

[7] J. P. Boyd, *Algorithm 840: Computation of grid points, quadrature weights and derivatives for spectral element methods using prolate spheroidal wave functions—Prolate elements*, ACM Trans. Math. Software, 31 (2005), pp. 149–165.

[8] Q.-Y. Chen, D. Gottlieb, and J. S. Hesthaven, *Spectral methods based on prolate spheroidal wave functions for hyperbolic PDEs*, SIAM J. Numer. Anal., 43 (2005), pp. 1912–1933.

[9] A. Dutt, M. Gu, and V. Rokhlin, *Fast algorithms for polynomial interpolation, integration, and differentiation*, SIAM J. Numer. Anal., 33 (1996), pp. 1689–1711.

[10] B. Fornberg, *A Practical Guide to Pseudospectral Methods*, Cambridge University Press, Cambridge, UK, 1998.

[11] D. Gottlieb and S. A. Orszag, *Numerical Analysis of Spectral Methods: Theory and Applications*, CBMS-NSF Regional Conf. Ser. in Appl. Math. 26, SIAM, Philadelphia, 1977.

[12] L. Greengard, *The Rapid Evaluation of Potential Fields in Particle Systems*, MIT Press, Cambridge, MA, 1988.

[13] L. Greengard, *The numerical solution of the N-body problem*, Computers in Physics, 4 (1990), pp. 142–152.

[14] L. Greengard, *Spectral integration and two-point boundary value problems*, SIAM J. Numer. Anal., 28 (1991), pp. 1071–1080.

[15] L. Greengard and V. Rokhlin, *A fast algorithm for particle simulations*, J. Comput. Phys., 73 (1987), pp. 325–348.

[16] D. J. Higham and L. N. Trefethen, *Stiffness of ODEs*, BIT, 33 (1993), pp. 285–303.

[17] N. Kovvali, W. Lin, and L. Carin, *Pseudospectral method based on prolate spheroidal wave functions for frequency-domain electromagnetic simulations*, IEEE Trans. Antennas and Propagation, 53 (2005), pp. 3990–4000.

[18] H. J. Landau and H. O. Pollak, *Prolate spheroidal wave functions, Fourier analysis and uncertainty* II, Bell System Tech. J., 40 (1961), pp. 65–84.

[19] W. Lin, *Theory and Applications of Biorthogonal Ridgelets and Prolate Spheroidal Wave Functions*, Ph.D. thesis, Duke University, Durham, NC, 2005.

[20] W. Lin, N. Kovvali, and L. Carin, *A practical procedure for finding the bandwidth parameter in the prolate pseudospectral method*, J. Comput. Phys., submitted.

[21] S. A. Orszag, *Fast eigenfunction transforms*, in Science and Computers, G. C. Rota, ed., Academic Press, New York, 1986, pp. 23–30.

[22] S. C. Reddy and L. N. Trefethen, *Lax-stability of fully discrete spectral methods via stability regions and pseudo-eigenvalues*, Comput. Methods Appl. Mech. Engrg., 80 (1990), pp. 147–164.

[23] S. C. Reddy and L. N. Trefethen, *Stability of the method of lines*, Numer. Math., 62 (1992), pp. 235–267.

[24] D. Slepian, *Some comments on Fourier analysis, uncertainty, and modeling*, SIAM Rev., 25 (1983), pp. 379–393.

[25] D. Slepian and H. O. Pollak, *Prolate spheroidal wave functions, Fourier analysis, and uncertainty* I, Bell System Tech. J., 40 (1961), pp. 43–63.

[26] L. N. Trefethen, *Finite Difference and Spectral Methods for Ordinary and Partial Differential Equations*, http://web.comlab.ox.ac.uk/oucl/work/nick.trefethen/pdetext.html (1996).

[27] L. N. Trefethen, *Spectral Methods in MATLAB*, Software Environ. Tools 10, SIAM, Philadelphia, 2000.

[28] L. N. Trefethen and M. R. Trummer, *An instability phenomenon in spectral methods*, SIAM J. Numer. Anal., 24 (1987), pp. 1008–1023.

[29] H. Xiao, V. Rokhlin, and N. Yarvin, *Prolate spheroidal wavefunctions, quadrature, and interpolation*, Inverse Problems, 17 (2001), pp. 805–838.

SIAM J. SCI. COMPUT.
Vol. 28, No. 2, pp. 498–516
© 2006 Society for Industrial and Applied Mathematics

AN INTERFACE-STRIP DOMAIN DECOMPOSITION PRECONDITIONER*

ALFIO QUARTERONI†, MARZIO SALA‡, AND ALBERTO VALLI§

Abstract. In this paper we consider domain decomposition preconditioners based on a vertex-oriented (VO) decomposition of the computational domain. In element-oriented (EO) decompositions, each element of the grid belongs to a different domain, while in VO decompositions each vertex belongs to a different subdomain. Based on VO decompositions, we present some preconditioners for the solution of the original system, as well as for that of the Schur complement system. Theoretical properties are investigated for a finite element approximation of an elliptic problem. Numerical results and comparison with state-of-the-art preconditioners are also reported. The numerical results presented here show the effectiveness of the proposed preconditioners and their good scalability properties.

Key words. parallel preconditioner, domain decomposition, Schur complement system

AMS subject classifications. 65N55, 65N30, 35J25, 35J20

DOI. 10.1137/04061057X

1. Introduction. We consider the iterative solution by Krylov methods of sparse linear systems of type

$$A\mathbf{u} = \mathbf{f}, \tag{1.1}$$

where $A \in \mathbb{R}^{n\times n}$ is a nonsingular matrix, arising from the discretization of a boundary value problem for partial differential equations (PDEs), $\mathbf{u} \in \mathbb{R}^n$ is the solution vector, and $\mathbf{f} \in \mathbb{R}^n$ is the right-hand side of the known terms.

The convergence rate of the adopted Krylov method typically depends on the spectral properties of A, mainly on its condition number $\kappa(A)$. The latter is defined as $|||A|||\cdot|||A^{-1}|||$ (for a suitable matrix norm $|||\cdot|||$), or simply as $\lambda_{\max}(A)/\lambda_{min}(A)$ if the matrix is symmetric and positive definite. Since the matrix A in (1.1) is typically ill-conditioned (see, e.g., [21]), one most often resorts to solving the preconditioned system

$$P^{-1}A\mathbf{u} = P^{-1}\mathbf{f}, \tag{1.2}$$

where P is a nonsingular matrix called the [illegible]. The operator P should approximate A^{-1} as closely as possible so that $\kappa(P^{-1}A) \ll \kappa(A)$, while still being reasonably cheap to invert. Ideally, P should be [illegible]; i.e., $\kappa(P^{-1}A)$ ought not depend on the mesh size, and, when considering parallel preconditioners, a further requirement for P is that it is [illegible]; i.e., $\kappa(P^{-1}A)$ should not depend on the number of processors used in the computations.

In this paper, we address parallel preconditioners issuing from domain decomposition (DD) ideas [27, 22]. DD is currently a very active area of research in the

*Received by the editors June 25, 2004; accepted for publication (in revised form) October 28, 2005; published electronically April 7, 2006.
http://www.siam.org/journals/sisc/28-2/61057.html
†CMCS-IACS, EPFL, CH-1015 Lausanne, Switzerland, and MOX, Politecnico di Milano, I-20133 Milano, Italy (alfio.quarteroni@epfl.ch).
‡COLAB, ETHZ, CH-8092 Zürich, Switzerland (msala@sandia.gov).
§Dipartimento di Matematica, Università di Trento, I-38050 Povo (Trento), Italy (valli@science.unitn.it).

field of numerical approximation of boundary value problems for PDEs. The key idea of (homogeneous) DD techniques is to split the original computational domain, say Ω, into smaller components $\Omega_i, i = 1, \ldots, M$ (which may or may not overlap), called subdomains, and then solve the boundary value problem on each subdomain. Clearly, additional interface conditions have to be imposed on $\partial\Omega_i \setminus \partial\Omega$. Usually, the original global problem is reformulated as an iterative procedure on the subdomains.

Two approaches can be followed to define DD preconditioners. In [illegible], local Dirichlet-type problems are solved on each Ω_i. Here the communication between the solutions on the different subdomains is guaranteed by the overlapping regions. This procedure, often referred to as the one-level Schwarz method, is not scalable. In fact, the information exchange among the subdomains is only through the overlapping regions. A good scalability may be recovered by the addition of a coarse correction, whose role is to spread out information among distant subdomains. On the other side, [illegible] are usually reformulated in terms of the so-called Schur complement (SC) system. The idea is to partition the set of nodes assigned to each subdomain into the sets of boundary nodes and internal nodes. This latter set can be "condensed." Thus, instead of solving a linear system with matrix A, one can solve a linear system with the SC matrix S. If A is a symmetric and positive definite matrix, it is possible to show that $\kappa(S) \leq \kappa(A)$; see [1].

The very first step in both overlapping and nonoverlapping DD approaches is the decomposition of the computational domain Ω into the subdomains. Here, we analyze two different strategies that can be used to define this decomposition. Let $\Omega \subset \mathbb{R}^d$, $d = 2, 3$, be a polygonal domain, let $\mathcal{T}_h$ be a triangulation made of polygons or polyhedra, and, for the sake of simplicity, assume that the boundary of Ω coincides with that of the triangulation.

We consider the following two cases. In the first approach, $\mathcal{T}_h$ is partitioned into M nonoverlapping parts, namely $\mathcal{T}_{h,1}$, $\mathcal{T}_{h,2}, \ldots, \mathcal{T}_{h,M}$. For each $i = 1, \ldots, M$ we associate to $\mathcal{T}_{h,i}$ a subdomain Ω_i, formed by the interior of the union of the elements of $\mathcal{T}_{h,i}$, while Ω_Γ is composed of a finite number of $d-1$ manifolds; see the left-hand side of Figure 1.1 for an example in the case $M = 2$. This situation represents the common case where Ω_Γ is the union of the part of the boundary of Ω_i that is internal to Ω. This type of decomposition is called [illegible] (EO) because each element of $\mathcal{T}_h$ belongs exclusively to one of the M subdomains Ω_i.

An alternative approach consists of partitioning Ω into $M + 1$ nonoverlapping parts $\mathcal{T}_{h,1}$, $\mathcal{T}_{h,2}, \ldots, \mathcal{T}_{h,M}$, and $\mathcal{T}_{h,\Gamma}$. By construction, the portion of space Ω_Γ of which $\mathcal{T}_{h,\Gamma}$ is a triangulation is now formed by the union of a finite number of "strips" lying between the Ω_i. This decomposition is called [illegible] (VO) because each vertex belongs exclusively to one of the M subdomains Ω_i. This time Ω_Γ is a d-dimensional domain as well (see the right-hand side of Figure 1.1).

Note that, by construction, in an EO decomposition all the vertices on Ω_Γ are shared by two or more subdomains Ω_i, while in a VO decomposition they belong to exactly two subdomains: Ω_Γ and one of the Ω_i.

In this paper we deal with VO decompositions. Indeed, VO decompositions are adopted by several state-of-the-art parallel linear algebra packages. From the point of view of parallel computing, a distinguishing feature of VO decompositions is that the transition region Ω_Γ can be replicated among the processors which hold the Ω_i, which provides a way for data communication. In addition, using a VO approach one may derive the global SC operator from the assembled global matrix A, while adopting

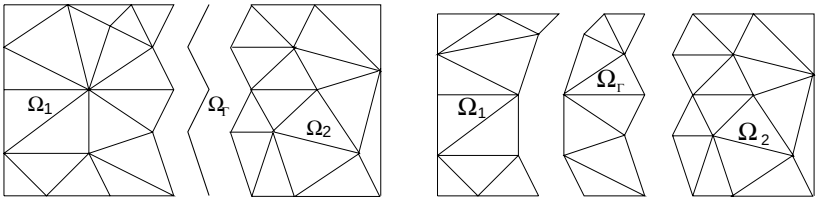


FIG. 1.1. *Example of EO (left) and VO (right) decomposition.*

an EO decomposition would require working at the level of the subdomain assembly process.

In this paper we are concerned with the determination of efficient preconditioners for VO decompositions. Our contribution is twofold. First, we detail how to derive from a VO decomposition a two-level overlapping DD preconditioner that does not require a coarse space. Indeed, for problems arising from PDEs, a typical choice for defining the coarse correction is to solve the original PDE problem on a coarse grid. However, the construction of a coarse grid and that of the corresponding restriction and extension operators may be difficult or computationally expensive on general unstructured grids and arbitrary geometries. In addition, for a general, possibly nonconvex domain it is difficult to ensure that the boundary conditions are correctly represented on the coarse level.

To overcome these difficulties, we show how the VO decomposition easily allows the definition of a coarse operator based on aggregation procedures. The latter have been introduced and analyzed in [29] and [3], and then used for the shallow water equations and two-dimensional (2D) potential flows [20], for three-dimensional (3D) potential flow computations [9], for groundwater flows [13], for multiphase flows [15], for discontinuous Galerkin approximation of advection-diffusion problems [16], and for 3D compressible Euler equations on unstructured tetrahedral grids [24, 10], to mention some uses.

The second, most important, contribution of this paper is the definition of a novel preconditioner, which reformulates classical EO schemes in the VO framework. We detail a new Dirichlet–Neumann DD preconditioner that we call , based on a VO decomposition like that presented in the right-hand side of Figure 2.1. The Dirichlet step is performed in parallel in the domains Ω_i, while the Neumann step is performed in the domain Ω_Γ. When considering the stiffness matrix derived by the finite element approximation of the Laplace operator in two or three dimensions, the condition number of the preconditioned SC turns out to be $O(H/h)$, H being a bound for the diameter of Ω_i and h being the mesh size (therefore, the condition number gets better as the number of subdomains becomes larger, a property that is interesting for parallel numerical computation).

A similar asymptotic behavior of the condition number is proved to hold for other diagonal or diagonal-like preconditioners, as in the case of the SC after condensation [2], the Schwarz method with minimum overlap [6], and the FETI method with lumping [8]. We point out that this estimate is less favorable than others, such as those obtained for the balancing Neumann–Neumann method or the FETI method. However, our algorithm is (it does not necessarily require the introduction of an explicit coarse space), and our preconditioner is , as only one Neumann solver in the domain Ω_Γ (which contains very few mesh nodes) is needed. This last problem is global and, to some extent, is a surrogate for coarse correction. For these reasons,

it turns out that the performance of our method is comparable to that realized by more sophisticated two-level methods in which the coarse space is indeed explicitly introduced, and some presented numerical results confirm this claim.

Finally, another important feature of our algorithm is that the condition number of the preconditioned Schur complement is [illegible] in the coefficients of the elliptic operator.

The paper is organized as follows. In section 2 we introduce the algebraic formalism for EO and VO decompositions. Linear system (1.1) is reformulated for both the EO and the VO approach, pointing out the main differences. The EO approach has been extensively investigated in the literature (see, for instance, [4, 18, 27, 22]); hence, in section 3 we present several preconditioners for the VO decomposition. The solutions of both the original system and the SC system are considered. For the latter, an $O(H/h)$ theoretical estimate of the condition number is obtained. Numerical results are presented in section 4, while in section 5 we draw some conclusions.

2. Element-oriented and vertex-oriented decompositions. The difference between EO and VO decompositions can be easily appreciated in the case of two subdomains. Therefore, unless otherwise stated, in the following discussion we consider a decomposition in two subdomains Ω_1 and Ω_2, so that $\overline{\Omega} = \overline{\Omega_1} \cup \overline{\Omega_2} \cup \overline{\Omega_\Gamma}$.

We need to introduce some notation. The nodes on $\overline{\Omega_\Gamma}$ will be called [illegible] nodes, and in particular those on $\overline{\Omega_i} \cap \overline{\Omega_\Gamma}$ are the border nodes of the domain Ω_i. A node of $\overline{\Omega_i}$ which is not a border node is called [illegible] to Ω_i, $i = 1, \ldots, M$.

For vectors, we will consistently use the subscripts I and B to indicate internal and border, respectively, while the superscript (i) will denote the domain to which we are referring. Thus, $\mathbf{u}_I^{(i)}$ will indicate the vector of unknowns associated to nodes internal to Ω_i, while $\mathbf{u}_B^{(i)}$ is associated to the border nodes. The dimensions of these vectors will be denoted $n_I^{(i)}$ and $n_B^{(i)}$, respectively. Unless otherwise specified, for matrices we will use the superscript (i) to indicate the subdomain to which we are referring.

2.1. Element-oriented decompositions. In the case of two subdomains, one can rewrite (1.1) so that the nodes associated to Ω_1 are ordered first, followed by the nodes internal to Ω_2, and finally by those on Ω_Γ. Therefore,

$$\begin{pmatrix} A_{II}^{(1)} & 0 & A_{IB}^{(1)} \\ 0 & A_{II}^{(2)} & A_{IB}^{(2)} \\ A_{BI}^{(1)} & A_{BI}^{(2)} & A_{BB}^{(1)} + A_{BB}^{(2)} \end{pmatrix} \begin{pmatrix} \mathbf{u}_I^{(1)} \\ \mathbf{u}_I^{(2)} \\ \mathbf{u}_B \end{pmatrix} = \begin{pmatrix} \mathbf{f}_I^{(1)} \\ \mathbf{f}_I^{(2)} \\ \mathbf{f}_B \end{pmatrix}. \tag{2.1}$$

If A is symmetric, then $A_{BI}^{(i)} = (A_{IB}^{(i)})^T$. $A_{BB}^{(i)}$ represents the mutual influence of the border nodes of Ω_i. For example, in the case of a finite element discretization, one has

$$[A_{BB}^{(i)}]_{kj} = a(\varphi_j|_{\Omega_i}, \varphi_k|_{\Omega_i}),$$

where φ_k, φ_j are the finite element shape functions associated to border nodes k and j, respectively, restricted to the subdomain Ω_i. The two null blocks arise because of the local support of the shape functions.

Eliminating the internal nodes from system (2.1) gives

$$S_{EO}\mathbf{u}_B = \mathbf{g}, \tag{2.2}$$

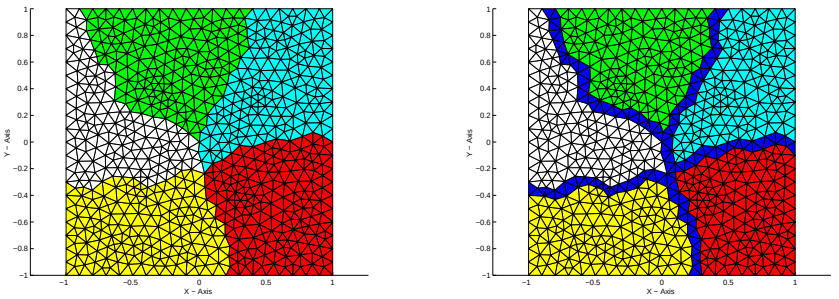


FIG. 2.1. *On the left, EO decomposition: Each element is assigned to a different subdomain* $\Omega_i, i = 1, \dots, M$. *On the right, VO decomposition: Each vertex is assigned to a different subdomain. The blue domain is called* Ω_Γ.

where $S_{EO} = S_1 + S_2$ and $\mathbf{g} = \mathbf{g}^{(1)} + \mathbf{g}^{(2)}$, with

$$S_i = A_{BB}^{(i)} - A_{BI}^{(i)} {A_{II}^{(i)}}^{-1} A_{IB}^{(i)}, \tag{2.3}$$

$$\mathbf{g}^{(i)} = \mathbf{f}_B^{(i)} - A_{BI}^{(i)} {A_{II}^{(i)}}^{-1} \mathbf{f}_I^{(i)}. \tag{2.4}$$

The generalization to an arbitrary number of subdomains (see the left-hand side of Figure 2.1) is straightforward. We introduce the restriction matrix $\bar{R}_i \in \mathbb{R}^{n_B^{(i)} \times n_B}$ which from a global vector $\mathbf{u}_B \in \mathbb{R}^{n_B}$ extracts the entries corresponding to the border nodes of Ω_i, i.e., $\mathbf{u}_B^{(i)} = \bar{R}_i \mathbf{u}_B$. Its transpose $\bar{R}_i^T$ is the prolongation matrix relative to the subdomain Ω_i which extends by zero a vector of border values. Then,

$$S_{EO} = \left(\sum_{i=1}^{M} \bar{R}_i^T S_i \bar{R}_i \right), \quad \mathbf{g} = \mathbf{f}_B - \sum_{i=1}^{M} \bar{R}_i^T A_{BI}^{(i)} {A_{II}^{(i)}}^{-1} \mathbf{f}_I^{(i)}.$$

2.2. Vertex-oriented decompositions. The strategy of section 2.1 can be followed for VO decompositions. In this case, (1.1) can be rewritten as

$$A\mathbf{u} = \begin{pmatrix} A_{II}^{(1)} & A_{IB}^{(1)} & 0 & 0 \\ A_{BI}^{(1)} & A_{BB}^{(1)} + A_\Gamma^{(1,1)} & 0 & A_\Gamma^{(1,2)} \\ 0 & 0 & A_{II}^{(2)} & A_{IB}^{(2)} \\ 0 & A_\Gamma^{(2,1)} & A_{BI}^{(2)} & A_{BB}^{(2)} + A_\Gamma^{(2,2)} \end{pmatrix} \begin{pmatrix} \mathbf{u}_I^{(1)} \\ \mathbf{u}_B^{(1)} \\ \mathbf{u}_I^{(2)} \\ \mathbf{u}_B^{(2)} \end{pmatrix} = \begin{pmatrix} \mathbf{f}_I^{(1)} \\ \mathbf{f}_B^{(1)} \\ \mathbf{f}_I^{(2)} \\ \mathbf{f}_B^{(2)} \end{pmatrix}. \tag{2.5}$$

The border nodes are split into two subsets, one belonging to $\overline{\Omega_1}$ and the other to $\overline{\Omega_2}$. A_Γ represents the mutual influence of border nodes, considering only the contribution from the subdomain Ω_Γ. For instance, in the case of a finite element discretization, one has

$$[A_\Gamma^{(l,m)}]_{ij} = a(\varphi_j|_{\Omega_\Gamma}, \varphi_i|_{\Omega_\Gamma})$$

for $i \in \overline{\Omega_l} \cap \overline{\Omega_\Gamma}$ and $j \in \overline{\Omega_m} \cap \overline{\Omega_\Gamma}$.

The elimination of the vector associated to the internal nodes leads to the system

$$S_{VO}\mathbf{u}_B = \begin{pmatrix} S_1 + A_\Gamma^{(1,1)} & A_\Gamma^{(1,2)} \\ A_\Gamma^{(2,1)} & S_2 + A_\Gamma^{(2,2)} \end{pmatrix} \begin{pmatrix} \mathbf{u}_B^{(1)} \\ \mathbf{u}_B^{(2)} \end{pmatrix} = \begin{pmatrix} \mathbf{g}^{(1)} \\ \mathbf{g}^{(2)} \end{pmatrix}, \tag{2.6}$$

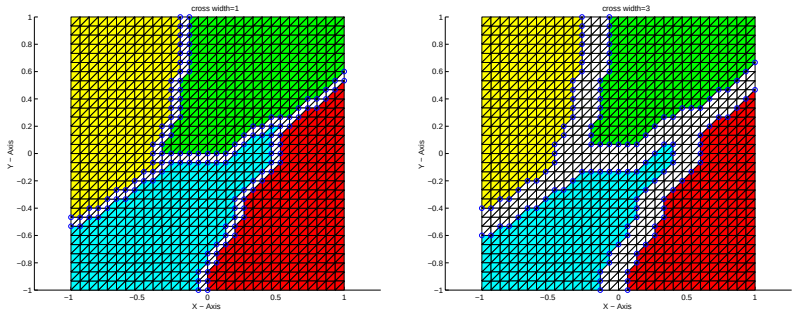


FIG. 2.2. *Two VO unstructured decompositions, with $M = 4$ and cross width $\delta = 1$ (left) and $\delta = 3$ (right).*

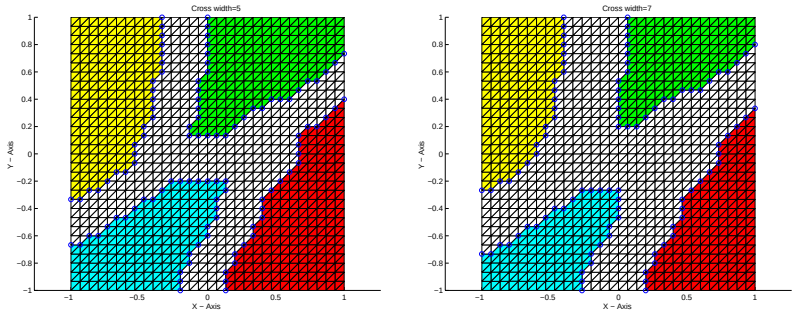


FIG. 2.3. *Two VO unstructured decompositions, with $M = 4$ and cross width $\delta = 5$ (left) and $\delta = 7$ (right).*

with S_i and $\mathbf{g}^{(i)}$ defined by (2.3) and (2.4). Its generalization to the case of M subdomains gives

(2.7)
$$\begin{pmatrix} S_1 + A_\Gamma^{(1,1)} & A_\Gamma^{(1,2)} & \dots & A_\Gamma^{(1,M)} \\ A_\Gamma^{(2,1)} & S_2 + A_\Gamma^{(2,2)} & \dots & A_\Gamma^{(2,M)} \\ \vdots & \vdots & \ddots & \vdots \\ A_\Gamma^{(M,1)} & A_\Gamma^{(M,2)} & \dots & S_M + A_\Gamma^{(M,M)} \end{pmatrix} \begin{pmatrix} \mathbf{u}_B^{(1)} \\ \mathbf{u}_B^{(2)} \\ \vdots \\ \mathbf{u}_B^{(M)} \end{pmatrix} = \begin{pmatrix} \mathbf{g}^{(1)} \\ \mathbf{g}^{(2)} \\ \vdots \\ \mathbf{g}^{(M)} \end{pmatrix}.$$

Some of the blocks $A_\Gamma^{(i,j)}$ may in fact be empty. This happens when no element in the triangulation contains nodes belonging to both $\overline{\Omega_i}$ and $\overline{\Omega_j}$.

The subdomain Ω_Γ can be characterized by its , that is, the minimum number of elements that one has to cross moving from a subdomain Ω_i to an $\Omega_j, i \neq j$. Formally, this width can be defined as follows. Let us indicate with Δ_{ij} the minimum number of edges that one has to cross when moving from an arbitrary node belonging to Ω_i to a node of Ω_j, $i, j = 1, \dots, M, i \neq j$; then

$$\delta = \min_{i,j} \Delta_{ij}$$

represents the width of Ω_Γ. In the VO decomposition described so far, we have $\delta = 1$, while an EO decomposition has $\delta = 0$. Examples of VO decomposition for $M = 4$ and $\delta = 1, 3, 5, 7$ are given in Figures 2.2 and 2.3. The algorithm reported in Figure 2.4 can be used to increment the width of a given decomposition.

```
1  For k = 1, ..., N Do
2      For every element j of T_{h,i}, i = 1, ..., M Do:
3          If k is a border node, assign the element j to T_{h,Γ}
4      End For
5      Update the set of internal and border nodes
6  End For
```

FIG. 2.4. *Algorithmic procedure to increment δ for a given VO decomposition.*

3. Preconditioners for vertex-oriented decompositions. We now focus on the definition of overlapping and nonoverlapping Schwarz preconditioners for VO decompositions. First, in section 3.1 we revisit aggregation techniques and illustrate how to take advantage of VO decompositions to define a two-level Schwarz preconditioner for the original system (1.1). Then, in section 3.2 we introduce a novel preconditioner, based on the application of Dirichlet–Neumann algorithms to the VO decomposition, and present a bound on the condition number of the resulting system.

3.1. Preconditioners for the original system. Let $A_i = R_i A R_i^T$, $i = 1, \ldots, M$, where R_i is the rectangular (restriction) matrix which, when applied to a vector corresponding to $\overline{\Omega}$, returns its components in $\overline{\Omega_i}$. The matrix A_i can also be defined as

$$A_i = \begin{pmatrix} A_{II}^{(i)} & A_{IB}^{(i)} \\ A_{BI}^{(i)} & A_{BB}^{(i)} + A_{\Gamma}^{(i,i)} \end{pmatrix}.$$

Given the A_i's, a two-level Schwarz preconditioner $P_{S,C,add}$ can be expressed as

$$P_{S,C,add}^{-1} = \sum_{i=0}^{M} R_i^T A_i^{-1} R_i. \tag{3.1}$$

A_0 is the so-called [illegible], whose role is to spread out information among the distant subdomains. The construction of A_0 involves the discretization of the PDE problem on a coarser grid, whose number of elements n_0 is much lower than that of the original fine grid.

This procedure may be difficult or computationally expensive on general unstructured grids and arbitrary geometries. In addition, for a general, possibly nonconvex domain, it is difficult to ensure that the boundary conditions are correctly represented on the coarse level.

In VO decompositions, however, the definition of a coarse operator can be based on aggregation procedures. Precisely, on each subdomain Ω_i we build $n_{0,i}$ [illegible] $\vartheta_{i,s}$, $i = 1, \ldots, M$, $s = 1, \ldots, n_{0,i}$, each of them representing a set of contiguous vertices; see Figure 3.1 for an example in two dimensions. Then on every Ω_i, each aggregate is associated to a vector $\boldsymbol{\eta}_{s,i}$, $s = 1, \ldots, n_{0,i}$, whose elements are built following the rule

$$\boldsymbol{\eta}_{s,i}(k) = \begin{cases} 1 & \text{if } k \text{ belongs to } \vartheta_{s,i}, \\ 0 & \text{otherwise.} \end{cases}$$

At this stage, the construction of R_0, called the smoothed aggregation (SA) procedure, is made in two steps: coarsening and grid transfer construction. The former corresponds to building a nonsmoothed restriction operator $\tilde{R}_0$ so that each fine

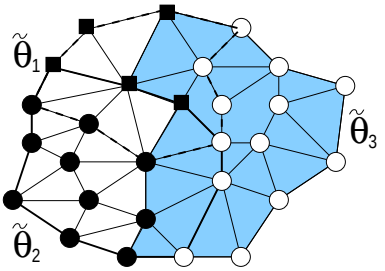


FIG. 3.1. *Examples of aggregates for a 2D domain. The nodes marked with ■, ○, and • belong to $\tilde{\vartheta}_1$, $\tilde{\vartheta}_2$, and $\tilde{\vartheta}_3$, respectively.*

grid node is included in just one aggregate. This decomposition can be obtained using a graph partitioning algorithm (VO decomposition of the grid). The value $n_0 = \sum_{i=1}^{M} n_{0,i}$ represents the dimension of the coarse space, since each aggregate will be given a coarse grid basis function. We will indicate with $\tilde{\vartheta}_i$ the set of nodes that form the nonsmoothed aggregate i. The entries of $\tilde{R}_0$ are

$$\tilde{R}_0(i,j) = \begin{cases} 1 & \text{if } j \in \tilde{\vartheta}_i, \\ 0 & \text{otherwise.} \end{cases} \tag{3.2}$$

$\tilde{R}_0$ can be viewed as a simple grid transfer operator corresponding to piecewise constant interpolation. A 2D example is reported in Figure 3.1. The second step (which may actually be avoided) consists of applying a prolongation smoother $\tilde{S}_0$ to produce the final prolongation operator $R_0^T = \tilde{R}_0^T \tilde{S}_0$. For more details, the reader is referred to [29, 3].

3.2. Preconditioners for the SC system. Equation (2.7) may be reinterpreted as a two-subdomain formulation, the subdomains being given by Ω_Γ and $\Omega_\star = \cup_{i=1}^{M} \Omega_i$. Clearly one has $\overline{\Omega} = \overline{\Omega_\star} \cup \overline{\Omega_\Gamma}$, and

$$S_{VO} = S_\star + A_\Gamma, \tag{3.3}$$

with $S_\star = \text{blockdiag}\{S_i, i = 1, \ldots, M\}$. In (3.3), $S_\star$ is the SC obtained by eliminating the internal nodes of $\Omega_\star$, while the SC of domain Ω_Γ, usually indicated by S_Γ, coincides with A_Γ, since no internal nodes have been included in Ω_Γ.

Note that $S_\star$ has a (dense) diagonal block form, while A_Γ is sparse and, in general, features many zero-blocks.

For the matrix (3.3) we propose three preconditioners:

$$P_\star = S_\star, \quad P_\Gamma = A_\Gamma, \quad P_{NN} = (S_\star^{-1} + A_\Gamma^{-1})^{-1}. \tag{3.4}$$

$P_\star$ and P_Γ are formally equivalent to Dirichlet–Neumann (DN) preconditioners [22, section 3.2]. The inversion of $P_\star$ involves M Neumann solves (one in every Ω_i), whereas only one Neumann solve in Ω_Γ is required to invert P_Γ. The latter solve is somehow unusual since all the nodes of Ω_Γ stand on the border of the domain Ω_Γ. P_{NN} is the two-subdomain Neumann–Neumann (NN) preconditioner.

1. The DN and NN preconditioners are optimal for the two-subdomain case. However, the analysis requires the subdomains to have a measure independent of h. This condition is not verified when we use as preconditioner $P_\star$, P_Γ, or P_{NN}, since

the width of Ω_Γ is of order h. Clearly, if the subdivision is modified so that Ω_Γ has a measure independent of h, then $P_\star$, P_Γ, and P_{NN} become optimal preconditioners.

The preconditioner P_Γ behaves better (both theoretically and numerically) than $P_\star$ (see the theoretical results in Proposition 1, which we have been able to prove only for the P_Γ preconditioner; see, moreover, the numerical results in Table 6). On the other hand, though the condition number of $P_{NN}^{-1}S_{VO}$ is in general slightly lower than that of $P_\Gamma^{-1}S_{VO}$ (see Table 6), the computational complexity of P_{NN} is higher than that of P_Γ, as it requires $M+1$ solves. Moreover, in the "floating" subdomains Ω_i (those internal to Ω), the solution of a pure Neumann problem can demand compatibility conditions on the data. For this reason, in what follows we will focus almost exclusively on the preconditioner P_Γ.

We now derive a convergence estimate for the interface-strip preconditioner. For the sake of simplicity, the estimate of the condition number of the preconditioned Schur complement $P_\Gamma^{-1}S$ will be presented only in a case of simple geometry for the Laplace operator with homogeneous boundary conditions. However, using similar arguments, the interested reader could easily extend the result to more general situations: For instance, the result is true for any second-order elliptic operator (without zero-order terms and being associated to a symmetric and coercive bilinear form), and also for the two-dimensional elasticity system; however, other boundary conditions and domains are allowed as well. In particular, in Remark 2 we consider the important case of a diffusion coefficient with jumps, showing that the condition number of the preconditioned Schur complement is independent of the jumps.

Let us consider the square $\Omega = (0,1)\times(0,1)$ and a family of uniform triangulations $\mathcal{T}_h$ of it. We assume that Ω is subdivided into the union of the disjoint squares Ω_j, $j = 1,\dots,M$, with side-length H such that $\sqrt{M}(H+h) = 1+h$, and of the domain Ω_Γ, which is connected and consists of the union of $2(\sqrt{M}-1)$ nondisjoint strips of length 1 and width h. Let us also set

$$\Omega_\star = \cup_{j=1}^{M}\Omega_j, \quad \Omega_\Gamma = \Omega \setminus \overline{\Omega_\star}, \quad \Gamma = \overline{\Omega_\star}\cap\overline{\Omega_\Gamma}. \tag{3.5}$$

We denote by η_h the trace on Γ of the piecewise [illegible] functions v_h defined in Ω. Then, we indicate by $E_{h,\star}\eta_h$ the discrete harmonic extension of η_h in $\Omega_\star$ and by $E_{h,\Gamma}\eta_h$ the discrete harmonic extension of η_h in Ω_Γ. Since in Ω_Γ there are no internal nodes (the width of Ω_Γ is equal to h), the function $E_{h,\Gamma}\eta_h$ is simply the interpolant of η_h in $\overline{\Omega_\Gamma}$.

The action of both the Schur complement S and P_Γ can be expressed through the local bilinear forms in the following way (here $\boldsymbol{\eta}$ and $\boldsymbol{\mu}$ denote the vector of the nodal values of η_h and μ_h, respectively) (see, e.g., [22, p. 22]):

$$S\boldsymbol{\eta}\cdot\boldsymbol{\mu} = \int_{\Omega_\star}\nabla E_{h,\star}\eta_h\cdot\nabla E_{h,\star}\mu_h + \int_{\Omega_\Gamma}\nabla E_{h,\Gamma}\eta_h\cdot\nabla E_{h,\Gamma}\mu_h,$$

$$P_\Gamma\boldsymbol{\eta}\cdot\boldsymbol{\mu} = \int_{\Omega_\Gamma}\nabla E_{h,\Gamma}\eta_h\cdot\nabla E_{h,\Gamma}\mu_h.$$

We can prove the following proposition.

PROPOSITION 1. [illegible] C_1 [illegible] h [illegible] H [illegible]

$$P_\Gamma\boldsymbol{\eta}\cdot\boldsymbol{\eta} \le S\boldsymbol{\eta}\cdot\boldsymbol{\eta} \le C_1\,\frac{H}{h}\,P_\Gamma\boldsymbol{\eta}\cdot\boldsymbol{\eta} \quad \forall\boldsymbol{\eta}\in\mathbb{R}^{n_B}.$$

[illegible] $P_\Gamma^{-1}S$ [illegible] $\kappa(P_\Gamma^{-1}S) \le C_1 H/h$

$\int_{\Omega_\star}|\nabla E_{h,\star}\eta_h|^2$, and this can be done by adding the contributions to the integral from each square Ω_j. The first inequality is obvious. To prove the second one, we need to bound

By means of a translation, instead of Ω_j we can consider the square Q_H centered in $(0,0)$ and with sides of length H. Then by the change of variable $(\widehat{x},\widehat{y}) = T(x,y) = (x/H, y/H)$, we map Q_H over the unit square $\widehat{Q}$. The triangulation over Q_H is transformed into a triangulation onto $\widehat{Q}$; the correspondent mesh size is given by h/H. Introducing, for each function g defined in Q_H, the function $\widehat{g} = g \circ T^{-1}$ defined in $\widehat{Q}$, and denoting by $\widehat{E}_{h/H}\widehat{\eta_h}$ the discrete harmonic extension of $\widehat{\eta_h}$ with respect to the triangulation in $\widehat{Q}$, it is easily verified that $\widehat{E}_{h/H}\widehat{\eta_h} = \widehat{E_{h,\star}\eta_h}$ and that

$$\int_{Q_H}|\nabla E_{h,\star}\eta_h|^2 = \int_{\widehat{Q}}|\widehat{\nabla}\widehat{E}_{h/H}\widehat{\eta_h}|^2. \tag{3.6}$$

The uniform extension theorem for finite elements (see, e.g., [30, sect. 4.2.2]) gives that

$$\int_{\widehat{Q}}|\widehat{\nabla}\widehat{E}_{h/H}\widehat{\eta_h}|^2 \le C_0|\widehat{\eta}_h|^2_{1/2,\partial\widehat{Q}}\ , \tag{3.7}$$

where the constant C_0 does not depend on h and H and $|\cdot|_{1/2,\partial\widehat{Q}}$ denotes the seminorm of $H^{1/2}(\partial\widehat{Q})$.

Let us now observe that the set Ω_Γ is the union of $F_j \cap \Omega$, where the "frames" F_j are the sets of width h around Ω_j.

Note that these sets are not disjoint, but any possible intersection concerns at most four of them. Moreover, by means of a translation, we can think that each set F_j is the frame $F_{H,h}$ of the square Q_H.

Let us consider now the set $\widehat{F}_{1,1/2}$, that is, the frame of width $1/2$ around $\widehat{Q}$. We want to construct a function in $\widehat{F}_{1,1/2}$ such that its value on the internal boundary of $\widehat{F}_{1,1/2}$ (that is, on $\partial\widehat{Q}$) is equal to $\widehat{\eta}_h$. The natural idea is to start from $E_{h,\Gamma}\eta_h$, that is, from the interpolant of η_h in $\overline{\Omega_\Gamma}$. For simplicity, we will write $\psi_h = E_{h,\Gamma}\eta_h$, and we recall that $\psi_{h|\Gamma} = \eta_h$.

Now we have to pass from the domain F_j (or, equivalently, from $F_{H,h}$) onto $\widehat{F}_{1,1/2}$. First of all, note that the transformation T maps $F_{H,h}$ onto the set $\widehat{F}_{1,h/H}$. Moreover, we define

$$t(\xi) := \begin{cases} \xi & \text{if } |\xi| \le 1/2, \\ 1/2[1 + h^{-1}H(|\xi| - 1/2)]\xi/|\xi| & \text{if } 1/2 \le |\xi| \le 1/2 + h/H; \end{cases}$$

then we introduce the transformation

$$K(\widehat{x},\widehat{y}) := (t(\widehat{x}), t(\widehat{y})),$$

which maps $\widehat{F}_{1,h/H}$ onto $\widehat{F}_{1,1/2}$, leaving the internal boundary $\partial\widehat{Q}$ unchanged. We are now in a position to set

$$\psi^\sharp : \widehat{F}_{1,1/2} \to \mathbb{R}, \quad \psi^\sharp := \psi_{h|F_{H,h}} \circ T^{-1} \circ K^{-1},$$

and we easily verify that $\psi^\sharp_{|\partial\widehat{Q}} = \widehat{\eta}_h$.

We can thus use the trace theorem to obtain

$$|\widehat{\eta}_h|^2_{1/2,\partial\widehat{Q}} \leq C_1 ||\nabla\psi^\sharp||^2_{0,\widehat{F}_{1,1/2}} \tag{3.8}$$

(see [30, Theorem 4.1]). To estimate the right-hand side, taking into consideration the explicit expression of the transformations T and K, we can prove

$$||\nabla\psi^\sharp||^2_{0,\widehat{F}_{1,1/2}} \leq C_2 h^{-1} H \int_{F_{H,h}} |\nabla\psi_h|^2 \ . \tag{3.9}$$

By collecting (3.6), (3.7), (3.8), and (3.9) we have thus obtained

$$\begin{aligned} \textstyle\int_{\Omega_j} |\nabla E_{h,\star}\eta_h|^2 &\leq C_3 h^{-1} H \textstyle\int_{F_j} |\nabla E_{h,\Gamma}\eta_h|^2 \\ &= C_3 h^{-1} H \textstyle\int_{F_j\cap\Omega} |\nabla E_{h,\Gamma}\eta_h|^2. \end{aligned} \tag{3.10}$$

The last equality holds true as, for those squares Ω_j having at least one side on $\partial\Omega$, the function η_h has been extended by 0 on $\partial\Omega_j \cap \partial\Omega$ and on the boundary of F_j external to Ω, so that $E_{h,\Gamma}\eta_h = 0$ on $F_j \setminus (F_j \cap \Omega)$.

Bearing in mind the geometry of Ω_Γ we easily obtain the estimate

$$4 \int_{\Omega_\Gamma} |\nabla E_{h,\Gamma}\eta_h|^2 \geq \sum_j \int_{F_j\cap\Omega} |\nabla E_{h,\Gamma}\eta_h|^2.$$

The thesis now follows by summing up the contribution from all the domains Ω_j and the frames F_j. □

. . . . 2. The same result also holds for the elliptic operator

$$Lu := -\sum_i D_i[\rho(\mathbf{x}) D_i u],$$

where $\rho(x)$ is a positive and bounded function in Ω.

A more precise conclusion can be reached for a piecewise constant elliptic coefficient, that is, when $\rho(\mathbf{x}) = \rho_j > 0$ in $\Omega_j \cup (F_j^{1/2} \cap \Omega)$, $j = 1, \ldots, M$, where $F_j^{1/2}$ is the frame of width $h/2$ around Ω_j: In this case the constant C_1 in Proposition 1 does not depend on the jumps of the coefficients ρ_j.

In fact, in each domain Ω_j the extension operator with respect to the operator L coincides with the harmonic extension, therefore (3.7) holds. Moreover, proceeding as before we obtain (3.10) (with $F_j^{1/2}$ instead of F_j) and therefore

$$\begin{aligned} S_\star \boldsymbol{\eta}\cdot\boldsymbol{\eta} &= \sum_{j=1}^M \rho_j \int_{\Omega_j} |\nabla E_{h,\star}\eta_h|^2 \\ &\leq C_3 h^{-1} H \sum_{j=1}^M \rho_j \int_{F_j^{1/2}\cap\Omega} |\nabla E_{h,\Gamma}\eta_h|^2 \\ &= C_3 h^{-1} H P_\Gamma \boldsymbol{\eta}\cdot\boldsymbol{\eta}, \end{aligned}$$

where the constant C_3 does not depend on the jumps of the coefficients ρ_j.

TABLE 1
2D model problem. Condition number of $P_S^{-1}A$ using the one-level Schwarz preconditioner P_S with minimal overlap ($\delta = h$).

$\kappa(P_S^{-1}A)$	$H = 1/4$	$H = 1/8$	$H = 1/16$	$H = 1/32$
$h = 1/15$	24.29	45.92	-	-
$h = 1/31$	51.66	98.63	194.97	-
$h = 1/63$	106.31	203.80	403.31	804.48
$h = 1/127$	215.58	414.03	819.73	1635.34

4. Numerical experiments. We consider a piecewise linear finite element approximation of the problem

$$\begin{cases} -\sum_i D_i[\rho(x) D_i u] = f & \text{in } \Omega, \\ u = g & \text{on } \partial\Omega, \end{cases} \tag{4.1}$$

to numerically validate the preconditioners proposed in section 3 and compare them with other popular DD preconditioners.

For 2D problems, $\Omega = (0,1)\times(0,1)$ and the mesh is built by dividing Ω into $(1/h)^2$ uniform squares, which are then subdivided into two triangles. For 3D problems, $\Omega = (0,1) \times (0,1) \times (0,1)$ and the mesh is divided into $(1/h)^3$ uniform hexahedra, subdivided into 5 tetrahedra. Unless otherwise stated, $\rho(x,y) = 1$ and $f(x,y)$ and $g(x,y)$ are chosen so that $u(x,y) = \sin x \sin y$ in 2D and $u(x,y,z) = \sin x \sin y \sin z$ in 3D.

As regards the decomposition into subdomains, we consider overlapping squares (for 2D problems) or hexahedra (for 3D problems) Ω_i, of equal area H^2 or volume H^3, plus the subdomain Ω_Γ in the case of a VO decomposition. Unless otherwise specified, the width of Ω_Γ is of one element, and the linear systems (1.1), (2.2), and (2.7) are solved by a preconditioned conjugate gradient (PCG) method [12, 23]. The estimated condition number (ECN) reported in some of the tables is computed according to the algorithm given in [11, pp. 128–130].

4.1. Solution of the original system. We start by considering the solution of the system (1.1), using a one-level Schwarz preconditioner on a VO decomposition of the grid. In this case, Ω_Γ represents the union of the overlapping regions among the subdomains Ω_i. In Table 1 we report the ECN of the preconditioned system. Table 2 reports the ECN for two-level Schwarz preconditioners, using, respectively, a coarse grid and the aggregation procedure with nonsmoothed aggregation. The latter approach yields a preconditioned matrix that, for any given H and h, has a condition number asymptotically equivalent but larger than that obtained using a classical coarse grid. Nonetheless, this drawback is somehow compensated for by the ease of generating nonsmoothed aggregation. For further results on this subject, see [17, 25, 26].

4.2. Interface-strip preconditioners. We now focus on the solution of system (2.7). For regular decompositions such as those presented in Proposition 1, the Schur matrix S_{VO} is (as expected) badly conditioned: From Table 3 one sees that $\kappa(S_{VO})$ is slightly more than linear with h^{-1}; therefore preconditioning this matrix is mandatory. The preconditioner P_Γ proposed in section 3.2 is also studied in Table 3. The results of Table 3 confirm the theoretical estimates of Proposition 1, while Table 4 suggests that a similar estimate may hold for 3D domains as well.

TABLE 2
2D model problem. Condition number of $P_{S,C,add}^{-1}A$, with a standard coarse space (in parentheses: with a coarse space built using nonsmoothed aggregation).

$\kappa(P_{S,C,add}^{-1}A)$	$H = 1/4$	$H = 1/8$	$H = 1/16$	$H = 1/32$
$h = 1/15$	4.17 (12.09)	3.86 (8.53)	-	-
$h = 1/31$	6.77 (25.47)	4.79 (17.23)	4.20 (9.76)	-
$h = 1/63$	12.22 (52.86)	7.19 (34.64)	5.13 (19.06)	4.36 (10.11)
$h = 1/127$	23.08 (107.91)	12.84 (69.61)	7.37 (39.95)	5.35 (20.33)

TABLE 3
2D model problem. Condition number of S_{VO} (in parentheses: that of $P_\Gamma^{-1}S_{VO}$).

	$H = 1/2$	$H = 1/4$	$H = 1/8$	$H = 1/16$
$h = 1/15$	41.33 (6.72)	62.75 (4.87)	90.52 (2.00)	- (-)
$h = 1/31$	96.85 (13.54)	155.75 (9.76)	264.73 (5.27)	- (2.00)
$h = 1/63$	209.27 (27.71)	347.06 (19.76)	627.94 (10.83)	- (5.44)
$h = 1/127$	431.87 (54.50)	732.06 (39.48)	1366.15 (21.93)	- (11.09)

TABLE 4
3D model problem. Condition number of S_{VO} (in parentheses: that of $P_\Gamma^{-1}S_{VO}$).

	$H = 1/2$	$H = 1/3$	$H = 1/4$
$h = 1/15$	74.45 (5.19)	95.54 (4.11)	- (3.23)
$h = 1/23$	- (7.44)	- (5.54)	- (4.49)
$h = 1/31$	390.13 (9.74)	507.33 (6.87)	596.79 (5.82)
$h = 1/47$	- (14.36)	- (10.61)	- (8.54)
$h = 1/63$	1785.84 (19.01)	2391.22 (13.76)	2892.44 (11.28)

TABLE 5
2D model problem. Condition number of $P_\Gamma^{-1}S_{VO}$ for problem (4.1). The computational domain is divided into squares (of length H). The coefficient ρ has a checkerboard distribution, with $\rho = 1$ in the white squares and $\rho = \rho_1$ in the black squares. Ω_Γ has width $\delta = 2$.

H	h	$\rho_1 = 1$	$\rho_1 = 10^2$	$\rho_1 = 10^4$	$\rho_1 = 10^6$
1/2	1/16	3.38	3.70	3.72	3.72
	1/32	6.35	7.14	7.16	7.16
	1/64	12.33	14.09	14.17	14.17
1/4	1/16	2.84	3.09	3.09	3.09
	1/32	4.82	5.50	5.52	5.52
	1/64	9.69	10.94	10.99	10.99
1/8	1/32	2.90	3.08	3.09	3.09
	1/64	5.09	5.76	5.79	5.79
1/16	1/64	2.91	3.08	3.09	3.09

Table 5 is related to a model problem with discontinuous coefficients. The computational domain $\Omega = (0,1)\times(0,1)$ is subdivided into nonoverlapping squares of side H. Using this checkerboard distribution, the coefficient ρ of problem (4.1) is set to 1 in all the white squares and to ρ_1 in the black squares. Ω_Γ is constructed as follows: Let us indicate the interface between the squares with Γ. Then, Ω_Γ contains all the elements that share at least one edge with Γ. Note that, by construction, $\delta = 2$. The results reported in the table confirm that the condition number of $P_\Gamma^{-1}S_{VO}$ does not depend on the jumps in the coefficient ρ of (4.1); see Remark 2 for more details.

The results for the other preconditioners proposed in section 3.2 are reported in Table 6. For a given h, we have first constructed a VO decomposition in 4 subdomains,

TABLE 6
2D model problem. Condition number for the VO SC matrix, with no preconditioning and using the preconditioners proposed in section 3. The domain is partitioned into 4 subdomains, and the influence of the cross width is reported. (Note that $S_\Gamma = A_\Gamma$ if $\delta = 1$ but not if $\delta \geq 2$.)

	δ	$\kappa(S_{VO})$	$\kappa(S_\Gamma^{-1}S_{VO})$	$\kappa(S_\star^{-1}S_{VO})$	$\kappa(P_{NN}^{-1}S_{VO})$
$h = 1/15$	1	41.33	6.72	16.01	4.92
	3	20.47	2.77	4.67	1.98
	5	12.95	2.08	2.97	1.15
$h = 1/31$	1	96.85	13.54	31.95	8.86
	3	61.59	5.00	9.66	3.22
	5	48.99	3.36	5.66	2.19
$h = 1/63$	1	209.27	27.27	63.86	16.81
	3	149.69	9.66	20.01	5.84
	5	133.05	6.05	11.69	3.72
$h = 1/127$	1	431.87	54.51	127.79	32.78
	3	329.70	18.71	41.28	11.15
	5	309.76	11.52	24.29	6.88

TABLE 7
2D model problem. Condition number of S_{VO} (in parentheses: that of $P_\Gamma^{-1}S_{VO}$). Unstructured decomposition into M subdomains (obtained using METIS).

	$M = 12$	$M = 20$	$M = 32$
$h = 1/15$	80.54 (11.67)	85.62 (15.69)	90.42 (7.65)
$h = 1/31$	150.22 (47.16)	184.14 (35.76)	219.02 (17.93)
$h = 1/47$	235.24 (20.60)	286.27 (40.57)	330.72 (24.84)
$h = 1/63$	300.47 (45.93)	382.03 (48.63)	464.09 (34.27)

as described in section 2 (that is, $\delta = 1$). Then, our algorithm (see Figure 2.4) was used to increase the width. In the table, $P_{NN}^{-1} = S_\star^{-1} + A_\Gamma^{-1}$. All the tested preconditioners share the same asymptotic behavior; however, $P_\star$ results in a consistently larger condition number than P_{NN} and P_Γ. As M additional Neumann problems have to be solved to invert P_{NN}, in the following we will focus only on P_Γ.

In Table 7 the ECN of S_{VO} and $P_\Gamma^{-1}S_{VO}$ is again reported, this time for 2D decompositions into M subdomains; the subdomain partition has been obtained by using the software METIS (see [14]). The analogous results for the 3D case are reported in Table 8. These examples again show the good performance of the interface-strip preconditioner, especially in the 3D case, where the ECN appears essentially independent of the mesh size and the number of subdomains and always remains quite small.

4.3. Comparisons between interface-strip and balancing Neumann–Neumann preconditioners. In order to compare P_Γ with more standard preconditioners, we have used one of the state-of-the-art preconditioners for nonoverlapping decompositions, namely, the balancing Neumann–Neumann (BNN). The BNN preconditioner is based on the NN preconditioner for EO decompositions

$$P_{NN}^{-1} = \sum_{i=1}^{M} \bar{R}_i D_i {S_i}^{-1} D_i \bar{R}_i. \tag{4.2}$$

The scaling diagonal matrices D_i allow us to preserve good conditioning even when the subdomains have different characteristics, for instance, because of the different values of the coefficients of the differential operator.

TABLE 8
3D model problem. Condition number of S_{VO} (in parentheses: that of $P_\Gamma^{-1}S_{VO}$). Unstructured decomposition into M subdomains (obtained using METIS).

	$M = 12$	$M = 20$	$M = 32$
$h = 1/15$	84.75 (20.50)	92.00 (22.32)	97.78 (20.54)
$h = 1/31$	458.46 (21.44)	500.93 (21.73)	569.71 (21.50)
$h = 1/47$	1177.10 (23.19)	1253.82 (26.23)	1422.79 (21.83)

TABLE 9
2D model problem. Condition number of $P_{BNN}^{-1}S_{EO}$.

$\kappa((P_{BNN})^{-1}S_{EO})$	$H = 1/2$	$H = 1/4$	$H = 1/8$	$H = 1/16$
$h = 1/15$	1.67	1.48	1.27	-
$h = 1/31$	2.17	2.03	1.47	1.29
$h = 1/63$	2.78	2.76	2.08	1.55
$h = 1/127$	3.51	3.67	2.81	2.07

TABLE 10
Comparison of P_{BNN} (EO decomposition) and P_Γ (VO decomposition). `it_BNN` *and* `it_`Γ *indicate the iterations needed to converge using P_{BNN} and P_Γ, respectively, while* `flops_BNN` *and* `flops_`Γ *indicate the MATLAB flops required by the solution of the problem using CG.*

h	M	$\kappa(S_{EO})$	`it_BNN`	`flops_BNN`	$\kappa(S_{VO})$	`it_`Γ	`flops_`Γ
1/20	4	34.1	9	4.55×10^5	80.5	30	4.09×10^5
	8	52.7	13	9.20×10^5	109.5	33	5.46×10^5
	16	91.3	18	1.80×10^6	145.8	36	7.99×10^5
	32	111.8	16	2.94×10^6	190.6	39	1.08×10^6
1/40	4	71.9	10	3.65×10^6	180.1	38	1.78×10^6
	8	128.2	16	5.90×10^6	254.1	48	2.99×10^6
	16	168.2	19	8.31×10^6	343.9	52	3.96×10^6
	32	230.1	20	1.33×10^7	462.3	53	4.91×10^6
	64	343.8	22	2.66×10^7	608.7	51	7.72×10^6
	128	463.8	21	5.72×10^7	770.4	44	1.29×10^7
1/60	4	125.1	12	1.65×10^7	268.4	45	4.69×10^6
	8	187.7	16	1.82×10^7	406.3	59	8.04×10^6
	16	255.2	20	2.31×10^7	522.0	57	8.97×10^6
	32	357.8	22	3.37×10^7	712.3	59	1.12×10^7
	64	539.4	22	5.61×10^7	971.4	59	1.53×10^7

It is possible to prove that the preconditioner (4.2) satisfies the estimate

$$\kappa\left(P_{NN}^{-1}S\right) \leq \frac{C}{H^2}\left(1 + \log(H/h)\right)^2.$$

Despite the term $1/H^2$, the convergence of the NN algorithm is quite good for a small number of subdomains, as noted in [5, 18]. The BNN preconditioner, proposed by Mandel [19], adds a coarse correction to (4.2), constructed by using piecewise constant coarse grid space. The corresponding coarse operator involves one unknown per subdomain in the scalar case. It can be proved that the resulting preconditioner satisfies

$$\kappa(P_{BNN}^{-1}A) \leq C\left(1 + \log\frac{H}{h}\right)^2. \tag{4.3}$$

For numerical results, we refer the reader to [19, 7] or [22, sect. 3.2.2]; see also Table 9.

A comparison between BNN (with an EO decomposition) and P_Γ (with a VO decomposition) is shown in Table 10. The 2D grid has been partitioned, by means of

the software METIS, into M subdomains using a VO decomposition. Then, in order to obtain an EO decomposition, each of the elements in Ω_Γ has been assigned to one of the neighboring subdomains. The SC system has been solved with a tolerance of 10^{-10}.

In Table 10, `it_`Γ and `it_BNN` indicate the number of iterations needed to converge using P_Γ and P_{BNN}, respectively. `flops_`Γ and `flops_BNN` refer to the flops required to solve the problem, as indicated by the MATLAB command `flops`. We have not included in the flop count the operations required to explicitly form the SC matrix and the coarse matrix of BNN. The Neumann problems have been solved using an LU decomposition. The operations needed to compute the LU factors have not been included in the flop count.

Since the interface-strip preconditioner is suboptimal, in terms of iterations it is always outperformed by BNN, whose asymptotic behavior (4.3) is more favorable than that presented in Proposition 1, especially for large values of H/h. However, each application of BNN requires the solution of M Neumann problems; hence, when M is large (i.e., H is small), the BNN method is much more expensive than one application of P_Γ, which requires only the solution of a very sparse Neumann problem on Ω_Γ. In fact, the results presented in this section show that the proposed interface-strip preconditioner can be the right choice for the presented values of h and H, and therefore it should be preferred to BNN in these cases.

4.4. Parallel computations. We now report some preliminary numerical results for assessing the scalability of the interface-strip preconditioner. In a parallel environment, the strip Ω_Γ is replicated among the processors and therefore the preconditioning step

$$P_\Gamma \mathbf{z} = \mathbf{r} \tag{4.4}$$

must be solved in a distributed way. One can either adopt a parallel direct solver or solve the Neumann problem on Ω_Γ iteratively, using a nested iterative solver. Some numerical results (not reported here) suggested that we focus on the iterative solver. We have considered two variants. In the first one, indicated in the tables by $P_{\Gamma,J}$, two steps of the Jacobi method are applied to (4.4) with initial guess $\mathbf{z}_0 = \mathbf{0}$. The second one, indicated by $P_{\Gamma,CG}$, uses a nested conjugate gradient, iterated up to the fulfillment of the criterion

$$\frac{\|\mathbf{r} - P_\Gamma \mathbf{z}_i\|_2}{\|\mathbf{r}\|_2} \leq 10^{-2}.$$

Note that this latter approach may result in a nonconstant preconditioner, leading PCG to diverge (unless the linear system with P_Γ is solved up to machine precision). For this reason, in the numerical experiments using a nested Krylov solver, we have resorted to GMRESR [28], which allows variable preconditioning.

Tables 11 and 12 report the iterations required to solve up to a tolerance of $\|\mathbf{r}_k\|/\|\mathbf{r}_0\| \leq 10^{-5}$ our model problem on $(0,1) \times (0,1)$ and on $(0,1) \times (0,1) \times (0,1)$, respectively, while Table 13 reports the CPU times, obtained on a LINUX cluster with 12 nodes, each one composed by a biprocessor Pentium IV. Although simple to implement, $P_{\Gamma,J}$ has some effects only for 2D problems, while in 3D the reduction is only marginal. For example, in 2D with $h = 1/512$ and $H = 1/4$, the iterations are reduced from 428 in the nonpreconditioned system to 186 (see Table 11), while in 3D with $h = 1/64$ and $H = 1/4$ the iterations are reduced from 70 to 40 (see Table 12).

TABLE 11
Iterations required for convergence for the 2D model problem.

h	H	S_{VO}	$P_{\Gamma,J}^{-1}S_{VO}$	$P_{\Gamma,CG}^{-1}S_{VO}$
1/128	1/2	110	93	26
	1/3	120	135	25
	1/4	134	94	25
1/256	1/2	140	90	26
	1/3	209	106	28
	1/4	256	102	30
1/512	1/2	148	114	39
	1/3	374	163	40
	1/4	428	186	46

TABLE 12
Iterations required for convergence for the 3D model problem.

h	H	S_{VO}	$P_{\Gamma,J}^{-1}S_{VO}$	$P_{\Gamma,CG}^{-1}S_{VO}$
1/16	1/2	26	16	11
	1/4	33	21	12
1/32	1/2	51	24	14
	1/4	57	29	15
1/64	1/2	65	34	15
	1/4	70	40	17

TABLE 13
CPU time for the 2D model problem, obtained on a LINUX cluster with a tolerance $\|\mathbf{r}_k\|/\|\mathbf{r}_0\| \leq 10^{-5}$.

h	H	S_{VO}	$P_{\Gamma,J}^{-1}S_{VO}$	$P_{\Gamma,CG}^{-1}S_{VO}$
1/128	1/2	3.29	2.92	1.27
	1/3	1.42	1.65	0.60
	1/4	0.99	0.75	0.45
1/256	1/2	24.09	16.51	6.41
	1/3	12.93	6.75	2.92
	1/4	8.80	3.68	1.78
1/512	1/2	208.75	184.36	75.39
	1/3	150.32	65.69	22.34
	1/4	80.89	35.84	12.71

$P_{\Gamma,CG}$ behaves better in both 2D and 3D. However, both $P_{\Gamma,J}$ and $P_{\Gamma,CG}$ converge much faster than the nonpreconditioned system in terms of CPU time, as Table 13 shows.

5. Conclusions. For the numerical solution of large linear systems arising from the discretization of elliptic partial differential equations, we have presented a new approach for the iterative solution of the SC system with many subdomains. The presented method does not need the explicit definition of a coarse space and it is based on the decomposition of the original computational domain in two subdomains, in such a way that one of the two subdomains, say, Ω_Γ, is connected to all the others. This subdomain can be seen as an interface-strip, whose width is of one or more elements.

Among the available preconditioners for nonoverlapping two-subdomain decompositions, we have investigated a DN preconditioner, where a Neumann problem is

solved in Ω_Γ. This (global) Neumann problem acts as a coarse solver: This becomes apparent when considering the problem on the interface-strip solved by eliminating the edges first. The remaining problem on substructure vertices has the same connectivity as the usual vertex-based coarse problem. The strip on the edges of the subdomain interfaces acts basically as a diagonal preconditioner.

As presented in section 3, the condition number of the preconditioned system is of order H/h (therefore getting better as the number of subdomains becomes larger, a property that is interesting for parallel numerical computation). This makes the proposed preconditioner asymptotically equivalent to other methods, such as, for instance, the two-level Schwarz preconditioner with small overlap [6]. Although other two-level preconditioners with a better asymptotic behavior are available in the literature (for example, BNN or FETI), the presented numerical results show that, from the point of view of computational cost, the advocated approach can be competitive. In fact, on one hand it does not require an explicit coarse space and, on the other hand, the preconditioning step needs only the solution of one Neumann problem on a (small) subdomain of special shape, with limited computational cost per iteration.

As shown in Remark 2 and Table 5, another interesting feature of the proposed method is that the condition number of the preconditioned Schur complement does not depend on the jumps of the coefficients.

For a scalar implementation one can factorize and store P_Γ, while on a parallel computer a strategy for solving the global problem associated to P_Γ is necessary. Note also that this linear system does not need to be solved exactly.

Acknowledgment. The authors wish to thank the referees, whose remarks have helped improve the presentation of the paper.

REFERENCES

[1] O. Axelsson, *Iterative Solution Methods*, Cambridge University Press, Cambridge, UK, 1994.
[2] P. E. Bjørstad and O. B. Widlund, *To overlap or not to overlap: A note on a domain decomposition method for elliptic problems*, SIAM J. Sci. Statist. Comput., 10 (1989), pp. 1053–1061.
[3] M. Brezina and P. Vaněk, *A black-box iterative solver based on a two-level Schwarz method*, Computing, 63 (1999), pp. 233–263.
[4] T. F. Chan and T. P. Mathew, *Domain decomposition algorithms*, in Acta Numer., Cambridge University Press, Cambridge, UK, 1994, pp. 61–143.
[5] Y.-H. De Roeck and P. Le Tallec, *Analysis and test of a local domain-decomposition preconditioner*, in Fourth International Symposium on Domain Decomposition Methods for Partial Differential Equations, R. Glowinski et al., eds., SIAM, Philadelphia, 1991, pp. 112–128.
[6] M. Dryja and O. B. Widlund, *Domain decomposition algorithms with small overlap*, SIAM J. Sci. Comput., 15 (1994), pp. 604–620.
[7] M. Dryja, B. F. Smith, and O. B. Widlund, *Schwarz analysis of iterative substructuring algorithms for elliptic problems in three dimensions*, SIAM J. Numer. Anal., 31 (1994), pp. 1662–1694.
[8] C. Farhat, J. Mandel, and F.-X. Roux, *Optimal convergence properties of the FETI domain decomposition method*, Comput. Methods Appl. Mech. Engrg., 115 (1994), pp. 365–385.
[9] L. Formaggia, A. Scheinine, and A. Quarteroni, *A numerical investigation of Schwarz domain decomposition techniques for elliptic problems on unstructured grids*, Math. Comput. Simulation, 44 (1997), pp. 313–330.
[10] L. Formaggia, M. Sala, and F. Saleri, *Domain decomposition techniques*, in Numerical Solutions of Partial Differential Equations on Parallel Computers, A. M. Brauset and A. Tveiko, eds., Springer-Verlag, Berlin, 2006, to appear.
[11] G. H. Golub and C. F. van Loan, *Matrix Computations*, 2nd ed., The Johns Hopkins University Press, Baltimore, MD, 1989.

[12] M. Hestenes and E. Steifel, *Methods of conjugate gradients for solving linear systems*, J. Research Nat. Bur. Standards, 49 (1952), pp. 409–436.
[13] E. W. Jenkins, C. E. Kees, C. T. Kelley, and C. T. Miller, *An aggregation-based domain decomposition preconditioner for groundwater flow*, SIAM J. Sci. Comput., 23 (2001), pp. 430–441.
[14] G. Karypis and V. Kumar, *Multilevel k-way partitioning scheme for irregular graphs*, J. Parallel Distrib. Comput., 48 (1998), pp. 96–129.
[15] C. E. Kees, C. T. Miller, E. W. Jenkins, and C. T. Kelley, *Versatile two-level Schwarz preconditioners for multiphase flow*, Comput. Geosci., 7 (2003), pp. 91–114.
[16] C. Lasser and A. Toselli, *An overlapping domain decomposition preconditioner for a class of discontinuous Galerkin approximations of advection-diffusion problems*, Math. Comp., 72 (2003), pp. 1215–1238.
[17] C. Lasser and A. Toselli, *Convergence of some two-level overlapping domain decomposition preconditioners with smoothed aggregation coarse spaces*, in Recent Developments in Domain Decomposition Methods, L. F. Pavarino and A. Toselli, eds., Springer-Verlag, Berlin, 2002, pp. 95–117.
[18] P. Le Tallec, *Domain decomposition methods in computational mechanics*, Comput. Mech. Adv., 1 (1994), pp. 121–220.
[19] J. Mandel, *Balancing domain decomposition*, Comm. Numer. Methods Engrg., 9 (1993), pp. 233–241.
[20] L. Paglieri, A. Scheinine, L. Formaggia, and A. Quarteroni, *Parallel conjugate gradient with Schwarz preconditioner applied to fluid dynamics problems*, in Parallel Computational Fluid Dynamics '96. Algorithms and Results Using Advanced Computers, P. Schiano et al., eds., Elsevier, Amsterdam, 1997, pp. 21–30.
[21] A. Quarteroni and A. Valli, *Numerical Approximation of Partial Differential Equations*, Springer-Verlag, Berlin, 1994.
[22] A. Quarteroni and A. Valli, *Domain Decomposition Methods for Partial Differential Equations*, Oxford University Press, New York, 1999.
[23] Y. Saad, *Iterative Methods for Sparse Linear Systems*, PWS, Boston, 1996.
[24] M. Sala, *Domain Decomposition Preconditioners: Theoretical Properties, Application to the Compressible Euler Equations, Parallel Aspects*, Ph.D. thesis, École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland, 2003.
[25] M. Sala, *Analysis of two-level domain decomposition preconditioners based on aggregation*, M2AN Math. Model. Numer. Anal., 38 (2004), pp. 765–780.
[26] M. Sala, J. N. Shadid, and R. S. Tuminaro, *An improved convergence bound for aggregation-based domain decomposition preconditioners*, SIAM J. Matrix Anal. Appl., 27 (2005), pp. 744–756.
[27] B. F. Smith, P. Bjørstad, and W. D. Gropp, *Domain Decomposition. Parallel Multilevel Methods for Elliptic Partial Differential Equations*, Cambridge University Press, Cambridge, UK, 1996.
[28] H. Van der Vorst and C. Vuik, *GMRESR: A family of nested GMRES methods*, Numer. Linear Algebra Appl., 1 (1994), pp. 369–386.
[29] P. Vaněk, J. Mandel, and M. Brezina, *Algebraic multigrid by smoothed aggregation for second and fourth order elliptic problems*, Computing, 56 (1996), pp. 179–196.
[30] J. Xu and J. Zou, *Some nonoverlapping domain decomposition methods*, SIAM Rev., 40 (1998), pp. 857–914.

SIAM J. SCI. COMPUT.
Vol. 28, No. 2, pp. 517–532
© 2006 Society for Industrial and Applied Mathematics

A SINGULAR FUNCTION BOUNDARY INTEGRAL METHOD FOR LAPLACIAN PROBLEMS WITH BOUNDARY SINGULARITIES*

CHRISTOS XENOPHONTOS†, MILTIADES ELLIOTIS‡, AND GEORGIOS GEORGIOU‡

Abstract. A singular function boundary integral method for Laplacian problems with boundary singularities is analyzed. In this method, the solution is approximated by the truncated asymptotic expansion for the solution near the singular point and the Dirichlet boundary conditions are weakly enforced by means of Lagrange multiplier functions. The resulting discrete problem is posed and solved on the boundary of the domain, away from the point of singularity. The main result of this paper is the proof of convergence of the method; in particular, we show that the method approximates the generalized stress intensity factors, i.e., the coefficients in the asymptotic expansion, at an exponential rate. A numerical example illustrating the convergence of the method is also presented.

Key words. boundary singularities, boundary approximation methods, Lagrange multipliers, stress intensity factors

AMS subject classifications. 65N15, 65N38, 65N12, 65N30

DOI. 10.1137/050622742

1. Introduction.

In many engineering problems (e.g., fracture mechanics applications) governed by elliptic partial differential equations (PDEs), boundary singularities arise when there is a sudden change in the boundary conditions and/or on the boundary itself. Singularities are known to affect adversely the accuracy and the convergence of standard numerical methods, such as finite element, boundary element, finite difference, and spectral methods. Grid refinement and high-order discretizations are common strategies aimed at improving the convergence rate and accuracy of the above-mentioned standard methods. If, however, the form of the singularity is taken into account and is properly incorporated into the numerical scheme, then a more effective method may be constructed (see, e.g., [10, 15]). (For a recent survey of treatment of singularities in elliptic boundary value problems see [17] and the references therein.)

In the case of the two-dimensional Laplace equation, for example, the form of the singularity is visible through the asymptotic expansion for the solution u near the singular point. When the boundaries sharing the singular point are not curved, u is given by [12]

$$u(r,\theta) = \sum_{j=1}^{\infty} \alpha_j r^{\beta_j} \phi_j(\theta), \tag{1.1}$$

where (r,θ) are polar coordinates centered at the singular point, $\alpha_j \in \mathbb{R}$ are the unknown singular coefficients, and $\beta_j \in \mathbb{R}, \phi_j(\theta)$ are the eigenvalues and eigenfunctions of the problem, respectively, which are uniquely determined by the geometry and the boundary conditions along the boundaries sharing the singular point. The constants

*Received by the editors January 14, 2005; accepted for publication (in revised form) October 28, 2005; published electronically April 7, 2006.
http://www.siam.org/journals/sisc/28-2/62274.html
†Mathematical Sciences Department, Loyola College, 4501 N. Charles Street, Baltimore, MD 21210 (cxenophontos@loyola.edu).
‡Department of Mathematics and Statistics, University of Cyprus, P.O. Box 20537, 1678 Nicosia, Cyprus (eliotis@ucy.ac.cy, georgios@ucy.ac.cy).

α_j are often called [illegible] (GSIFs) or flux intensity factors. In the case of elasticity problems with cracks, these constants are called stress intensity factors (SIFs) and serve as a measure of the stress at which failure occurs. In most commonly used methods, such as the finite element method (FEM), the SIFs are calculated as a postsolution operation (see, e.g., [4, 26, 27]). If, however, the goal of the computation is the calculation of the SIFs, then methods which calculate these quantities [illegible] may be preferable. The singular function boundary integral method (SFBIM), which we will analyze in this article, falls in the latter category. (See also [19, 20, 22, 23] and the references therein for additional information on the determination and importance of the aforementioned coefficients.)

The SFBIM was originally developed by Georgiou and coworkers [10, 11] and was subsequently refined and expanded by Elliotis, Georgiou, and Xenophontos [6, 7, 8, 9]. The method uses the leading terms in the local asymptotic expansion for the solution near the singular point as an approximation, while any Dirichlet boundary conditions are weakly enforced by means of Lagrange multiplier functions. The resulting problem is posed on the boundary of the domain; hence the dimension of the problem is reduced by one, leading to considerable computational cost reduction. We should also mention here the works of Li et al. [14, 15, 16] and Arad et al. [1], who also developed similar methods. (See also [21] for a review of SIF evaluation and modeling of singularities in boundary integral methods.)

The SFBIM has been successfully applied to a number of problems in solid and fluid mechanics and excellent numerical results have been obtained thus far [6, 7, 8, 9]. In particular, it was observed that the method (i) approximates the SIFs at an exponential rate of convergence, (ii) is very efficient, and (iii) compares extremely well with other accurate methods found in the literature. Our main goal in this article is to prove the observed convergence rates of the method.

The rest of the paper is organized as follows: In section 2 we present the formulation of the method for a two-dimensional Laplacian problem with a boundary singularity. In section 3 we present the convergence analysis, and in section 4 we comment on how the method can be efficiently implemented. Finally, section 5 includes the results of some numerical computations illustrating the convergence of the method. Throughout the paper the usual notation $H^k(\Omega)$ will be used for spaces containing functions on the domain $\Omega \subset \mathbb{R}^2$ with boundary $\partial\Omega$, having k generalized derivatives in $L^2(\Omega)$; the norm and seminorm on $H^k(\Omega)$ will be denoted by $\|\cdot\|_{k,\Omega}$ and $|\cdot|_{k,\Omega}$, respectively. Also, the letters C and c will be used to denote generic positive constants independent of any discretization parameters and possibly having different values in each occurrence.

2. The model problem and its formulation. For simplicity we consider the Laplacian problem stated below and depicted graphically in Figure 2.1. Find u such that

$$\Delta u = 0 \text{ in } \Omega, \tag{2.1}$$

$$\frac{\partial u}{\partial n} = 0 \text{ on } S_1, \tag{2.2}$$

$$u = 0 \text{ on } S_2, \tag{2.3}$$

$$u = f(r,\theta) \text{ on } S_3, \tag{2.4}$$

$$\frac{\partial u}{\partial n} = g(r,\theta) \text{ on } S_4, \tag{2.5}$$

where $\partial\Omega = \bigcup_{i=1}^{4} S_i$.

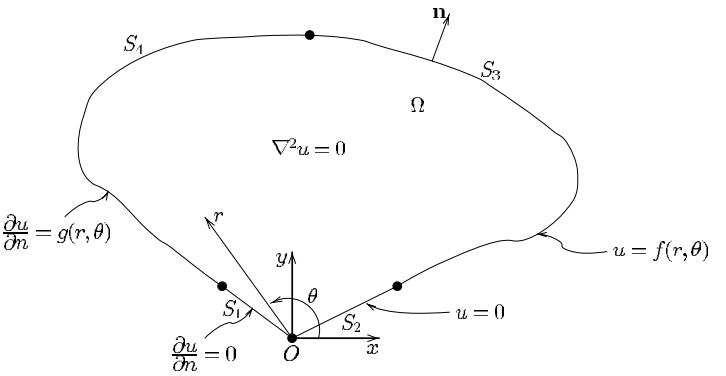


FIG. 2.1. *A two-dimensional Laplace equation problem with one boundary singularity.*

In (2.1)–(2.5), Δ denotes the Laplacian operator, the variables (r, θ) denote polar coordinates centered at O, and the functions f and g are known. It is assumed that f, g, and the boundary $\partial\Omega$ are such that there is only one boundary singularity at O. The local solution near O is given by an asymptotic expansion of the form (1.1) [12].

Even though the presentation and analysis of the method will be restricted to the type of boundary singularity shown in Figure 2.1, any type of boundary singularity for which an asymptotic expansion (1.1) exists can be treated.

Multiplying (2.1) by a test function $v \in V_1$ (to be specified shortly) and integrating over Ω, we get

$$\iint_{\Omega} v\Delta u = 0,$$

and then, by means of Green's theorem, we obtain

$$-\iint_{\Omega} \nabla v \cdot \nabla u + \int_{\partial\Omega} v\frac{\partial u}{\partial n} = 0.$$

Since $\frac{\partial u}{\partial n} = 0$ on S_1 and $\frac{\partial u}{\partial n} = g(r,\theta)$ on S_4, we further have

$$\iint_{\Omega} \nabla v \cdot \nabla u - \int_{S_2} v\frac{\partial u}{\partial n} - \int_{S_3} v\frac{\partial u}{\partial n} = \int_{S_4} vg. \tag{2.6}$$

Suppose v is chosen to satisfy

$$\Delta v = 0 \text{ in } \Omega,\ v = 0 \text{ on } S_2,\ \frac{\partial v}{\partial n} = 0 \text{ on } S_1, \tag{2.7}$$

one such choice being $v \equiv r^{\beta_j}\phi_j(\theta)$ (see (1.1)). Then (2.6) becomes

$$\iint_{\Omega} \nabla v \cdot \nabla u - \int_{S_3} v\frac{\partial u}{\partial n} = \int_{S_4} vg. \tag{2.8}$$

Now, since $u = f$ on S_3 we have

$$-\int_{S_3} \frac{\partial v}{\partial n}(u - f) = 0,$$

so adding this to (2.8), we get

$$\iint_{\Omega} \nabla v \cdot \nabla u - \int_{S_3} v \frac{\partial u}{\partial n} - \int_{S_3} \frac{\partial v}{\partial n}(u - f) = \int_{S_4} vg$$

or, equivalently,

$$\iint_{\Omega} \nabla v \cdot \nabla u - \int_{S_3} v \frac{\partial u}{\partial n} - \int_{S_3} \frac{\partial v}{\partial n} u = \int_{S_4} vg - \int_{S_3} \frac{\partial v}{\partial n} f.$$

Letting

$$\lambda = \left.\frac{\partial u}{\partial n}\right|_{S_3}, \ \mu = \left.\frac{\partial v}{\partial n}\right|_{S_3}, \tag{2.9}$$

the above equation becomes

$$\iint_{\Omega} \nabla v \cdot \nabla u - \int_{S_3} v\lambda - \int_{S_3} \mu u = \int_{S_4} vg - \int_{S_3} \mu f.$$

Hence, the [illegible] to be solved reads as follows: Find $(u, \lambda) \in V_1 \times V_2$ such that

$$B(u, v) + b(u, v; \lambda, \mu) = F(v, \mu) \quad \forall \ (v, \mu) \in V_1 \times V_2, \tag{2.10}$$

where

$$B(u, v) = \iint_{\Omega} \nabla v \cdot \nabla u, \tag{2.11}$$

$$b(u, v; \lambda, \mu) = -\int_{S_3} v\lambda - \int_{S_3} \mu u, \tag{2.12}$$

$$F(v, \mu) = \int_{S_4} vg - \int_{S_3} \mu f. \tag{2.13}$$

The spaces V_1 and V_2 are chosen as follows. Let the [illegible] of functions in $H^1(\Omega)$ be denoted by

$$H^{1/2}(\partial\Omega) = \left\{ u \in H^1(\Omega) : u|_{\partial\Omega} \in L^2(\partial\Omega) \right\}. \tag{2.14}$$

With $T : H^1(\Omega) \to H^{1/2}(\partial\Omega)$ denoting the trace operator, the norm of $H^{1/2}(\partial\Omega)$ is defined as

$$\|\psi\|_{1/2,\partial\Omega} = \inf_{u \in H^1(\Omega)} \left\{ \|u\|_{1,\Omega} : Tu = \psi \right\}. \tag{2.15}$$

Then, we define $H^{-1/2}(\partial\Omega)$ as the [illegible] of $H^0(\partial\Omega) \equiv L^2(\partial\Omega)$ with respect to the norm

$$\|\varphi\|_{-1/2,\partial\Omega} = \sup_{\psi \in H^{1/2}(\partial\Omega)} \frac{\int_{\partial\Omega} \varphi\psi}{\|\psi\|_{1/2,\partial\Omega}}. \tag{2.16}$$

(See, e.g., [24] for more details.) We then define

$$H_*^1(\Omega) = \left\{u \in H^1(\Omega) : u|_{S_2} = 0\right\}, \tag{2.17}$$

and we take

$$V_1 = H_*^1(\Omega),\ V_2 = H^{-1/2}(S_3). \tag{2.18}$$

The discrete problem corresponding to (2.10) then reads as follows: Find $(u_N, \lambda_h) \in [V_1^N \times V_2^h] \subset [H_*^1(\Omega) \times H^{-1/2}(S_3)]$ such that

$$B(u_N, v) + b(u_N, v; \lambda_h, \mu) = F(v, \mu)\ \ \forall\ (v, \mu) \in V_1^N \times V_2^h, \tag{2.19}$$

with $B(u,v), b(u,v;\lambda,\mu)$, and $F(v,\mu)$ given by (2.11)–(2.13), and with V_1^N, V_2^h finite dimensional spaces to be chosen shortly.

[illegible] 1. The above formulation will be used in the analysis of the method; for the implementation, we will use an equivalent boundary integral formulation in which all integrations will be one-dimensional and carried out away from the point causing the singularity. Details will be given in section 4.

3. Error analysis. We begin by defining the finite dimensional spaces V_1^N and V_2^h which will be used in the approximate problem (2.19). First, with

$$v_i \equiv r^{\beta_i}\phi_i(\theta) \tag{3.1}$$

denoting the singular functions, we define the finite dimensional space

$$V_1^N = span\left\{v_i\right\}_{i=1}^N. \tag{3.2}$$

Next, let S_3 be divided into quasiuniform sections $\Gamma_i, i = 1, \ldots, n$, such that $S_3 = \bigcup_{i=1}^n \Gamma_i$. Let $h_i = |\Gamma_i|$ and set $h = \max_{1\le i\le n} h_i$. We assume that for each segment Γ_i, there exists an invertible mapping $\mathcal{F}_i : I \to \Gamma_i$ which maps the interval $I = [-1, 1]$ to Γ_i, and we define the space

$$V_2^h = \left\{\lambda_h : \lambda_h|_{\Gamma_i} \circ \mathcal{F}_i^{-1} \in \mathcal{P}_p(I), i = 1, \ldots, n\right\}, \tag{3.3}$$

where $\mathcal{P}_p(I)$ is the set of polynomials of degree $\le p$ on $I = [-1, 1]$. In practice, the representation of the boundary S_3 determines the mappings $\mathcal{F}_i$; i.e., if S_3 is represented by a polynomial, then an isoparametric mapping may be used, and if S_3 has some (general) parametric representation, then the [illegible] technique may be used (see Ch. 6 in [25]).

We have the following theorem.

THEOREM 3.1. [illegible] (u, λ) [illegible] (u_N, λ_h) [illegible] (2.10) [illegible] (2.19) [illegible] c_0, c, β [illegible] γ [illegible] N [illegible] h [illegible]

$$B(v,v) \ge c_0 \|v\|_{1,\Omega}^2 \quad |B(u,v)| \le c\|u\|_{1,\Omega}\|v\|_{1,\Omega}\ \forall v \in V_1^N, \tag{3.4}$$

$$\exists\, 0 \ne v_N \in V_1^N \quad \left|\int_{S_3} \mu_h v_N\right| \ge \beta \|\mu_h\|_{-1/2,S_3}\|v_N\|_{1,\Omega}\ \forall \mu_h \in V_2^h, \tag{3.5}$$

$$\left|\int_{S_3} \lambda v\right| \le \gamma \|\lambda\|_{-1/2,S_3}\|v\|_{1,\Omega}\ \forall v \in V_1^N. \tag{3.6}$$

(3.7)
$$\|u-u_N\|_{1,\Omega}+\|\lambda-\lambda_h\|_{-1/2,S_3}\leq C\left\{\inf_{v\in V_1^N}\|u-v\|_{1,\Omega}+\inf_{\eta\in V_2^h}\|\lambda-\eta\|_{-1/2,S_3}\right\},$$

$C\in\mathbb{R}^+$ N h

This is similar to the classical result for saddle point problems (cf. [5]), and its proof for our problem appears in the appendix; see also Theorem 6.1 in [14].

Before verifying that (3.4)–(3.6) hold for the problem under consideration, we will make certain assumptions that will aid in the analysis. First, we note that for any function w which can be expressed in the form given by (1.1), we can always write

$$w=w_N+r_N,$$

where

$$w_N=\sum_{j=1}^{N}\alpha_j v_j\in V_1^N,\ r_N=\sum_{j=N+1}^{\infty}\alpha_j v_j. \tag{3.8}$$

We will assume that there exists $a\in(0,1)$ such that

$$|r_N|\leq Ca^N \tag{A1}$$

and

$$\left|\frac{\partial r_N}{\partial r}\right|\leq CNa^N. \tag{A2}$$

If $r<1$ in (1.1), assumptions (A1), (A2) may be replaced with the assumption that $|\alpha_j|<\infty\ \forall\, j$, since then, by (3.1) and the fact that ϕ_j is harmonic,

$$|r_N|\leq\sum_{j=N+1}^{\infty}|\alpha_j|\,r^{\beta_j}\leq C\frac{r^{\beta_{N+1}}}{1-r}\leq Ca^N,$$

with $r<a<1$, and $C\in\mathbb{R}^+$ independent of a and N. Similarly,

$$\begin{aligned}\left|\frac{\partial r_N}{\partial r}\right|&\leq\sum_{j=N+1}^{\infty}|\alpha_j|\,r^{\beta_j-1}=\sum_{j=N+1}^{\infty}|\alpha_j|\left\{\frac{d}{dr}\int_0^r\rho^{\beta_j-1}d\rho\right\}=\frac{d}{dr}\sum_{j=N+1}^{\infty}|\alpha_j|\left\{\int_0^r\rho^{\beta_j-1}d\rho\right\}\\&\leq\frac{d}{dr}\sum_{j=N+1}^{\infty}|\alpha_j|\,r^{\beta_j}\leq C\frac{d}{dr}\left(\frac{r^{\beta_{N+1}}}{1-r}\right)\leq CNa^N.\end{aligned}$$

If $r\geq 1$, one may partition the domain Ω into subdomains in which separate approximations may be obtained, including one near O which is valid for $r<1$. The solution over the entire domain can then be composed by combining the solutions from each subdomain and properly dealing with their interactions across the interfaces separating them (see, e.g., [18]).

Let us now verify that (3.4)–(3.6) hold for the problem given by (2.19). First, note that $B(v,v)=|v|_{1,\Omega}^2$ so that, by Poincaré's inequality,

$$B(v,v)\geq c_0\|v\|_{1,\Omega}^2\ \forall v\in H_*^1(\Omega). \tag{3.9}$$

By the Cauchy–Schwarz inequality,

$$B(u,v) \le c\,\|u\|_{1,\Omega}\,\|v\|_{1,\Omega}\ \forall u,v \in H^1_*(\Omega), \tag{3.10}$$

so that (3.9) and (3.10) give (3.4). To verify (3.5), consider the following auxiliary problem (for which a unique solution exists): Find w such that

$$\Delta w = 0 \text{ in } \Omega, \tag{3.11}$$

$$\frac{\partial w}{\partial n} = \mu_h \text{ on } S_3, \tag{3.12}$$

$$w = 0 \text{ on } S_2, \tag{3.13}$$

$$\frac{\partial w}{\partial n} = 0 \text{ on } S_1 \cup S_4, \tag{3.14}$$

where $\mu_h \in V_2^h$ in (3.12). From (3.11) and (3.12) we obtain (using Green's formula and Poincaré's inequality)

$$\int_{S_3} \mu_h w = \int_{S_3} w\frac{\partial w}{\partial n} = \iint_{\Omega} w\Delta w + \iint_{\Omega} |\nabla w|^2 = |w|^2_{1,\Omega} \ge c_0\,\|w\|^2_{1,\Omega}\,, \tag{3.15}$$

with $c_0 \in \mathbb{R}^+$. Also (cf. [2])

$$\|\mu_h\|_{-1/2,S_3} = \left\|\frac{\partial w}{\partial n}\right\|_{-1/2,S_3} \le C\,\|w\|_{1,\Omega}\,, \tag{3.16}$$

so that by (3.15) and (3.16)

$$\int_{S_3} \mu_h w \ge c_0\,\|w\|^2_{1,\Omega} \ge \beta\,\|w\|_{1,\Omega}\,\|\mu_h\|_{-1/2,S_3}, \tag{3.17}$$

with $\beta \in \mathbb{R}^+$ independent of w and h. Now, let $w_N \in V_1^N$ be such that $w = w_N + r_N$, as given by (3.8). We have

$$\int_{S_3} \mu_h w_N = \int_{S_3} \mu_h w - \int_{S_3} \mu_h r_N, \tag{3.18}$$

and also

$$\int_{S_3} \mu_h r_N \le \|\mu_h\|_{-1/2,S_3}\,\|r_N\|_{1/2,S_3} \le C_1\,\|\mu_h\|_{-1/2,S_3}\,\|r_N\|_{1,\Omega}, \tag{3.19}$$

so that, combining (3.17)–(3.19), we get

$$\int_{S_3} \mu_h w_N \ge \beta\,\|w\|_{1,\Omega}\,\|\mu_h\|_{-1/2,S_3} - C_1\,\|\mu_h\|_{-1/2,S_3}\,\|r_N\|_{1,\Omega}. \tag{3.20}$$

Now, using the reverse triangle inequality, we have

$$\|w\|_{1,\Omega} = \|w_N + r_N\|_{1,\Omega} \ge \|w_N\|_{1,\Omega} - \|r_N\|_{1,\Omega}\,,$$

which along with (3.20) gives

$$\begin{aligned}\int_{S_3} \mu_h w_N &\geq \beta \left(\|w_N\|_{1,\Omega} - \|r_N\|_{1,\Omega} \right) \|\mu_h\|_{-1/2,S_3} - C_1 \|\mu_h\|_{-1/2,S_3} \|r_N\|_{1,\Omega} \\ &\geq \beta \|w_N\|_{1,\Omega} \|\mu_h\|_{-1/2,S_3} - (C_1+\beta) \|\mu_h\|_{-1/2,S_3} \|r_N\|_{1,\Omega} . \end{aligned} \tag{3.21}$$

Since by assumption (A1), r_N converges to 0 exponentially (or, equivalently, w_N converges to w exponentially), we have

$$\lim_{N\to\infty} \frac{\|r_N\|_{1,\Omega}}{\|w_N\|_{1,\Omega}} = 0,$$

which means that for any $\varepsilon > 0$ there exists N^* such that

$$\frac{\|r_N\|_{1,\Omega}}{\|w_N\|_{1,\Omega}} < \varepsilon$$

whenever $N > N^*$. Hence, for N sufficiently large we may write

$$\frac{\|r_N\|_{1,\Omega}}{\|w_N\|_{1,\Omega}} < \frac{\beta}{2\,(C_1+\beta)}, \tag{3.22}$$

where C_1 and β are the constants from above. Combining (3.21) and (3.22) leads to

$$\int_{S_3} \mu_h w_N \geq \frac{\beta}{2} \|\mu_h\|_{-1/2,S_3} \|w_N\|_{1,\Omega} ,$$

which in turn gives (3.5) once we replace w_N by v_N and $\beta/2$ by β. Condition (3.6) follows directly from the definition of the $H^{-1/2}$-norm (see also (3.19)). The preceding discussion leads to the following theorem.

THEOREM 3.2. [illegible] (u,λ) [illegible] (u_N,λ_h) [illegible] (2.10) [illegible] (2.19) [illegible] $\lambda \in H^k(S_3)$ [illegible] $k \geq 1$ [illegible] C [illegible] N, h [illegible] $a \in (0,1)$ [illegible]

$$\|u-u_N\|_{1,\Omega} + \|\lambda-\lambda_h\|_{-1/2,S_3} \leq C \left\{ \sqrt{N} a^N + h^m p^{-k} \right\}, \tag{3.23}$$

[illegible] $m = \min\{k, p+1\}$.

[illegible]. From Theorem 3.1 we have

$$\|u-u_N\|_{1,\Omega} + \|\lambda-\lambda_h\|_{-1/2,S_3} \leq C \left\{ \inf_{v\in V_1^N} \|u-v\|_{1,\Omega} + \inf_{\eta\in V_2^h} \|\lambda-\eta\|_{-1/2,S_3} \right\}.$$

Now,

$$\inf_{v\in V_1^N} \|u-v\|_{1,\Omega} \leq \|u-w_N\|_{1,\Omega} = \|r_N\|_{1,\Omega} ,$$

with w_N, r_N given by (3.8). Using (A1) and (A2) we get

$$\inf_{v\in V_1^N} \|u-v\|_{1,\Omega} \leq \|r_N\|_{0,\Omega} + |r_N|_{1,\Omega} \leq C \left\{ a^N + \sqrt{N} a^N \right\} \leq C\sqrt{N} a^N, \tag{3.24}$$

with $C \in \mathbb{R}^+$ independent of N. Next, let λ_I be the pth-order interpolant of λ on the partition $\{\Gamma_i\}_{i=1}^n$ of S_3. We have (cf. [3])

$$\|\lambda - \lambda_I\|_{0,S_3} \leq Ch^m p^{-k} \|\lambda\|_{k,S_3}, \tag{3.25}$$

where $m = \min\{k, p+1\}$, and $C > 0$ a constant independent of h and p. Now,

$$\inf_{\eta \in V_2^h} \|\lambda - \eta\|_{-1/2,S_3} \leq \|\lambda - \lambda_I\|_{-1/2,S_3} \leq \|\lambda - \lambda_I\|_{0,S_3}, \tag{3.26}$$

so that, since $\lambda \in H^k(S_3)$, (3.24)–(3.26) give the desired result. □

REMARK 2. The above theorem shows that if the number of singular functions N is increased then u_N converges to u at an exponential rate. The theorem also shows that the convergence of λ_h to λ can occur in one of three ways: (i) by keeping p fixed and reducing h, (ii) by keeping h fixed and increasing p, or (iii) by doing both. These three "options" loosely correspond to the three versions of the FEM, namely, the h, p, and hp versions (cf. [3]).

REMARK 3. Based on the above theorem, one may obtain the "optimal" matching between N and h, i.e., the relationship between the number of singular functions and the number of Lagrange multipliers used in the method, by choosing them in such a way so that the error in (3.23) is balanced. For example, in the case when p is kept fixed and $h \to 0$, we take $h^{p+1} \approx \sqrt{N} a^N$. This leads to the following approximate expression for N:

$$N \approx (p+1) \left| \frac{\ln h}{\ln a} \right|. \tag{3.27}$$

The approximation of the GSIFs is given by the following corollary.

COROLLARY 3.3.

$$u = \sum_{j=1}^{\infty} \alpha_j r^{\beta_j} \phi_j(\theta)$$

$$u_N = \sum_{j=1}^{N} \alpha_j^N r^{\beta_j} \phi_j(\theta)$$

(2.10) (2.19) α_j, α_j^N C N $a \in (0,1)$

$$\left|\alpha_j - \alpha_j^N\right| \leq Ca^N. \tag{3.28}$$

. This is a direct consequence of (A1) and the fact that

$$\left|\alpha_j - \alpha_j^N\right| \leq C \|u - u_N\|_{0,\Omega}.$$

(See also (3.24).) □

4. Implementation. We now give a description of the implementation in order to emphasize the properties of the method. As mentioned in Remark 1, the discretized equation will be solved on the boundary of the domain; here we describe how this is done. Recall the discrete version of the problem given by (2.19), which may be rewritten in the form as follows: Find $(u_N, \lambda_h) \in [V_1^N \times V_2^h]$ such that

$$\iint_\Omega \nabla v \cdot \nabla u_N - \int_{S_3} v\lambda_h = \int_{S_4} vg \;\forall\; v \in V_1^N, \tag{4.1}$$

$$\int_{S_3} \mu\,(u_N - f) = 0 \;\forall\; \mu \in V_2^h. \tag{4.2}$$

We wish to reduce the double integral in (4.1) to a boundary one. To this end, using Green's theorem, we have

$$\iint_\Omega \nabla v \cdot \nabla u_N = \int_{\partial\Omega} \frac{\partial v}{\partial n} u_N - \iint_\Omega u_N \Delta v = \int_{\partial\Omega} \frac{\partial v}{\partial n} u_N$$

by (2.7). Moreover, since $\frac{\partial v}{\partial n}\big|_{S_1} = 0$ by (2.7), and $u_N|_{S_2} = 0$ by (2.17), we get

$$\int_{\partial\Omega} \frac{\partial v}{\partial n} u_N = \int_{S_3} \frac{\partial v}{\partial n} u_N + \int_{S_4} \frac{\partial v}{\partial n} u_N,$$

and (4.1)–(4.2) may be written as follows: Find $(u_N, \lambda_h) \in [V_1^N \times V_2^h]$ such that

$$\int_{S_3} \frac{\partial v}{\partial n} u_N + \int_{S_4} \frac{\partial v}{\partial n} u_N - \int_{S_3} v\lambda_h = \int_{S_4} vg \;\forall\; v \in V_1^N, \tag{4.3}$$

$$\int_{S_3} \mu\,(u_N - f) = 0 \;\forall\; \mu \in V_2^h. \tag{4.4}$$

Obviously if $(u_N, \lambda_h) \in [V_1^N \times V_2^h]$ solves (2.19) (or (4.1)–(4.2)), then it also solves (4.3)–(4.4); it is straightforward to reverse the above steps and see that the two-dimensional formulation analyzed in the previous section is in fact equivalent to the one-dimensional formulation given by (4.3)–(4.4). The latter are the equations which we discretize, since they are posed on the boundary of the domain Ω. This reduces the dimension of the problem by one and leads to significant computational savings. Now, to obtain the linear system of equations corresponding to (4.3)–(4.4), we write

$$u_N = \sum_{j=1}^{N} \alpha_j^N v_j \in V_1^N, \tag{4.5}$$

and

$$\lambda_h = \sum_{i=1}^{M} \gamma_i \psi_i \in V_2^h, \tag{4.6}$$

with α_j^N and γ_k the unknowns in the system, and $V_1^N = span\,\{v_j\}_{j=1}^N$, $V_2^h = span\,\{\psi_i\}_{i=1}^N$. Upon inserting these into (4.3)–(4.4), we obtain the $(N+M)\times(N+M)$ linear system

$$\begin{bmatrix} K_1 & K_2 \\ K_2^T & 0 \end{bmatrix} \begin{bmatrix} \overrightarrow{\alpha} \\ \overrightarrow{\gamma} \end{bmatrix} = \begin{bmatrix} \overrightarrow{G} \\ \overrightarrow{F} \end{bmatrix}, \tag{4.7}$$

where $\overrightarrow{\alpha} = \left[\alpha_1^N, \ldots, \alpha_N^N\right]^T$, $\overrightarrow{\gamma} = [\gamma_1, \ldots, \gamma_M]^T$, and

$$[K_1]_{j,k} = \int_{S_3} v_k \frac{\partial v_j}{\partial n} + \int_{S_4} v_k \frac{\partial v_j}{\partial n}\ ,\ j = 1, \ldots, N,\ k = 1, \ldots, N,$$

$$[K_2]_{j,i} = -\int_{S_3} v_j \psi_i\ ,\ j = 1, \ldots, N,\ i = 1, \ldots, M,$$

$$\left[\overrightarrow{G}\right]_j = \int_{S_4} v_j g,\ j = 1, \ldots, N,$$

$$\left[\overrightarrow{F}\right]_i = \int_{S_3} f\psi_i,\ i = 1, \ldots, M.$$

It is easily shown that the coefficient matrix in (4.7) is symmetric. This matrix, however, is singular if $N < M$. Hence, we should choose N larger than M, but not too large since for excessively large values of N the linear system (4.7) becomes ill conditioned and the results obtained are unreliable. The relationship between N and h (hence M) described in Remark 2 should be our guide in deciding how large to choose these values; see also section 5. As a final remark in this section, we should point out that all integrals involved in the determination of the coefficient matrix (and right-hand side) in (4.7) are along the sides of the domain that [illegible]. Moreover, they are one-dimensional and can be approximated by standard techniques, such as Gaussian quadrature.

5. Numerical results. Even though the analysis of the method was carried out for the rather general case of a domain having curved boundaries (except the ones intersecting at the singular point), and the nonhomogeneous boundary conditions can be given by any smooth function, we will consider for simplicity the model problem depicted in Figure 5.1, originally studied in [13]. The numerical results presented here correspond to the following choices of all relevant parameters: The Lagrange multiplier function λ_h used to impose the Dirichlet condition along S_3 is expanded in terms of [illegible] basis functions ψ_i (see (4.6)), and boundary S_3 is divided into $2n$ quadratic elements of equal size. For the integration, boundary S_4 is also subdivided into n intervals of equal size. All integrals involved are calculated numerically by subdividing each interval above into 10 subintervals and using a 15-point Gauss–Legendre quadrature over each one. In computing the coefficient matrix in (4.7), its symmetry is taken into account.

To determine the relationship between the number of singular functions N and the number of Lagrange multipliers M, we proceed as follows: Since for λ_h we are using $p = 2$ and $h = 2/n$, we have $M = 2n + 1$. For the moment, we fix $n = 8$ (say), which amounts to $M = 17$, and solve the linear system (4.7) for various values of $N > M$ (e.g., $N = 19, 21, 23, \ldots$). We concentrate only on the calculation of the first GSIF α_1^N and record our results in Table 5.1. From the table we see that, for this choice for M, the value of α_1^N is converged up to 14 significant digits once $N = 35$. Moreover, from (3.27) we have

$$N \approx (p+1)\frac{\ln h}{\ln a} = (p+1)\frac{\ln(2/n)}{\ln a} = (p+1)\frac{\ln\left(\frac{2p}{M-1}\right)}{\ln a}$$

$$\Rightarrow \ln a \approx \frac{p+1}{N} \ln\left(\frac{2p}{M-1}\right) \Rightarrow a \approx \left(\frac{2p}{M-1}\right)^{\frac{p+1}{N}};$$

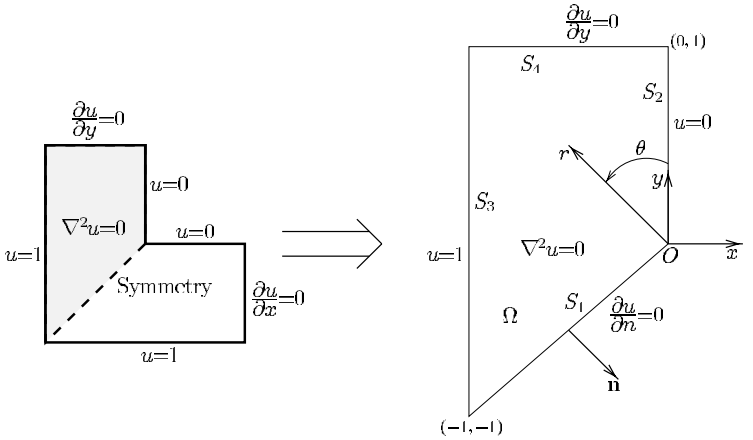


FIG. 5.1. *Geometry and boundary conditions for the model problem.*

TABLE 5.1
Approximate GSIF α_1^N *with* $M = 17$.

N	α_1^N
19	1.12797929883135
21	1.12798071689013
23	1.12798444682112
25	1.12798040013968
27	1.12798040107216
29	1.12798040105877
31	1.12798040105983
33	1.12798040105939
35	1.12798040105939

TABLE 5.2
Approximate GSIFs α_j^N, $j = 1, \ldots, 5$.

j	α_j^N (SFBIM)	α_j^N (*hp*-FEM)	α_j^N (Ref. [13])
1	1.12798040105939	1.12798010	1.1280
2	0.16993386650225	0.16993387	0.1699
3	−0.02304097399348	−0.0230419	−0.0230
4	0.0034711966582	0.0034755	0.0035
5	0.0009151570991	0.0009126	0.0009

hence, using $M = 17$ and $N = 35$, we find that $a \approx 0.89$. With a known, we may now compute the rest of the GSIFs and/or any other quantities of interest by choosing N and M via (3.27). For example, for $M = 41$ (i.e., $h = 1/10$), we find that $N \approx 60$.

Table 5.2 above shows the converged approximate values for the first five GSIFs obtained using the SFBIM (with $M = 41$ and $N = 60$), as well as the *hp* version of the FEM, which is considered to be the state-of-the-art method for problems with singularities [25]; the values from reference [13] are also included for comparison.

These results suggest that the SFBIM can be an attractive (and often preferable) method for problems in which the GSIFs are the main goal of the computation. Figure 5.2 shows the convergence of the leading GSIFs with N; in particular, the figure shows a semilog plot of the estimated percentage relative error in α_j^N, $j = 1, \ldots, 5$; since no

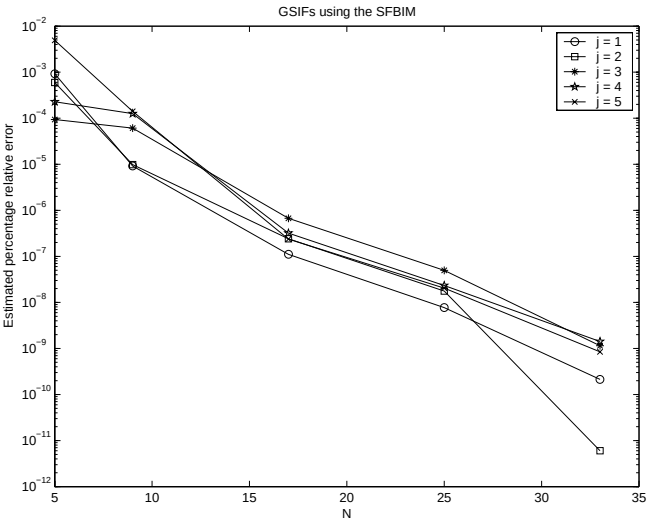


FIG. 5.2. *Convergence of* α_j^N, $j = 1, \ldots, 5$, *using the SFBIM.*

exact values exist for the true GSIFs, we use α_j^{60} as our reference/"exact" value. Since each error curve is essentially straight, this illustrates the exponential convergence of the method, as predicted by Corollary 3.3.

Appendix. Proof of Theorem 3.1. Here, for completeness, we give a proof of Theorem 3.1. First, we note that since (u, λ) also satisfy (2.19), we have $\forall\ (v, \mu) \in V_1^N \times V_2^h$

$$B(u - u_N, v) = -b(u - u_N, v; \lambda - \lambda_h, \mu) = \int_{S_3} v(\lambda - \lambda_h) + \int_{S_3} \mu(u - u_N). \tag{A.1}$$

Since $u = f$ on S_3 and $\int_{S_3} \mu(u_N - f) = 0\ \forall\ \mu \in V_2^h$, we have

$$\int_{S_3} \mu u_N = \int_{S_3} \mu u\ \forall \mu \in V_2^h, \tag{A.2}$$

and thus the last integral in (A.1) is zero. Hence,

$$B(u - u_N, v) = \int_{S_3} v(\lambda - \lambda_h). \tag{A.3}$$

Letting $w = u_N - v, v \in V_1^N$, we get

$$B(u_N - v, w) = B(u_N - u, w) + B(u - v, w) = B(u - v, w) - \int_{S_3} w(\lambda - \lambda_h)$$

$$= B(u - v, w) - \int_{S_3} w(\lambda - \eta) - \int_{S_3} w(\eta - \lambda_h).$$

By means of (A.2), the above equation becomes

$$B(u_N - v, w) = B(u - v, w) - \int_{S_3} w(\lambda - \eta) - \int_{S_3} (\lambda_h - \eta)(u - v). \tag{A.4}$$

Now, from (3.4), (3.6), and (A.4) we get

$$\begin{aligned} c_0 \|u_N - u\|_{1,\Omega}^2 &= c_0 \|w\|_{1,\Omega}^2 \\ &\le \|u - v\|_{1,\Omega} \|w\|_{1,\Omega} + \gamma \|w\|_{1,\Omega} \|\lambda - \eta\|_{-1/2,S_3} + \gamma \|u - v\|_{1,\Omega} \|\lambda_h - \eta\|_{-1/2,S_3} \\ &\le C\big\{(\|u - v\|_{1,\Omega} + \|\lambda - \eta\|_{-1/2,S_3}) \|w\|_{1,\Omega} + \|u - v\|_{1,\Omega} \|\lambda_h - \eta\|_{-1/2,S_3}\big\}. \end{aligned}$$

This is an inequality of order 2:

$$c_0 x^2 \le bx + d\ ,\ b, d > 0,$$

where

$$x = \|w\|_{1,\Omega}\ ,\ b = C(\|u - v\|_{1,\Omega} + \|\lambda - \eta\|_{-1/2,S_3}),$$

$$d = C \|u - v\|_{1,\Omega} \|\lambda_h - \eta\|_{-1/2,S_3} \le \frac{C}{2} \left(\varepsilon \|\lambda_h - \eta\|_{-1/2,S_3} + \frac{1}{\varepsilon} \|u - v\|_{1,\Omega} \right)^2,\ \varepsilon > 0.$$

Therefore, we obtain the bound

$$x \le \frac{b + \sqrt{b^2 + 4c_0 d}}{2c_0}$$

or, equivalently,

(A.5)
$$\|w\|_{1,\Omega} \le C \left\{ \|u - v\|_{1,\Omega} + \|\lambda - \eta\|_{-1/2,S_3} + \frac{1}{\varepsilon} \|u - v\|_{1,\Omega} \right\} + C\varepsilon \|\lambda_h - \eta\|_{-1/2,S_3}.$$

Next, using (3.5) with $\mu_h = \lambda_h - \eta$, we find that there exists $0 \neq v_N \in V_1^N$ such that

$$\|\lambda_h - \eta\|_{-1/2,S_3} \le \beta \frac{\left| \int_{S_3} (\lambda_h - \eta) v_N \right|}{\|v_N\|_{1,\Omega}}. \tag{A.6}$$

Also, it follows from (A.3) that

$$\begin{aligned} \int_{S_3} (\lambda_h - \eta) v_N &= \int_{S_3} (\lambda_h - \lambda) v_N + \int_{S_3} (\lambda - \eta) v_N = B(u - u_N, v_N) + \int_{S_3} (\lambda - \eta) v_N \\ &\le c \|u - u_N\|_{1,\Omega} \|v_N\|_{1,\Omega} + \gamma \|v_N\|_{1,\Omega} \|\lambda - \eta\|_{-1/2,S_3}, \end{aligned} \tag{A.7}$$

so that by (A.6) and (A.7),

$$\begin{aligned} \|\lambda_h - \eta\|_{-1/2,S_3} &\le C \left\{ \|u - u_N\|_{1,\Omega} + \|\lambda - \eta\|_{-1/2,S_3} \right\} \\ &\le C \left\{ \|u - v\|_{1,\Omega} + \|v - u_N\|_{1,\Omega} + \|\lambda - \eta\|_{-1/2,S_3} \right\}. \end{aligned}$$

Since $\|v - u_N\|_{1,\Omega} = \|w\|_{1,\Omega}$, we have from (A.5)

$$\begin{aligned} \|\lambda_h - \eta\|_{-1/2,S_3} \le C_1 \Big\{ &\|u - v\|_{1,\Omega} + \|\lambda - \eta\|_{-1/2,S_3} + \varepsilon^{-1} \|u - v\|_{1,\Omega} \Big\} \\ &+ \varepsilon \|\lambda_h - \eta\|_{-1/2,S_3} \end{aligned}$$

with $C_1 \in \mathbb{R}^+$ independent of v, η, and ε. Letting $C_1\varepsilon \leq 1/2$ we get

$$\|\lambda_h - \eta\|_{-1/2,S_3} \leq C\left\{\|u - v\|_{1,\Omega} + \|\lambda - \eta\|_{-1/2,S_3}\right\}. \tag{A.8}$$

Using the triangle inequality, we immediately get

$$\begin{aligned}\|\lambda - \lambda_h\|_{-1/2,S_3} &\leq \|\lambda - \eta\|_{-1/2,S_3} + \|\lambda_h - \eta\|_{-1/2,S_3} \\ &\leq C\left\{\|u - v\|_{1,\Omega} + \|\lambda - \eta\|_{-1/2,S_3}\right\}.\end{aligned} \tag{A.9}$$

Similarly, using (A.5) and (A.9), we get

$$\begin{aligned}\|u - u_N\|_{1,\Omega} &\leq \|u - v\|_{1,\Omega} + \|v - u_N\|_{1,\Omega} = \|u - v\|_{1,\Omega} + \|w\|_{1,\Omega} \\ &\leq C\left\{\|u - v\|_{1,\Omega} + \|\lambda - \eta\|_{-1/2,S_3}\right\},\end{aligned}$$

which gives the desired result.

Acknowledgments. The authors would like to thank the anonymous referees for the valuable comments which led to an improved manuscript.

REFERENCES

[1] M. Arad, Z. Yosibash, G. Ben-Dor, and A. Yakhot, *Comparing the flux intensity factors by a boundary element method for elliptic equations with singularities*, Comm. Numer. Methods Engrg., 14 (1998), pp. 657–670.

[2] I. Babuška, *The finite element method with Lagrangian multipliers*, Numer. Math., 20 (1973), pp. 179–192.

[3] I. Babuška and M. Suri, *The p and h-p versions of the finite element method, basic principles and properties*, SIAM Rev., 36 (1994), pp. 578–632.

[4] S. C. Brenner, *Multigrid methods for the computation of singular solutions and stress intensity factors* I*: Corner singularities*, Math. Comp., 68 (1999), pp. 559–583.

[5] F. Brezzi and M. Fortin, *Mixed and Hybrid Finite Element Methods*, Springer-Verlag, Berlin, 1991.

[6] M. Elliotis, G. Georgiou, and C. Xenophontos, *The solution of Laplacian problems over L-shaped domains with a singular function boundary integral method*, Comm. Numer. Methods Engrg., 18 (2002), pp. 213–222.

[7] M. Elliotis, G. Georgiou, and C. Xenophontos, *Solving Laplacian problems with boundary singularities: A comparison of a singular function boundary integral method with the p/hp version of the finite element method*, Appl. Math. Comput., 169 (2005), pp. 485–499.

[8] M. Elliotis, G. Georgiou, and C. Xenophontos, *Solution of the planar Newtonian stick-slip problem with the singular function boundary integral method*, Internat. J. Numer. Methods Fluids, 48 (2005), pp. 1001–1021.

[9] M. Elliotis, G. Georgiou, and C. Xenophontos, *The singular function boundary integral method for a two-dimensional fracture problem*, Engineering Analysis with Boundary Elements, 30 (2006), pp. 100–106.

[10] G. C. Georgiou, L. G. Olson, and Y. S. Smyrlis, *A singular function boundary integral method for the Laplace equation*, Comm. Numer. Methods Engrg., 12 (1996), pp. 127–134.

[11] G. Georgiou, A. Boudouvis, and A. Poullikkas, *Comparison of two methods for the computation of singular solutions in elliptic problems*, J. Comput. Appl. Math., 79 (1997), pp. 277–290.

[12] P. Grisvard, *Singularities in Boundary Value Problems*, Springer-Verlag, Berlin, 1992.

[13] H. Igarashi and T. Honma, *A boundary element method for potential fields with corner singularities*, Appl. Math. Modelling, 20 (1996), pp. 847–852.

[14] Z.-C. Li, *Combined Methods for Elliptic Equations with Singularities, Interfaces and Infinities*, Kluwer Academic Publishers, Dordrecht, The Netherlands, 1998.

[15] Z.-C. Li, *Penalty combinations of Ritz-Galerkin and finite-difference methods for singularity problems*, J. Comput. Appl. Math., 81 (1997), pp. 1–17.

[16] Z.-C. Li, Y. L. Chan, G. Georgiou, and C. Xenophontos, *Special boundary approximation methods for Laplace's equation with singularities*, Comput. Math. Appl., 51 (2006), pp. 115–142.
[17] Z.-C. Li and T. T. Lu, *Singularities and treatments of elliptic boundary value problems*, Math. Comput. Modelling, 31 (2000), pp. 97–145.
[18] Z.-C. Li, R. Mathon, and P. Sermer, *Boundary methods for solving elliptic problems with singularities and interfaces*, SIAM J. Numer. Anal., 24 (1987), pp. 487–498.
[19] V. G. Maz'ya, S. A. Nazarov, and B. A. Plamenevskiĭ, *Calculation of the asymptotics of coefficients of intensity in the coming together of corner or conic points*, Zh. Vychisl. Mat. Mat. Fiz., 23 (1983), pp. 333–346 (in Russian).
[20] V. G. Maz'ya and B. A. Plamenevskiĭ, *Coefficients in asymptotics of solutions to elliptic boundary value problems in domains with conic points*, Math. Nachr., 76 (1977), pp. 29–60.
[21] N. K. Mukhopadhyay, S. K. Maiti, and A. Kakodkar, *A review of SIF evaluation and modelling of singularities in BEM*, Comput. Mech., 25 (2000), pp. 358–375.
[22] E. Nicaise and A. M. Sändig. *General interface problems.* I, II, Math. Methods Appl. Sci., 17 (1994), pp. 395–429, 431–450.
[23] E. Nicaise and A. M. Sändig. *Transmission problems for the Laplace and elasticity operators: Regularity and boundary integral formulation*, Math. Models Methods Appl. Sci., 9 (1999), pp. 855–898.
[24] J. Pitkäranta, *Boundary subspaces for the finite element method with Lagrange multipliers*, Numer. Math., 33 (1973), pp. 273–289.
[25] B. A. Szabó and I. Babuška, *Finite Element Analysis*, John Wiley & Sons, New York, 1991.
[26] Z. Yosibash and B. A. Szabó, *Numerical analysis of singularities in two dimensions. Part* 1: *Computation of the eigenpairs*, Internat. J. Numer. Methods Engrg., 38 (1995), pp. 2055–2082.
[27] B. A. Szabó and Z. Yosibash, *Numerical analysis of singularities in two dimensions. Part* 2: *Computation of generalized flux/stress intensity factors*, Internat. J. Numer. Methods Engrg., 39 (1996), pp. 409–434.

SIAM J. SCI. COMPUT.
Vol. 28, No. 2, pp. 533–560
© 2006 Society for Industrial and Applied Mathematics

NONOSCILLATORY CENTRAL SCHEMES FOR ONE- AND TWO-DIMENSIONAL MAGNETOHYDRODYNAMICS EQUATIONS. II: HIGH-ORDER SEMIDISCRETE SCHEMES*

JORGE BALBÁS† AND EITAN TADMOR‡

Abstract. We present a new family of high-resolution, nonoscillatory semidiscrete central schemes for the approximate solution of the ideal magnetohydrodynamics (MHD) equations. This is the second part of our work, where we are passing from the fully discrete staggered schemes in [J. Balbás, E. Tadmor, and C.-C. Wu, *J. Comput. Phys.*, 201 (2004), pp. 261–285] to the semidiscrete formulation advocated in [A. Kurganov and E. Tadmor, *J. Comput. Phys.*, 160 (2000), pp. 241–282]. This semidiscrete formulation retains the simplicity of fully discrete central schemes while enhancing efficiency and adding versatility. The semidiscrete algorithm offers a wider range of options to implement its two key steps: nonoscillatory reconstruction of point values followed by the evolution of the corresponding point valued fluxes. We present the solution of several prototype MHD problems. Solutions of one-dimensional Brio–Wu shock-tube problems and the two-dimensional Kelvin–Helmholtz instability, Orszag–Tang vortex system, and the disruption of a high density cloud by a strong shock are carried out using third- and fourth-order central schemes based on the central WENO reconstructions. These results complement those presented in our earlier work and confirm the remarkable versatility and simplicity of central schemes as black-box, Jacobian-free MHD solvers. Furthermore, our numerical experiments demonstrate that this family of semidiscrete central schemes preserves the $\nabla \cdot \mathbf{B} = 0$-constraint within machine round-off error; happily, no constrained-transport enforcement is needed.

Key words. multidimensional conservation laws, ideal magnetohydrodynamics (MHD) equations, high-resolution central schemes, nonoscillatory reconstructions, Jacobian-free form, semi-discrete schemes

AMS subject classifications. Primary 65M10; Secondary 65M05, 76W05

DOI. 10.1137/040610246

1. Introduction.

In this paper we present third- and fourth-order accurate, nonoscillatory semidiscrete central schemes for the approximate solution of the equations of ideal magnetohydrodynamics,

$$\rho_t = -\nabla \cdot (\rho \mathbf{v}), \tag{1.1}$$

$$(\rho \mathbf{v})_t = -\nabla \cdot \left[\rho \mathbf{v}\mathbf{v}^\top + \left(p + \frac{1}{2} B^2 \right) I_{3\times 3} - \mathbf{B}\mathbf{B}^\top \right], \tag{1.2}$$

$$\mathbf{B}_t = \nabla \times (\mathbf{v} \times \mathbf{B}), \tag{1.3}$$

$$e_t = -\nabla \cdot \left[\left(\frac{\gamma}{\gamma - 1} p + \frac{1}{2} \rho v^2 \right) \mathbf{v} - (\mathbf{v} \times \mathbf{B}) \times \mathbf{B} \right]. \tag{1.4}$$

Here, ρ and e are scalar quantities representing, respectively, the mass density and the total internal energy, $\mathbf{v} = (v_x, v_y, v_z)^\top$ is the velocity field with , , norm

*Received by the editors June 20, 2004; accepted for publication (in revised form) October 17, 2005; published electronically April 21, 2006. This research was supported in part by NSF grant 04-07704 and ONR grant N00014-91-J-1076.

http://www.siam.org/journals/sisc/28-2/61024.html.

†Department of Mathematics, University of Michigan, Ann Arbor, MI 48109 (jbalbas@umich.edu). Part of the research was carried out while this author was visiting the Center for Scientific Computation and Mathematical Modeling (CSCAMM) at the University of Maryland, College Park. Research was supported in part by an NSF VIGRE grant.

‡Department of Mathematics, Center of Scientific Computation and Mathematical Modeling (CSCAMM) and Institute for Physical Science and Technology (IPST), University of Maryland, College Park, MD 20742 (tadmor@cscamm.umd.edu).

$v^2 := \|\mathbf{v}\|^2$, and $\mathbf{B} = (B_x, B_y, B_z)^\top$ and $B^2 := \|\mathbf{B}\|^2$ represent the magnetic field and its norm. The pressure, p, is coupled to the internal energy, $e = \frac{1}{2}\rho v^2 + \frac{1}{2}B^2 + p/(\gamma - 1)$, where γ is the (fixed) ratio of specific heats. The system is augmented by the solenoidal constraint,

$$\nabla \cdot \mathbf{B} = 0; \tag{1.5}$$

that is, if the condition $\nabla \cdot \mathbf{B} = 0$ is satisfied initially at $t = 0$, then by (1.3) it remains invariant in time.

Our work [2] demonstrated the capability of central schemes to accurately and efficiently detect and resolve the steep gradients that characterize the solutions of (1.1)–(1.4); it suggested further development of higher-order, black-box-type central schemes for these equations. In this paper, we present a family of third- and fourth-order accurate, nonoscillatory semidiscrete central schemes which remain stable without any further explicit enforcement of the constraint (1.5). This semidiscrete formulation [14] retains the main advantages of central schemes over the upwind ones, namely, simplicity and efficiency resulting from the minimal amount of characteristic information required when evolving the solution over staggered grids, allowing for Jacobian-free formulations and avoiding dimensional splitting in multidimensional models. Moreover, semidiscrete schemes allow more flexibility in the calculation of the time step, Δt, than their fully discrete counterparts: on the one hand, they are implemented over nonstaggered grids, and thus the CFL stability condition allows for the use of larger time steps, and on the other hand, for rth-order methods, the numerical dissipation of the semidiscrete formulation is of order $\mathcal{O}((\Delta x)^{2r-1})$ versus that of the corresponding fully discrete version of order $\mathcal{O}((\Delta x)^{2r}/\Delta t)$, which allows the use of smaller time steps with considerably less smearing of discontinuities. Finally, the semidiscrete formulation also enjoys a remarkable adaptability to a wide range of nonoscillatory reconstruction techniques and evolution routines developed independently of the methods.

In addition to the five magnetohydrodynamics (MHD) prototype problems discussed in [2]—two one-dimensional Brio and Wu shock tube models originally proposed in [3], two different configurations (periodic and convective models) of the two-dimensional transverse Kelvin–Helmholtz instability problem, and the two-dimensional MHD vortex system of Orszag and Tang—we implement these new schemes to simulate the disruption of a high density cloud by a strong shock. The evolution of the Orszag–Tang vortex system involves the interaction between several shock waves traveling at various speed regimes, and the interaction between the high-density cloud and the strong shock describes a highly superfast flow, making both of these very suitable problems to test the robustness of numerical algorithms.

The paper is structured as follows. In section 2, we extend the derivation of the second- and third-order, central semidiscrete schemes presented in [14, 13, 12] to schemes of arbitrary order r. Starting with the staggered discretization that led to the fully discrete central schemes [20, 19, 10, 17], we evolve separately the smooth and nonsmooth parts of the solution and project the resulting cell averages back onto the original grid. We then pass to the $\Delta t \downarrow 0$-limit, yielding the rth-order semidiscrete formulation of central schemes. This rather general formulation of central schemes entertains a broad range of options to implement the two main steps of the algorithm: nonoscillatory [illegible] of point values from cell averages, followed by the [illegible] of the corresponding point valued fluxes. In section 3 we discuss the nonoscillatory reconstruction techniques and evolution routines that we implement to approximate

the semidiscrete formulation of (1.1)–(1.4). For the reconstruction of point values, we propose an extension of the third-order central WENO (CWENO) reconstruction of Kurganov and Levy [12] and the two-dimensional fourth-order CWENO reconstruction of Levy, Puppo, and Russo [18], developed originally in the context of fully discrete central schemes. For the evolution step, we use third- and fourth-order strong stability preserving (SSP) Runge–Kutta schemes [23, 25, 5]. In sections 4 and 5 we describe the prototype MHD problems mentioned above and present the numerical solutions obtained with our new family of high-resolution semidiscrete central schemes. In section 4 we report on the one-dimensional shock tube problems of Brio and Wu [3] and discuss the improved resolution and better control of spurious oscillations provided by the new schemes. The two-dimensional test problems and the corresponding numerical results are presented and discussed in section 5. These results confirm the efficiency of central schemes to accurately compute the solution of MHD equations. The combination of simplicity and high resolution allows us to obtain solutions comparable to those obtained with high-order upwind schemes, e.g., [3, 4, 11, 22, 21], without having to refine the computational mesh as in [2]. We therefore retain the simplicity of central schemes—they eliminate the need for a detailed knowledge of the eigen-structure of the Jacobian matrices and avoid dimensional splitting and the costly use of (approximate) Riemann solvers that serve as building blocks for upwind schemes; at the same time, this simplicity is achieved without an increase in the computational work that was enforced by the finer meshes required with the lower-order methods and/or any explicit enforcement of the solenoidal constraint $\nabla \cdot \mathbf{B} = 0$.

2. Semidiscrete central schemes. In [2] we demonstrated the robustness of black-box-type central schemes to calculate the approximate solutions of (1.1)–(1.4) and discussed the advantages of these methods over those based on upwind differencing. We also pointed out, however, that the efficiency resulting from a minimal amount of characteristic information required by central schemes may be compromised by the low-order accuracy of the schemes, since the latter often require finer meshes to resolve steep gradients. In order to circumvent this limitation and gain full advantage of the simplicity, ease of implementation, and efficiency of central schemes, we seek higher-order methods. Although it is possible to carry out a higher-order extension of the fully discrete central schemes discussed in [2] (see, e.g., [16, 18]), we find the semidiscrete formulation of Kurganov and Tadmor [14] more advantageous: its implementation over nonstaggered grids allows the use of larger time steps, it enjoys a reduced numerical dissipation (of order $\mathcal{O}((\Delta x)^{2r-1})$) in comparison to that of fully discrete schemes ($\sim \mathcal{O}((\Delta x)^{2r}/\Delta t)$), and it is derived independently of any reconstruction or evolution algorithm. Indeed, many of the existing high-order, nonoscillatory reconstruction algorithms and evolution routines can be easily incorporated into the semidiscrete central formulation, making the simplicity and ease of implementation of central schemes an attractive alternative.

2.1. One-dimensional schemes. The solutions of nonlinear hyperbolic systems of conservation laws,

$$u_t + f(u)_x = 0, \tag{2.1}$$

and in particular those of (1.1)–(1.4), are characterized by the onset and propagation of discontinuities. The higher-order extensions of the first-order Lax–Friedrichs central scheme [15] that have been developed over the past two decades (see, e.g., [20, 19,

10, 14]) provide a rather general approach for solving this type of problem with only a minimal amount of information on the eigen structure of the Jacobian matrix of $f(u)$. (Indeed, Jacobian-free formulations of central schemes completely avoid the computation of $\frac{\partial f}{\partial u}$.) These schemes are based on the evolution of the cell averages (we use the usual $\fint_I$ to denote the average, $1/|I| \cdot \int_I$),

$$\bar{u}_\alpha^n := \fint_{I_\alpha} u(\xi, t^n) d\xi, \qquad I_\alpha := \left[x_\alpha - \frac{\Delta x}{2}, x_\alpha + \frac{\Delta x}{2} \right],$$

while alternating over staggered grids, $\{I_j\} \times t^n$ and $\{I_{j+\frac{1}{2}}\} \times t^{n+1}$. Dividing (2.1) by Δx and integrating over the control volume $I_{j+\frac{1}{2}} \times [t^n, t^{n+1}]$ yields an exact discrete statement of the conservation law in terms of these cell averages,

$$\bar{u}_{j+\frac{1}{2}}^{n+1} = \bar{u}_{j+\frac{1}{2}}^n - \frac{1}{\Delta x} \left[\int_{t^n}^{t^{n+1}} f(u(x_{j+1}, \tau)) d\tau - \int_{t^n}^{t^{n+1}} f(u(x_j, \tau)) d\tau \right]. \tag{2.2}$$

The evaluation of the expressions on the right of (2.2) proceeds in two steps. First, the cell averages $\{\bar{u}_{j+\frac{1}{2}}^n\}$ are recovered by integrating a piecewise nonoscillatory polynomial reconstruction of the point-values of $u(x, t^n)$ from their cell averages $\{\bar{u}_j^n\}$ that we denote by

$$u(x, t^n) = R\,(x; \bar{u}^n) := \sum_j p_j^n(x) \cdot \mathbf{1}_{I_j}. \tag{2.3}$$

This reconstruction procedure is at the heart of high-resolution, nonoscillatory central schemes and requires the coefficients of the polynomials $\{p_j^n(x)\}$ to be determined so that $R(x; \bar{u}^n)$ satisfies the following three essential properties:

- $\mathcal{P}_1$—conservation of cell averages: $\bar{p}_j^n(x) = \bar{u}_j^n$.
- $\mathcal{P}_2$—accuracy: $R\,(x; \bar{u}^n) = u(x, t^n) + \mathcal{O}((\Delta x)^r)$ for rth-order accurate scheme, wherever $u(\cdot, t^n)$ is sufficiently smooth.
- $\mathcal{P}_3$—nonoscillatory behavior of $\sum_j p_j(x) \cdot \mathbf{1}_{I_j}$ which is characterized in different ways for different reconstructions. Examples include TVD reconstructions initiated by van Leer [27] and Harten [7], number of extrema diminishing (NED) and shape-preserving properties of the Liu–Tadmor third-order scheme [19], and the higher-order nonoscillatory extensions offered by the essentially nonoscillatory (ENO) reconstructions of Harten et al. [6] or their weighted versions (WENO), e.g., [24, 9], and their implementation within the central differencing framework, e.g., [12, 16, 18].

Equipped with the reconstructed point values $u(\cdot, t^n)$, the second step evaluates the two time integrals of $f(u(\cdot, \tau))$. Since the solution of (2.2) remains smooth along the lines $\{x = x_j\} \times [t^n, t^{n+1}]$, a simple quadrature rule suffices to approximate those integrals. This requires the intermediate values, $\{\bar{u}_j^{n+\beta}\}$, which are predicted either by a Taylor expansion or alternatively by Runge–Kutta solvers of the ODE $u_\tau = \hat{f}_x|_{x=x_j}$, $u(x_j, 0) = u_j^n$, $\tau > t^n$, where $\hat{f}_x$ stands for the numerical derivative of $f(u)$. This type of evolution over staggered grids based on cells with uniform width Δx does not admit a semidiscrete limit as $\Delta t \downarrow 0$. Instead, a modified approach proposed by Kurganov and Tadmor in [14] and based on variable size cells (depicted in Figure 2.1) yields the desired semidiscrete limit. Here, we extend the derivation of second- and third-order semidiscrete schemes of [14, 12, 13] to schemes of arbitrary order of accuracy r [1]. The first step passing from a fully discrete to a semidiscrete formulation consists of

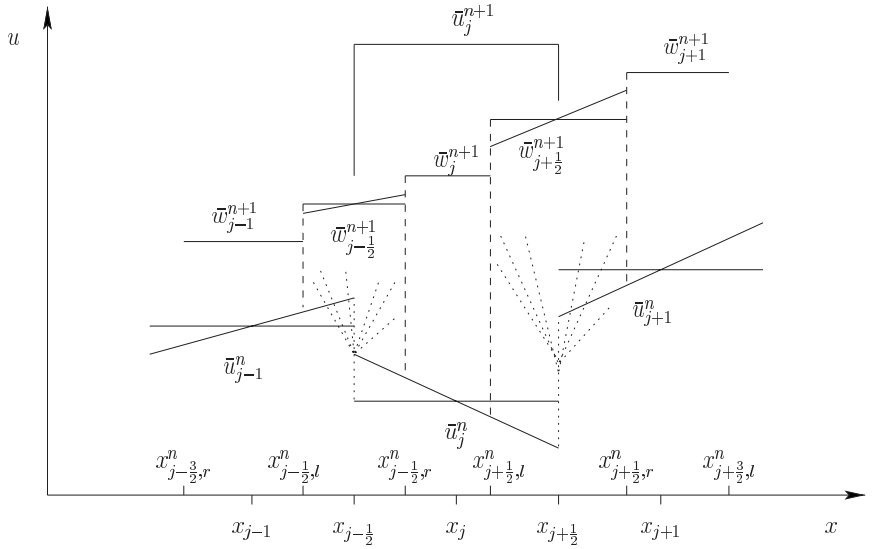


FIG. 2.1. *Modified central differencing of Kurganov and Tadmor.*

reducing the size of the staggered cells covering the Riemann fans. Instead of cells with uniform width Δx, we use cells with variable width of order $\mathcal{O}(\Delta t)$, by incorporating the maximal local speed of propagation,

$$a_{j+\frac{1}{2}}^n = \max_{u\in\mathcal{C}\left(u_{j+\frac{1}{2}}^-,u_{j+\frac{1}{2}}^+\right)} \rho\left(\frac{\partial f}{\partial u}(u)\right); \tag{2.4}$$

here $u_{j+\frac{1}{2}}^-$ and $u_{j+\frac{1}{2}}^+$ stand for, respectively, the reconstructed values of u to the left and right of the interfacing breakpoints, $u_{j+\frac{1}{2}}^{\pm} := R(x_{j+\frac{1}{2}}\pm, \bar{u}^n)$, and $\mathcal{C}$ is the phase curve connecting these two states through the Riemann fan. For practical purposes, we choose

$$a_{j+\frac{1}{2}}^n := \max\left\{\rho\left(\frac{\partial f}{\partial u}(u_{j+\frac{1}{2}}^-)\right), \rho\left(\frac{\partial f}{\partial u}(u_{j+\frac{1}{2}}^+)\right)\right\},$$

which is exact for genuinely nonlinear and linearly degenerate fields.

This information allows us to repartition the original grid so as to distinguish between subcells of width $2a_{j+\frac{1}{2}}^n\Delta t^n$, where the solution is dictated by the nonsmooth Riemann fan,

$$\widetilde{I}_{j+\frac{1}{2}} = \left[x_{j+\frac{1}{2},l}^n,\ x_{j+\frac{1}{2},r}^n\right] := \left[x_{j+\frac{1}{2}} - a_{j+\frac{1}{2}}^n\Delta t^n,\ x_{j+\frac{1}{2}} + a_{j+\frac{1}{2}}^n\Delta t^n\right],$$

and the other cells of width $\Delta x_j := \Delta x - \Delta t^n(a_{j-\frac{1}{2}}^n + a_{j+\frac{1}{2}}^n)$, which are beyond the influence of the Riemann fans so that the solution remains smooth,

$$\widetilde{I}_j = \left[x_{j-\frac{1}{2},r}^n,\ x_{j+\frac{1}{2},l}^n\right] := \left[x_{j-\frac{1}{2}} + a_{j-\frac{1}{2}}^n\Delta t^n,\ x_{j+\frac{1}{2}} - a_{j+\frac{1}{2}}^n\Delta t^n\right].$$

In nonsmooth regions, the cell averages $\{\bar{w}^{n+1}_{j+\frac{1}{2}}\}$ are computed via staggered evolution over the (narrower, $\mathcal{O}(\Delta t)$) cells covering the Riemann fans,

$$\begin{aligned}\bar{w}^{n+1}_{j+\frac{1}{2}} &:= \fint_{x^n_{j+\frac{1}{2},l}}^{x^n_{j+\frac{1}{2},r}} u(x,t^{n+1})dx \\ &= \fint_{x^n_{j+\frac{1}{2},l}}^{x^n_{j+\frac{1}{2},r}} R\,(x;\bar{u}^n)dx - \frac{1}{2a^n_{j+\frac{1}{2}}}\int_{t^n}^{t^{n+1}} \big[f(u(x^n_{j+\frac{1}{2},r},\tau)) - f(u(x^n_{j+\frac{1}{2},l},\tau))\big]d\tau, \end{aligned} \tag{2.5}$$

and in the smooth regions, direct integration of (2.1) yields the cell averages $\{\bar{w}^n_j\}$

$$\begin{aligned}\bar{w}^{n+1}_{j} &:= \fint_{x^n_{j-\frac{1}{2},r}}^{x^n_{j+\frac{1}{2},l}} u(x,t^{n+1})dx \\ &= \fint_{x^n_{j-\frac{1}{2},r}}^{x^n_{j+\frac{1}{2},l}} R\,(x;\bar{u}^n)dx - \frac{1}{\Delta x_j}\int_{t^n}^{t^{n+1}} \big[f(u(x^n_{j+\frac{1}{2},l},\tau)) - f(u(x^n_{j-\frac{1}{2},r},\tau))\big]d\tau. \end{aligned} \tag{2.6}$$

The [illegible] cell averages, $\{\bar{u}^{n+1}_j\}$, are then recovered via a new nonoscillatory piecewise polynomial reconstruction of the point values of $u(x,t^{n+1})$ from the new cell averages (2.5) and (2.6),

$$R\,(x;\bar{w}^{n+1}) = \sum_j q^{n+1}_j(x)\cdot \mathbf{1}_{\widetilde{I}_j} + q^{n+1}_{j+\frac{1}{2}}(x)\cdot \mathbf{1}_{\widetilde{I}_{j+\frac{1}{2}}}. \tag{2.7}$$

Projection onto the original grid, $\{I_j\}$, (2.7) yields

$$\begin{aligned}\bar{u}^{n+1}_j &:= \fint_{x_{j-\frac{1}{2}}}^{x_{j+\frac{1}{2}}} R\,(x;\bar{w}^{n+1})dx \\ &= \frac{1}{\Delta x}\left[\int_{x_{j-\frac{1}{2}}}^{x_{j-\frac{1}{2},r}} q^{n+1}_{j-\frac{1}{2}}(x)\,dx + \int_{x_{j-\frac{1}{2},r}}^{x_{j+\frac{1}{2},l}} q^{n+1}_{j}(x)\,dx + \int_{x_{j+\frac{1}{2},l}}^{x_{j+\frac{1}{2}}} q^{n+1}_{j+\frac{1}{2}}(x)\,dx\right]. \end{aligned} \tag{2.8}$$

The explicit calculation of the polynomials $q^{n+1}_j(x)$ and $q^{n+1}_{j+1/2}(x)$ will, in fact, prove to be irrelevant for the semidiscrete formulation that we seek. As we shall see below, only the [illegible] of these q's matter, and in the $\Delta t \downarrow 0$-limit, these averages are further reduced to the original point values reconstructed at the interfaces, $p(x_{j\pm\frac{1}{2}},t^n)$. We turn to the details. Without loss of generality, we assume that the q^{n+1} are polynomials of degree $r-1$ (required to preserve the underlying rth-order accuracy) and satisfy properties $\mathcal{P}_1$ – $\mathcal{P}_3$ for the cell averages (2.5) and (2.6) in the corresponding cells $\widetilde{I}_j$ and $\widetilde{I}_{j+\frac{1}{2}}$. We express these polynomials as $q^{n+1}_{j\pm\frac{1}{2}}(x) = \sum_{k=0}^{r-1} A^{(k)}_{j\pm\frac{1}{2}}(x - x_{j\pm\frac{1}{2}})^k/k!$; their coefficients can be uniquely determined by imposing the conservation constraints,[1]

$$\fint_{\widetilde{I}_{j+p}} q^{n+1}_{j+s}(x)dx = \bar{w}^n_{j+p}, \qquad p = 0, \pm\frac{1}{2}, \pm 1, \ldots, \pm\frac{r-1}{2}, \qquad s = -\frac{1}{2}, 0, \frac{1}{2}. \tag{2.9}$$

[1]The choice of symmetric intervals around $\tilde{I}_j$ yields methods of odd order. For methods of even order, it suffices to sacrifice one of the side constraints, $p = \pm\frac{r-1}{2}$.

Equipped with this notation, the first and last integrals on the right of (2.8) read

$$\begin{aligned}\int_{x_{j-\frac{1}{2}}}^{x_{j-\frac{1}{2},r}} q_{j-\frac{1}{2}}^{n+1}(x)\,dx &= \sum_{k=0}^{r-1}\left[\frac{A_{j-\frac{1}{2}}^{(k)}}{(k+1)!}(x-x_{j-\frac{1}{2}})^{k+1}\right]_{x_{j-\frac{1}{2}}}^{x_{j-\frac{1}{2}}+a_{j-\frac{1}{2}}^{n}\Delta t}\\ &= \sum_{k=0}^{r-1}\frac{(\Delta t)^{k+1}}{(k+1)!}(a_{j-\frac{1}{2}}^{n})^{k+1}A_{j-\frac{1}{2}}^{(k)}\end{aligned} \tag{2.10}$$

and

$$\begin{aligned}\int_{x_{j+\frac{1}{2},l}}^{x_{j+\frac{1}{2}}} q_{j+\frac{1}{2}}^{n+1}(x)\,dx &= \sum_{k=0}^{r-1}\left[\frac{A_{j+\frac{1}{2}}^{(k)}}{(k+1)!}(x-x_{j+\frac{1}{2}})^{k+1}\right]_{x_{j+\frac{1}{2}}-a_{j+\frac{1}{2}}^{n}\Delta t}^{x_{j+\frac{1}{2}}}\\ &= \sum_{k=0}^{r-1}(-1)^{k}\frac{(\Delta t)^{k+1}}{(k+1)!}(a_{j+\frac{1}{2}}^{n})^{k+1}A_{j+\frac{1}{2}}^{(k)}.\end{aligned} \tag{2.11}$$

For the second integral in (2.8), we observe that by property $\mathcal{P}_1$ the polynomial $q_j^{n+1}(x)$ must satisfy

$$\int_{x_{j-\frac{1}{2},r}}^{x_{j+\frac{1}{2}},l} q_j^{n+1}(x)\,dx = \Delta x_j \bar{w}_j^{n+1} = \left[\Delta x - \Delta t(a_{j-\frac{1}{2}}^{n} + a_{j+\frac{1}{2}}^{n})\right]\bar{w}_j^{n+1}. \tag{2.12}$$

Substituting (2.10)–(2.12) into (2.8), we arrive at a fully discrete, nonstaggered approximation (of arbitrary order of accuracy r) of the cell averages in (2.2),

$$\begin{aligned}\bar{u}_j^{n+1} = &\sum_{k=0}^{r-1}\frac{\lambda(\Delta t)^k}{(k+1)!}\left[(a_{j-\frac{1}{2}}^{n})^{k+1}A_{j-\frac{1}{2}}^{(k)} + (-1)^k(a_{j+\frac{1}{2}}^{n})^{k+1}A_{j+\frac{1}{2}}^{(k)}\right]\\ &+ \left[1-\lambda(a_{j-\frac{1}{2}}^{n} + a_{j+\frac{1}{2}}^{n})\right]\bar{w}_j^{n+1}, \quad \lambda := \frac{\Delta t}{\Delta x}.\end{aligned} \tag{2.13}$$

Given the appropriate reconstructions $R(x;\bar{u}^n)$, $R(x;\bar{w}^n)$ and quadrature rules to evaluate the flux integrals in (2.5) and (2.6), then (2.13) constitutes a family of fully discrete nonoscillatory central schemes, e.g., [8, 14]. This type of scheme admits a considerably simpler and more versatile semidiscrete limit, $d\bar{u}_j/dt := \lim_{\Delta t\to 0}(\bar{u}_j(t^n+\Delta t) - \bar{u}_j(t^n))/\Delta t$.

To this end, we rewrite (2.13) as

$$\begin{aligned}\frac{d\bar{u}_j}{dt} = &\lim_{\Delta t\to 0}\left\{\frac{1}{\Delta x}\left(a_{j-\frac{1}{2}}^{n}A_{j-\frac{1}{2}}^{(0)} + a_{j+\frac{1}{2}}^{n}A_{j+\frac{1}{2}}^{(0)}\right) + \frac{1}{\Delta t}(\bar{w}_j^{n+1} - \bar{u}_j^{n})\right.\\ &- \frac{1}{\Delta x}\left(a_{j-\frac{1}{2}}^{n} + a_{j+\frac{1}{2}}^{n}\right)\bar{w}_j^{n+1} + \sum_{k=1}^{r-1}\frac{\lambda(\Delta t)^{k-1}}{(k+1)!}\left[\left(a_{j-\frac{1}{2}}^{n}\right)^{k+1}A_{j-\frac{1}{2}}^{(k)}\right.\\ &\left.\left.+ (-1)^k\left(a_{j+\frac{1}{2}}^{n}\right)^{k+1}A_{j+\frac{1}{2}}^{(k)}\right]\right\}\\ = &\lim_{\Delta t\to 0}\left\{\frac{1}{\Delta x}\left(a_{j-\frac{1}{2}}^{n}A_{j-\frac{1}{2}}^{(0)} + a_{j+\frac{1}{2}}^{n}A_{j+\frac{1}{2}}^{(0)}\right) + \frac{1}{\Delta t}(\bar{w}_j^{n+1} - \bar{u}_j^{n})\right.\\ &\left.- \frac{1}{\Delta x}\left(a_{j-\frac{1}{2}}^{n} + a_{j+\frac{1}{2}}^{n}\right)\bar{w}_j^{n+1}\right\}.\end{aligned} \tag{2.14}$$

Since the width of the local Riemann fans approaches zero in the $\Delta t \downarrow 0$-limit, i.e., $x^n_{j+\frac{1}{2},l}, x^n_{j+\frac{1}{2},r} \to x_{j+\frac{1}{2}}$, we obtain by conservation of cell averages property $\mathcal{P}_1$, $A^{(0)}_{j\pm\frac{1}{2}} = \bar{w}^{n+1}_{j\pm\frac{1}{2}}$ so that (2.14) yields

$$\begin{aligned}\frac{d\bar{u}_j}{dt} = \lim_{\Delta t\to 0}\Bigg\{ &\frac{1}{\Delta x}\left(a^n_{j-\frac{1}{2}}\bar{w}^{n+1}_{j-\frac{1}{2}} + a^n_{j+\frac{1}{2}}\bar{w}^{n+1}_{j+\frac{1}{2}}\right) + \frac{1}{\Delta t}(\bar{w}^{n+1}_j - \bar{u}^n_j)\\ &- \frac{1}{\Delta x}\left(a^n_{j-\frac{1}{2}} + a^n_{j+\frac{1}{2}}\right)\bar{w}^{n+1}_j\Bigg\}.\end{aligned} \tag{2.15}$$

As $\Delta t \downarrow 0$ we end up with limiting values which depend solely on the reconstructed values $R(x,u)$ on both sides of the interfacing breakpoints, $x_{j\pm\frac{1}{2}}$, given by

$$\begin{aligned}u(x^n_{j+\frac{1}{2},l},t) &\to p^n_j(x_{j+\frac{1}{2}}) =: u^-_{j+\frac{1}{2}}(t),\\ u(x^n_{j+\frac{1}{2},r},t) &\to p^n_{j+1}(x_{j+\frac{1}{2}}) =: u^+_{j+\frac{1}{2}}(t);\end{aligned} \tag{2.16}$$

here $p^n_j(x)$ and $p^n_{j+1}(x)$ are the polynomial reconstructions of $R(x;\bar{u}^n)$ at the original time step t^n introduced in (2.3). The corresponding averages in (2.5) approach $\bar{w}^{n+1}_{j+\frac{1}{2}} \sim (\Delta f)_{j+\frac{1}{2}}/2a^n_{j+\frac{1}{2}}$, where $(\Delta f)_{j+\frac{1}{2}} := f(u^+_{j+\frac{1}{2}}) - f(u^-_{j+\frac{1}{2}})$, while the smoother averages in (2.6) are of order $\bar{w}^{n+1}_j \sim \bar{u}^n_j - \Delta t(\Delta f)_{j+\frac{1}{2}}$. Inserted into (2.15), we end up with a semidiscrete limit which can be written in its final conservative form,

$$\frac{d\bar{u}_j}{dt} = -\frac{H_{j+\frac{1}{2}}(t) - H_{j-\frac{1}{2}}(t)}{\Delta x}; \tag{2.17}$$

here $H_{j\pm\frac{1}{2}}(t)$ are the numerical fluxes given by

$$H_{j+\frac{1}{2}}(t) := \frac{f(u^+_{j+\frac{1}{2}}(t)) + f(u^-_{j+\frac{1}{2}}(t))}{2} - \frac{a_{j+\frac{1}{2}}(t)}{2}\left[u^+_{j+\frac{1}{2}}(t) - u^-_{j+\frac{1}{2}}(t)\right]. \tag{2.18}$$

Once again, we emphasize that among its several advantages, the versatility of this semidiscrete formulation is the independence of the specific reconstruction, (2.3), and evolution algorithms which are utilized to recover the interface values $u^-_{j+\frac{1}{2}}$ and $u^+_{j+\frac{1}{2}}$ and to evolve the cell averages in (2.17). A description of our particular choices for these algorithms is provided in section 3.

2.2. Two-dimensional schemes. Starting with a general hyperbolic conservation law in two space dimensions,

$$u_t + f(u)_x + g(u)_z = 0, \tag{2.19}$$

we proceed as in section 2.1 and consider the sliding averages of u over the cells $I_{j,k} := [x_{j-1/2}, x_{j+1/2}] \times [z_{k-1/2}, z_{k+1/2}]$,

$$\bar{u}^n_{j,k} := \fint_{I_{j,k}} u(x,z,t^n)\,dx\,dz,$$

where Δx and Δz represent the space scales in the x- and the z-direction, respectively.

As in the one-dimensional case, we seek a semidiscrete formulation for the evolution of the cell averages of the conservation law (2.19). We begin by incorporating

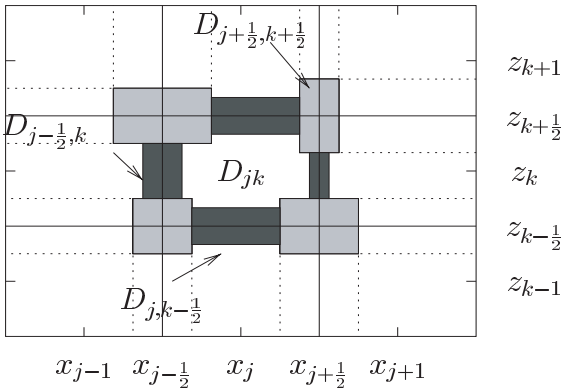


FIG. 2.2. *Floor plan for modified central differencing in two space dimensions.*

the information provided by the local speeds of propagation, which we approximate by

$$
\begin{aligned}
a^n_{j+\frac{1}{2},k} &:= \max\left\{\rho\left(\frac{\partial f}{\partial u}\left(u^W_{j+1,k}\right)\right), \rho\left(\frac{\partial f}{\partial u}\left(u^E_{j,k}\right)\right)\right\}, \quad b^n_{j,k+\frac{1}{2}}, \\
&:= \max\left\{\rho\left(\frac{\partial g}{\partial u}\left(u^S_{j,k+1}\right)\right), \rho\left(\frac{\partial g}{\partial u}\left(u^N_{j,k}\right)\right)\right\},
\end{aligned} \tag{2.20}
$$

where the cell interface values in the z- and x-directions,

$$
\begin{aligned}
u^N_{j,k} &:= p^n_{j,k}(x_j, z_{k+\frac{1}{2}}), \quad u^S_{j,k} := p^n_{j,k}(x_j, z_{k-\frac{1}{2}}), \\
u^E_{j,k} &:= p^n_{j,k}(x_{j+\frac{1}{2}}, z_k), \quad u^W_{j,k} := p^n_{j,k}(x_{j-\frac{1}{2}}, z_k),
\end{aligned} \tag{2.21}
$$

are calculated via a nonoscillatory piecewise polynomial reconstruction,

$$
R(x, z; \bar{u}^n) = \sum_{j,k} p^n_{j,k}(x, z)\mathbf{1}_{I_{j,k}}; \tag{2.22}
$$

the polynomials $\{p^n_{j,k}(x, z)\}$ are determined so that $R(x, z; \bar{u}^n)$ satisfies properties analogous to $\mathcal{P}_1$ – $\mathcal{P}_3$ above. This information allows us to separate between regions of smoothness, depicted as $D_{j,k}$ in Figure 2.2, and regions of nonsmoothness, depicted as the shaded regions. Nine sets of cell averages are calculated as follows:

- In the clear shaded regions, $D_{j\pm\frac{1}{2},k\pm\frac{1}{2}}$, the solution is not smooth in both directions; staggered evolution over the control volumes $D_{j\pm\frac{1}{2},k\pm\frac{1}{2}} \times [t^n, t^{n+1}]$ is used to obtain the new cell averages $\{\bar{w}^{n+1}_{j\pm\frac{1}{2},k\pm\frac{1}{2}}\}$,
- In the dark shaded regions, $D_{j\pm\frac{1}{2},k}$ and $D_{j,k\pm\frac{1}{2}}$, the solution is smooth only in one direction; staggered evolution is used along the nonsmooth interfaces to obtain, respectively, the cell averages $\{\bar{w}^{n+1}_{j\pm\frac{1}{2},k}\}$ and $\{\bar{w}^{n+1}_{j,k\pm\frac{1}{2}}\}$,
- To evolve the smooth part of the solution, we integrate the polynomial $p^n_{j,k}(x, z)$ and the corresponding fluxes over the nonrectangular control volume $D_{j,k} \times [t^n, t^{n+1}]$; these cell averages are denoted by $\{\bar{w}^{n+1}_{j,k}\}$.

This approach allows us to form a new polynomial, denoted $R(x, z; \bar{w}^{n+1})$, which is reconstructed from these smooth and nonsmooth portions of the solution, and to

reproject it back onto the original grid-cells,

$$\bar{u}_{j,k}^{n+1} = \fint_{I_{j,k}} R\,(x,z;\bar{w}^{n+1})\,dx\,dz. \tag{2.23}$$

The resulting nonstaggered fully discrete scheme admits a more versatile semidiscrete limit as $\Delta t \to 0$,

$$\frac{d}{dt}\bar{u}_{j,k}(t) = -\frac{H^x_{j+\frac{1}{2},k}(t) - H^x_{j-\frac{1}{2},k}(t)}{\Delta x} - \frac{H^z_{j,k+\frac{1}{2}}(t) - H^z_{j,k-\frac{1}{2}}(t)}{\Delta z}, \tag{2.24}$$

with numerical fluxes

$$\begin{aligned} H^x_{j+\frac{1}{2},k}(t) = {} & \frac{1}{12}\Big[f\left(u^{NW}_{j+1,k}(t)\right) + f\left(u^{NE}_{j,k}(t)\right) + 4\Big(f\left(u^{W}_{j+1,k}(t)\right) \\ & + f\left(u^{E}_{j,k}(t)\right)\Big) + f\left(u^{SW}_{j+1,k}(t)\right) + f\left(u^{SE}_{j,k}(t)\right)\Big] \\ & - \frac{a_{j+\frac{1}{2},k}(t)}{12}\Big(u^{NW}_{j+1,k}(t) - u^{NE}_{j,k}(t) + 4\Big(u^{W}_{j+1,k}(t) \\ & - u^{E}_{j,k}(t)\Big) + u^{SW}_{j+1,k}(t) - u^{SE}_{j,k}(t)\Big), \end{aligned} \tag{2.25}$$

$$\begin{aligned} H^z_{j,k+\frac{1}{2}}(t) = {} & \frac{1}{12}\Big[g\left(u^{SW}_{j,k+1}(t)\right) + g\left(u^{NW}_{j,k}(t)\right) + 4\Big(g\left(u^{S}_{j,k+1}(t)\right) \\ & + g\left(u^{N}_{j,k}(t)\right)\Big) + g\left(u^{SE}_{j,k+1}(t)\right) + g\left(u^{NE}_{j,k}(t)\right)\Big] \\ & - \frac{b_{j,k+\frac{1}{2}}(t)}{12}\Big(u^{SW}_{j,k+1}(t) - u^{NW}_{j,k}(t) + 4\Big(u^{S}_{j,k+1}(t) \\ & - u^{N}_{j,k}(t)\Big) + u^{SE}_{j,k+1}(t) - u^{NE}_{j,k}(t)\Big). \end{aligned} \tag{2.26}$$

This particular version of the numerical fluxes results from using Simpson's quadrature rule to approximate the integrals of the fluxes f and g along the cell boundaries $[z_{k-\frac{1}{2}}, z_{k+\frac{1}{2}}]$ and $[x_{j-\frac{1}{2}}, x_{j+\frac{1}{2}}]$, respectively, and it incorporates information from the [illegible] interface values,

$$\begin{aligned} u^{NE}_{j,k} &:= \hat{p}^n_{j,k}(x_{j+\frac{1}{2}}, z_{k+\frac{1}{2}}), \quad u^{SW}_{j,k} := \hat{p}^n_{j,k}(x_{j-\frac{1}{2}}, z_{k-\frac{1}{2}}), \\ u^{SE}_{j,k} &:= \hat{p}^n_{j,k}(x_{j+\frac{1}{2}}, z_{k-\frac{1}{2}}), \quad u^{NW}_{j,k} := \hat{p}^n_{j,k}(x_{j-\frac{1}{2}}, z_{k+\frac{1}{2}}), \end{aligned} \tag{2.27}$$

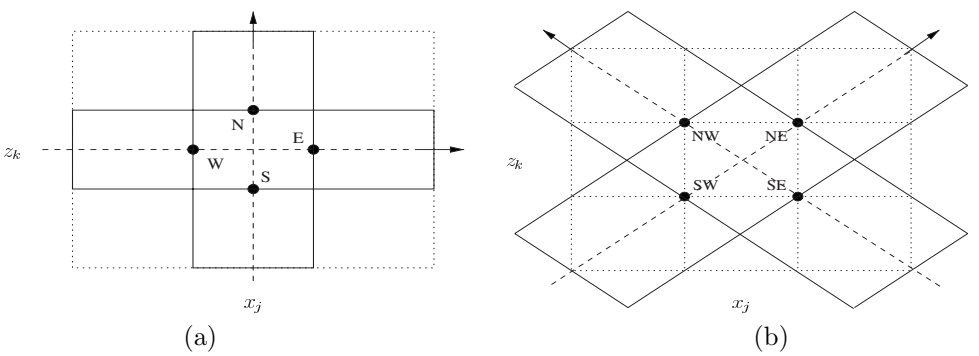


FIG. 2.3. (a) *Reconstruction in x- and z-directions.* (b) *Diagonal axes for reconstruction.*

into the scheme. These corner values are recovered via the nonoscillatory reconstruction $\hat{R}(x,z;\bar{u}^n) = \sum \hat{p}^n_{j,k}(x,z)\cdot \mathbf{1}_{I_{j,k}}$, which may coincide with the original reconstruction, $R(x,z;\bar{u}^n)$, or, alternatively, interpolate the cell values along the diagonal directions shown in Figure 2.3(b) so as to prevent the onset of spurious oscillations along those axes.

3. Implementation of semidiscrete central schemes. The framework of the semidiscrete scheme described above entertains a wide range of options for the implementation of their two main ingredients: nonoscillatory reconstruction and evolution. In this section we provide some examples of nonoscillatory reconstructions and evolution routines that we implemented for computing the solutions of the system (1.1)–(1.4) presented in sections 4 and 5.

3.1. Third-order CWENO reconstruction. Our first choice for the reconstruction of the point values of u is the third-order CWENO polynomial reconstruction of Kurganov and Levy [12]. A piecewise quadratic polynomial that satisfies the essential properties $\mathcal{P}_1$, $\mathcal{P}_2$, and $\mathcal{P}_3$ above is formed as follows: in each cell $I_j = [x_{j-\frac{1}{2}}, x_{j+\frac{1}{2}}]$, the polynomials $\{p^n_j(x)\}$ in (2.3) are written as a convex combination of three polynomials $P_{\rm L}(x)$, $P_{\rm C}(x)$, and $P_{\rm R}(x)$,

$$p^n_j(x) = w_{\rm L}P_{\rm L}(x) + w_{\rm C}P_{\rm C}(x) + w_{\rm R}P_{\rm R}(x), \qquad \sum_{i\in\{{\rm L},{\rm C},{\rm R}\}} w_i = 1, \tag{3.1}$$

where the linear polynomials

$$P_{\rm L}(x) = \bar{u}^n_j + \frac{\bar{u}^n_j - \bar{u}^n_{j-1}}{\Delta x}(x - x_j) \quad \text{and} \quad P_{\rm R}(x) = \bar{u}^n_j + \frac{\bar{u}^n_{j+1} - \bar{u}^n_j}{\Delta x}(x - x_j), \tag{3.2}$$

conserve the pair of cell averages $\bar{u}^n_{j-1}$, $\bar{u}^n_j$ and $\bar{u}^n_j$, $\bar{u}^n_{j+1}$, respectively, and the parabola centered around x_j,

$$\begin{aligned} P_{\rm C}(x) = \bar{u}^n_j &- \frac{1}{12}(\bar{u}^n_{j-1} - 2\bar{u}^n_j + \bar{u}^n_{j+1}) \\ &+ \frac{\bar{u}^n_{j+1} - \bar{u}^n_{j-1}}{2\Delta x}(x - x_j) + \frac{\bar{u}^n_{j-1} - 2\bar{u}^n_j + \bar{u}^n_{j+1}}{\Delta x^2}(x - x_j)^2, \end{aligned} \tag{3.3}$$

is determined so as to satisfy

$$c_{\rm L}P_{\rm L}(x) + c_{\rm R}P_{\rm R}(x) + (1 - c_{\rm L} - c_{\rm R})P_{\rm C}(x) = u^n_j + u'_j(x - x_j) + \frac{1}{2}u''_j(x - x_j)^2,$$

where u^n_j, u'_j, and u''_j approximate the point values $u(x_j,t^n)$, $u_x(x_j,t^n)$, and $u_{xx}(x_j,t^n)$ given respectively by

$$\begin{aligned} u^n_j &:= \bar{u}^n_j - \frac{1}{24}(\bar{u}^n_{j-1} - 2\bar{u}^n_j + \bar{u}^n_{j+1}), \\ u'_j &:= \frac{\bar{u}^n_{j+1} - \bar{u}^n_{j-1}}{2\Delta x}, \quad \text{and} \quad u''_j := \frac{\bar{u}^n_{j-1} - 2\bar{u}^n_j + \bar{u}^n_{j+1}}{\Delta x^2}. \end{aligned} \tag{3.4}$$

We guarantee the conservation of the cell averages $\bar{u}^n_{j-1}$, $\bar{u}^n_j$, and $\bar{u}^n_{j+1}$ by any symmetric choice of the weights c_i (e.g., $c_{\rm L} = c_{\rm R} = 1/4$, $c_{\rm C} = 1/2$) and the accuracy property, $\mathcal{P}_2$, in smooth regions [17],

$$p^n_j(x) = u(x,t^n) + \mathcal{O}((\Delta x)^3), \quad x \in [x_{j-\frac{1}{2}}, x_{j+\frac{1}{2}}].$$

Property $\mathcal{P}_3$—the nonoscillatory behavior of $\{p_j^n(x)\}$ is attained with the nonlinear weights

$$w_i = \frac{\alpha_i}{\sum_m \alpha_m} \quad \text{with} \quad \alpha_i = \frac{c_i}{(\epsilon + IS_i)^2}, \quad i, m \in \{\text{L}, \text{C}, \text{R}\}, \tag{3.5}$$

where $\epsilon \ll 1$ prevents the denominator from vanishing (for the calculations in sections 4 and 5 we choose $\epsilon = 10^{-6}$), and the smoothness indicators, IS_i, provide a local measure of the derivatives of $P_i(x)$,

$$IS_i = \sum_{l=1}^{2} \int_{I_j} (\Delta x)^{2l-1} (P_i^{(l)}(x))^2 dx, \quad i \in \{\text{L}, \text{C}, \text{R}\}, \tag{3.6}$$

switching automatically to the second-order reconstructions P_{L} and P_{R} in the presence of steep gradients and avoiding the onset of spurious oscillations [9, 12, 17]. In this case, they read

$$\begin{aligned} IS_{\text{L}} &= (\bar{u}_j^n - \bar{u}_{j-1}^n)^2, \qquad IS_{\text{R}} = (\bar{u}_{j+1}^n - \bar{u}_j^n)^2, \\ IS_{\text{C}} &= \frac{13}{3}(\bar{u}_{j+1}^n - 2\bar{u}_j^n + \bar{u}_{j-1}^n)^2 + \frac{1}{4}(\bar{u}_{j+1}^n - \bar{u}_{j-1}^n)^2. \end{aligned} \tag{3.7}$$

1. For this particular reconstruction, the one-dimensional interface values, $u^-_{j+\frac{1}{2}}(t)$ and $u^+_{j+\frac{1}{2}}(t)$, required in the numerical flux (2.18) take the explicit form

$$\begin{aligned} u^-_{j+\frac{1}{2}}(t^n) := p_j^n(x_{j+\frac{1}{2}}) &= \frac{1}{2} w_{\text{L}}(3\bar{u}_j^n - \bar{u}_{j-1}^n) \\ &\quad + \frac{1}{12} w_{\text{C}}(-5\bar{u}_{j+1}^n + 8\bar{u}_j^n - \bar{u}_{j-1}^n) + \frac{1}{2} w_{\text{R}}(\bar{u}_{j+1}^n + \bar{u}_j^n), \\ u^+_{j+\frac{1}{2}}(t^n) := p_{j+1}^n(x_{j+\frac{1}{2}}) &= \frac{1}{2} w_{\text{L}}(\bar{u}_{j+1}^n + \bar{u}_j^n) \\ &\quad + \frac{1}{12} w_{\text{C}}(-\bar{u}_{j+2}^n + 8\bar{u}_{j+1}^n - 5\bar{u}_j^n) + \frac{1}{2} w_{\text{R}}(-\bar{u}_{j+2}^n + 3\bar{u}_{j+1}^n). \end{aligned}$$

2. In the case of systems of equations, the choice of these smoothness indicators is, indeed, nontrivial. The different conserved quantities involved in any particular system (density, momentum, etc.) may very well develop discontinuities at different points throughout the solution domain that do not affect every other quantity in the system. Hence, choosing the smoothness indicators individually for each quantity may cause the scheme to use different stencils for different quantities during the same evolution step. This can be avoided by either using [illegible] smoothness indicators, e.g., an (norm-scaled) average of the individual ones, or by identifying those that are physically relevant, such as, e.g., the density near a contact discontinuity. (See section 4 for our particular choices in the one-dimensional case and [16] for an in-depth discussion about the selection of smoothness indicators for systems of equations.)

In two and higher space dimensions, a possibility for calculating the interface values in (2.21) and (2.27) in a rather straightforward way is to apply this one-dimensional third-order reconstruction [illegible] in the x- and z-directions (Figure 2.3(a)) and, if so desired, in the two diagonal directions of the coordinate frame displayed in Figure 2.3(b). In two dimensions, however, we note that the constant

term in the x-direction parabola, (3.3), requires an additional correction (in the transverse direction of the reconstruction) in order to guarantee third-order accuracy when the point value $u(x_j, z_k, t^n)$ is recovered from the neighboring cell averages,

$$u(x_j, z_k, t^n) = \bar{u}^n_{j,k} - \frac{1}{12}(\bar{u}^n_{j-1,k} - 2\bar{u}^n_{j,k} + \bar{u}^n_{j+1,k}) - \frac{1}{12}(\bar{u}^n_{j,k-1} - 2\bar{u}^n_{j,k} + \bar{u}^n_{j,k+1}) + \mathcal{O}((\max\{\Delta x, \Delta z\})^3).$$

That is, along the line $z = z_k$, the reconstruction in the x-direction is carried out using the linear polynomials $P_{\mathrm{L}}(x, z_k)$ and $P_{\mathrm{R}}(x, z_k)$ as in (3.2), and the parabola $P_{\mathrm{C}}(x, z_k)$, given by

$$P_{\mathrm{C}}(x, z_k) = \bar{u}^n_{j,k} - \frac{1}{12}(\bar{u}^n_{j-1,k} - 2\bar{u}^n_{j,k} + \bar{u}^n_{j+1,k}) - \frac{1}{12}(\bar{u}^n_{j,k-1} - 2\bar{u}^n_{j,k} + \bar{u}^n_{j,k+1}) + \frac{\bar{u}^n_{j+1,k} - \bar{u}^n_{j-1,k}}{2\Delta x}(x - x_j) + \frac{\bar{u}^n_{j-1,k} - 2\bar{u}^n_{j} + \bar{u}^n_{j+1,k}}{\Delta x^2}(x - x_j)^2. \tag{3.8}$$

The same reconstruction is carried out in the z-direction, holding $x = x_j$ fixed. And, should one choose to implement the reconstruction along the diagonal directions, the corresponding polynomials are given by (consult [13] for additional details)

$$\hat{p}^n_{j,k}(x, z) = \hat{w}_{\mathrm{L}}\hat{P}_{\mathrm{L}}(x, z) + \hat{w}_{\mathrm{C}}\hat{P}_{\mathrm{C}}(x, z) + \hat{w}_{\mathrm{R}}\hat{P}_{\mathrm{R}}(x, z) \tag{3.9}$$

with

$$\hat{P}_{\mathrm{L}}(x, z) = \bar{u}^n_{j,k} + \frac{\bar{u}^n_{j,k} - \bar{u}^n_{j-1,k-1}}{\Delta}\left(\frac{\Delta}{2\Delta z}(z - z_k) + \frac{\Delta}{2\Delta x}(x - x_j)\right),$$

$$\begin{aligned}\hat{P}_{\mathrm{C}}(x, z) = \bar{u}^n_{j,k} &- \frac{1}{12}(\bar{u}^n_{j+1,k+1} - 2\bar{u}^n_{j,k} + \bar{u}^n_{j-1,k-1}) \\ &- \frac{1}{12}(\bar{u}^n_{j-1,k+1} - 2\bar{u}^n_{j,k} + \bar{u}^n_{j+1,k-1}) \\ &+ \frac{\bar{u}^n_{j+1,k+1} - \bar{u}^n_{j-1,k-1}}{2\Delta}\left(\frac{\Delta}{2\Delta z}(z - z_k) + \frac{\Delta}{2\Delta x}(x - x_j)\right) \\ &+ \frac{\bar{u}^n_{j+1,k+1} - 2\bar{u}^n_{j,k} + \bar{u}^n_{j-1,k-1}}{\Delta^2}\left(\frac{\Delta}{2\Delta z}(z - z_k) + \frac{\Delta}{2\Delta x}(x - x_j)\right)^2,\end{aligned}$$

$$\hat{P}_{\mathrm{R}}(x, z) = \bar{u}^n_{j,k} + \frac{\bar{u}^n_{j+1,k+1} - \bar{u}^n_{j,k}}{\Delta}\left(\frac{\Delta}{2\Delta z}(z - z_k) + \frac{\Delta}{2\Delta x}(x - x_j)\right) \tag{3.10}$$

along the SW–NE axis and

$$\hat{P}_{\mathrm{L}}(x, z) = \bar{u}^n_{j,k} + \frac{\bar{u}^n_{j,k} - \bar{u}^n_{j+1,k-1}}{\Delta}\left(\frac{\Delta}{2\Delta z}(z - z_k) - \frac{\Delta}{2\Delta x}(x - x_j)\right),$$

$$\begin{aligned}\hat{P}_{\mathrm{C}}(x, z) = \bar{u}^n_{j,k} &- \frac{1}{12}(\bar{u}^n_{j+1,k+1} - 2\bar{u}^n_{j,k} + \bar{u}^n_{j-1,k-1}) \\ &- \frac{1}{12}(\bar{u}^n_{j-1,k+1} - 2\bar{u}^n_{j,k} + \bar{u}^n_{j+1,k-1}) \\ &+ \frac{\bar{u}^n_{j-1,k+1} - \bar{u}^n_{j+1,k-1}}{2\Delta}\left(\frac{\Delta}{2\Delta z}(z - z_k) - \frac{\Delta}{2\Delta x}(x - x_j)\right) \\ &+ \frac{\bar{u}^n_{j-1,k+1} - 2\bar{u}^n_{j,k} + \bar{u}^n_{j+1,k-1}}{\Delta^2}\left(\frac{\Delta}{2\Delta z}(z - z_k) - \frac{\Delta}{2\Delta x}(x - x_j)\right)^2,\end{aligned}$$

$$\hat{P}_{\mathrm{R}}(x, z) = \bar{u}^n_{j,k} + \frac{\bar{u}^n_{j+1,k+1} - \bar{u}^n_{j,k}}{\Delta}\left(\frac{\Delta}{2\Delta z}(z - z_k) - \frac{\Delta}{2\Delta x}(x - x_j)\right) \tag{3.11}$$

along the SE–NW axis, where $\Delta = \sqrt{(\Delta x)^2 + (\Delta z)^2}$. The nonlinear weights, $\hat{w}_i$, are also calculated direction by direction according to the one-dimensional recipe (3.5) with the smoothness indicators given by the [illegible] average of the componentwise indicators, $IS_i^{(m)}$, calculated as in (3.7),

$$IS_i = \frac{1}{d}\sum_{m=1}^{d} \frac{1}{\|u^{(m)}\|_2 + \epsilon}\, IS_i^{(m)}, \quad i \in \{\text{L}, \text{C}, \text{R}\}, \tag{3.12}$$

where $u^{(m)}$ stands for the mth component of u, and $\|u^{(m)}\|_2^2 = \sum_{j,k} |u_{j,k}^{(m)}|^2 \Delta x \Delta z$ represents its ℓ_2 norm over the discretized solution domain.

3.2. CWENO fourth-order reconstruction. In section 5 we demonstrate the versatility of the central semidiscrete scheme (2.24)–(2.26) by implementing the dimension-by-dimension third-order reconstruction described above and the [illegible] fourth-order CWENO reconstruction of Levy, Puppo, and Russo [18] and Balbás [1] that recovers the interface point values in (2.21) and (2.27) via a biquadratic polynomial interpolant satisfying the desired conservation, accuracy, and nonoscillatory properties in two-space dimensions. Following is an outline of this fourth-order reconstruction, originally developed by Levy and others within the central, fully discrete, staggered framework. In each cell $I_{j,k}$, the piecewise polynomial interpolants in (2.22), $p_{j,k}^n(x,z)$, are written as a convex combination of nine biquadratic polynomials, $P_{j+p,k+q}(x,z)$, that conserve the cell averages $\{\bar{u}_{j+p,k+q}^n\}_{p,q=-1}^{1}$ and approximate the point values of $u(x,z,t^n)$ within fourth-order accuracy,

$$p_{j,k}^n(x,z) = \sum_{p,q=-1}^{1} w_{j,k}^{p,q} P_{j+p,k+q}(x,z), \qquad \sum_{p,q=-1}^{1} w_{j,k}^{p,q} = 1, \quad w_{j,k}^{p,q} \geq 0, \tag{3.13}$$

where the polynomials $P_{j,k}(x,z)$ have the form (omitting, for simplicity, the j and k indices in the coefficients $\{b_m\}$),

$$\begin{aligned} P_{j,k}^n(x,z) = {} & b_0 + b_1(x-x_j) + b_2(z-z_k) + b_3(x-x_j)(z-z_k) \\ & + b_4(x-x_j)^2 + b_5(z-z_k)^2 + b_6(x-x_j)^2(z-z_k) \\ & + b_7(x-x_j)(z-z_k)^2 + b_8(x-x_j)^2(z-z_k)^2. \end{aligned} \tag{3.14}$$

The nine conservation constraints,

$$\fint_{x_{j-\frac{1}{2}}+p\Delta x}^{x_{j+\frac{1}{2}}+p\Delta x} \fint_{z_{k-\frac{1}{2}}+q\Delta z}^{z_{k+\frac{1}{2}}+q\Delta z} P_{j,k}(x,z)dz\, dx = \bar{u}_{j+p,k+q}^n, \quad p,q = -1,0,1, \tag{3.15}$$

uniquely determine the nine coefficients $\{b_m\}$,

$$\begin{aligned} b_0 &= \bar{u}^n - \frac{\Delta x^2}{24}\hat{u}_{xx} - \frac{\Delta z^2}{24}\hat{u}_{zz} + \frac{\Delta x^2 \Delta z^2}{24^2}\hat{u}_{xxzz}, \\ b_1 &= \hat{u}_x - \frac{\Delta z^2}{24}\hat{u}_{xzz}, & b_2 &= \hat{u}_z - \frac{\Delta x^2}{24}\hat{u}_{xxz}, \\ b_3 &= \hat{u}_{xz}, & b_4 &= \frac{1}{2}\hat{u}_{xx} - \frac{\Delta z^2}{48}\hat{u}_{xxzz}, \\ b_5 &= \frac{1}{2}\hat{u}_{zz} - \frac{\Delta x^2}{48}\hat{u}_{xxzz}, & b_6 &= \frac{1}{2}\hat{u}_{xxz}, \\ b_7 &= \frac{1}{2}\hat{u}_{xzz}, & b_8 &= \frac{1}{4}\hat{u}_{xxzz}, \end{aligned}$$

where the divided differences

$$\hat{u}_{x_{j,k}} = \frac{\bar{u}^n_{j+1,k} - \bar{u}^n_{j-1,k}}{2\Delta x}, \qquad \hat{u}_{z_{j,k}} = \frac{\bar{u}^n_{j,k+1} - \bar{u}^n_{j,k-1}}{2\Delta z},$$

$$\hat{u}_{xx_{j,k}} = \frac{\bar{u}^n_{j+1,k} - 2\bar{u}^n_{j,k} + \bar{u}^n_{j-1,k}}{\Delta x^2}, \qquad \hat{u}_{zz_{j,k}} = \frac{\bar{u}^n_{j,k+1} - 2\bar{u}^n_{j,k} + \bar{u}^n_{j,k-1}}{\Delta z^2},$$

$$\hat{u}_{xz_{j,k}} = \frac{\bar{u}^n_{j+1,k+1} - \bar{u}^n_{j+1,k-1} - \bar{u}^n_{j-1,k+1} + \bar{u}^n_{j-1,k-1}}{4\Delta x \Delta z},$$

$$\hat{u}_{xxz_{j,k}} = \frac{(\bar{u}^n_{j+1,k+1} - 2\bar{u}^n_{j,k+1} + \bar{u}^n_{j-1,k+1}) - (\bar{u}^n_{j+1,k-1} - 2\bar{u}^n_{j,k-1} + \bar{u}^n_{j-1,k-1})}{2\Delta x^2 \Delta z},$$

$$\hat{u}_{xzz_{j,k}} = \frac{(\bar{u}^n_{j+1,k+1} - 2\bar{u}^n_{j+1,k} + \bar{u}^n_{j+1,k-1}) - (\bar{u}^n_{j-1,k+1} - 2\bar{u}^n_{j-1,k} + \bar{u}^n_{j-1,k-1})}{2\Delta x \Delta z^2},$$

$$\hat{u}_{xxzz_{j,k}} = \frac{1}{\Delta x^2 \Delta z^2}\big[(\bar{u}^n_{j+1,k+1} - 2\bar{u}^n_{j+1,k} + \bar{u}^n_{j+1,k-1}) - 2(\bar{u}^n_{j,k+1} - 2\bar{u}^n_{j,k} + \bar{u}^n_{j,k-1}) + (\bar{u}^n_{j-1,k+1} - 2\bar{u}^n_{j-1,k} + \bar{u}^n_{j-1,k-1})\big]$$

serve to approximate the values of u and its partial derivatives at the points (x_j, z_k) within the fourth-order accuracy constraints of the method.

As in the third-order reconstruction, the nonlinear weights in (3.13), $w^{p,q}_{l,k}$, are computed so as to provide maximum accuracy in smooth regions and prevent oscillations in the nonsmooth regions by eliminating the contribution of polynomials with steep gradients across the cell interfaces. For each cell nine weights are calculated,

$$w^{p,q}_{j,k} = \frac{\alpha^{p,q}_{j,k}}{\sum_{m,n} \alpha^{m,n}_{j,k}} \quad \text{with} \quad \alpha^{p,q}_{j,k} = \frac{c^{p,q}_{j,k}}{(\epsilon + IS^{p,q}_{j,k})^2}, \quad m,n,p,q = -1,0,1, \tag{3.16}$$

and the linear coefficients, $c^{p,q}_{j,k}$, chosen so that symmetry guarantees fourth-order accuracy,

$$c^{0,0}_{j,k} = \frac{1}{2} \qquad \text{and} \qquad c^{p,q}_{j,k} = \frac{1}{16} \quad \forall\ p,q \neq 0;$$

convexity, $\sum^{1}_{p,q=-1} c^{p,q}_{j,k} = 1$, $c^{p,q}_{j,k} \geq 0$, guarantees conservation.

Global smoothness indicators are chosen according to (3.12). Unlike the third-order reconstruction where the exact calculation of the individual smoothness indicators is trivial and renders simple formulas, i.e., (3.7), in this fourth-order case, the exact calculation of the local L^2-norm of the partial derivatives of the polynomials $P^{p,q}_{j,k}$ is impractical. Instead, a Gaussian quadrature formula is used to approximate the integrals

$$IS^{p,q}_{j,k} = \int_{I_{j,k}} (|\partial_x P_{j+p,k+q}|^2 + |\partial_z P_{j+p,k+q}|^2 + (\Delta x)^2 |\partial_{xx} P_{j+p,k+q}|^2 + (\Delta z)^2 |\partial_{zz} P_{j+p,k+q}|^2)\, dx\, dz.$$

This approach still serves the purpose of automatically detecting and redirecting the numerical stencils in the direction of smoothness, thus preventing the onset of oscillations, [18].

3.3. Time evolution: Strong stability-preserving Runge–Kutta solvers. With the interface values (2.16), (2.21), and (2.27) computed with either of the above

CWENO reconstructions, we denote, for any grid function $w = \{w_j(t)\}$, the numerical fluxes on the right of (2.17) and (2.24) by

$$C[w(t)] = -\frac{H_{j+\frac{1}{2}}(w(t)) - H_{j-\frac{1}{2}}(w(t))}{\Delta x} \tag{3.17}$$

and

$$C[w(t)] = -\frac{H^x_{j+\frac{1}{2},k}(w(t)) - H^x_{j-\frac{1}{2},k}(w(t))}{\Delta x} - \frac{H^z_{j,k+\frac{1}{2}}(w(t)) - H^z_{j,k-\frac{1}{2}}(w(t))}{\Delta z}, \tag{3.18}$$

respectively, and evolve the solution $u^{(0)} := \bar{u}^n_j$ from t^n, to t^{n+1} with an appropriate ODE solver. For the numerical calculations reported in sections 4 and 5, we choose the strong stability-preserving (SSP) Runge–Kutta discretizations [23, 5]. In particular, for the third-order results, we employ

$$\begin{aligned} u^{(1)} &= u^{(0)} + \Delta t C[u^{(0)}], \\ u^{(2)} &= u^{(1)} + \frac{\Delta t}{4}(-3C[u^{(0)}] + C[u^{(1)}]), \\ u^{n+1} := u^{(3)} &= u^{(2)} + \frac{\Delta t}{12}(-C[u^{(0)}] - C[u^{(1)}] + 8\ C[u^{(2)}]), \end{aligned} \tag{3.19}$$

and for the fourth-order results of section 5.3,

$$\begin{aligned} u^{(1)} &= u^{(0)} + \frac{\Delta t}{2} C[u^{(0)}], \\ u^{(2)} &= u^{(1)} + \frac{\Delta t}{2}(-C[u^{(0)}] + C[u^{(1)}]), \\ u^{(3)} &= u^{(2)} + \frac{\Delta t}{2}(-C[u^{(1)}] + 2\ C[u^{(2)}]), \\ u^{n+1} := u^{(4)} &= u^{(3)} + \frac{\Delta t}{6}(C[u^{(0)}] + 2C[u^{(1)}] - 4C[u^{(2)}] + C[u^{(3)}]) \end{aligned} \tag{3.20}$$

is used to evolve the values reconstructed via the fourth-order CWENO interpolation algorithm of Levy and others of section 3.2.

4. One-dimensional numerical results. In one space dimension, equations (1.1)–(1.4) admit the conservative form (2.1) with

$$u = (\rho, \rho v_x, \rho v_y, \rho v_z, B_y, B_z, e)^\top, \tag{4.1}$$

$$\begin{aligned} f(u) = (&\rho v_x, \rho v_x^2 + p^* - B_x^2, \rho v_x v_y - B_x B_y, \rho v_x v_z - B_x B_z, B_y v_x - B_x v_y, \\ &B_z v_x - B_x v_z, (e + p^*) v_x - B_x (B_x v_x + B_y v_y + B_z v_z))^\top, \end{aligned} \tag{4.2}$$

where $p^* = p + \frac{1}{2}B^2$ stands for the total pressure (static plus magnetic).

In this section we present numerical simulations of the one-dimensional MHD equations, (2.1)–(4.2). The results were obtained using the semidiscrete central scheme (2.17)–(2.18) with the pointvalues $u^+_{j+\frac{1}{2}}(t)$ and $u^-_{j+\frac{1}{2}}(t)$, (2.16), computed via the CWENO third-order reconstruction of section 3.1, and evolved according to Shu's third-order SSP Runge–Kutta scheme as outlined in section 3.3. The schemes are implemented for computing the approximate solution of two coplanar shock tube MHD models described by Brio and Wu in [3]. We use a uniform grid in the space

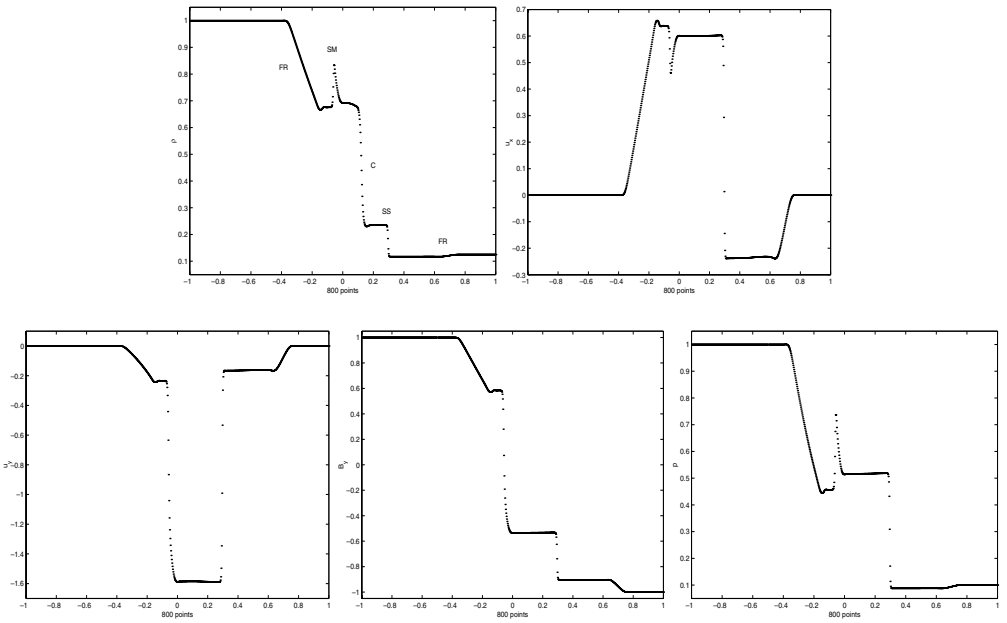


FIG. 4.1. *Results of Brio–Wu shock tube problem at* $t = 0.2$ *computed with* 800 *grid points using third-order CWENO reconstruction,* (3.1)*, and Shu's SSP Runge–Kutta solver,* (3.19).

discretization, and in both cases we choose the time step dynamically with CFL restriction

$$\Delta t = \frac{0.9\ \Delta x}{\max_k |a_k(u)|}, \tag{4.3}$$

where $\{a_k(u)\}_k$ are the eigenvalues of the Jacobian matrix of $f(u)$.

4.1. Brio–Wu shock tube problem. The first one-dimensional Riemann problem we consider consists of a shock tube with two initial equilibrium states, u_l and u_r, given by

$$(\rho, v_x, v_y, v_z, B_y, B_z, p)^\top = \begin{cases} (1.0, 0, 0, 0, 1.0, 0, 1.0)^\top & \text{for } x < 0, \\ (0.125, 0, 0, 0, -1.0, 0, 0.1)^\top & \text{for } x > 0, \end{cases} \tag{4.4}$$

and complemented with the constant values of $B_x \equiv 0.75$ and $\gamma = 2$. The problem is solved for $x \in [-1, 1]$ with 800 grid points, and numerical results are presented at $t = 0.2$. Figure 4.1 shows the density, the x- and y-velocity components, the y-magnetic field, and pressure profiles. The hydrodynamical data of this problem is the same as that in Sod's shock tube problem of gas dynamics. The variety of MHD waves, however, poses a considerable challenge for high resolution such as the [illegible] central schemes described in this paper. The solution of this problem consists of a left-moving fast rarefaction wave (FR), a slow compound wave (SM) which results from an intermediate shock that changes B_y from 0.58 to -0.31 and a slow rarefaction that changes B_y from -0.31 -0.53, a contact discontinuity (C), a right-moving slow shock (SS), and a right-moving fast rarefaction wave (FR). Note that the solution to this problem is not unique if B_z and v_z are not identically zero. The results in Figure 4.1 are in agreement with those previously reported in [2] and are comparable with

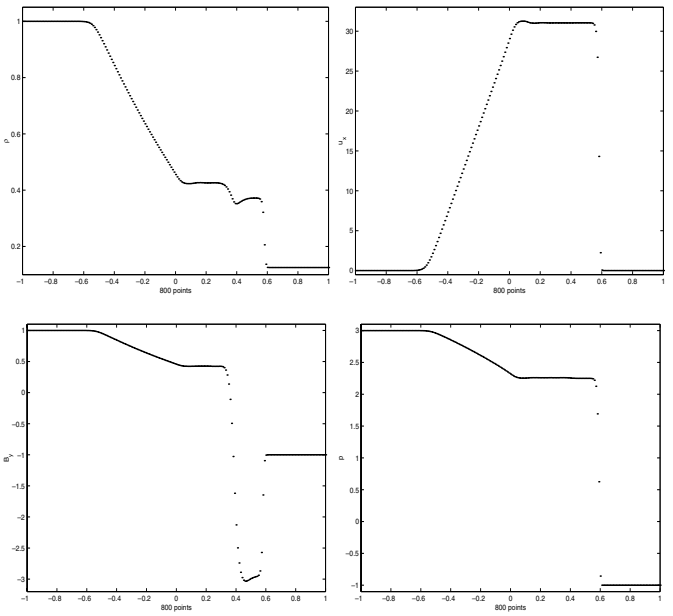


FIG. 4.2. *Results of Brio–Wu high Mach problem at* $t = 0.012$ *computed with* 200 *grid points using third-order CWENO reconstruction,* (3.1)*, and Shu's SSP Runge–Kutta solver,* (3.19).

the second-order upwind computations of Brio and Wu in [3] and with the fifth-order WENO computations presented by Jiang and Wu in [11]. In fact, the present results show a better control of the oscillations that typically appear at the trailing edge of the right-moving fast rarefaction wave when high-resolution schemes are employed. This better control is due to our choice of smoothness indicators. Our various numerical experiments show that in the absence of characteristic information, the average of the smoothness indicators of the density, ρ, and transverse magnetic field, B_y, scaled by their ℓ_2 norm, is the best combination for selecting a single stencil for reconstruction and evolution of all the conserved quantities in the system. Our numerical results confirm the ability of central schemes, whether in their fully discrete or semidiscrete formulations, to capture the main features of the discontinuous MHD solutions, while avoiding any characteristic information other than an estimate of the maximal speed of propagation, $\max_k |a_k(u)|$.

4.2. Brio–Wu high Mach shock tube problem. The following shock tube model proposed by Brio and Wu in [3] is commonly used to check the robustness of the numerical schemes for high Mach number problems. The initial equilibrium states, u_l and u_r, are given by

$$(\rho, v_x, v_y, v_z, B_y, B_z, p)^\top = \begin{cases} (1.0, 0, 0, 0, 1.0, 0, 1000)^\top & \text{for } x < 0, \\ (0.125, 0, 0, 0, -1.0, 0, 0.1)^\top & \text{for } x > 0 \end{cases} \tag{4.5}$$

and complemented with the values of $B_x \equiv 0$ and $\gamma = 2$. The Mach number of the right-moving shock wave is 15.5. If the plasma pressure is replaced by the sum of the static and magnetic pressures, denoted by p^* above, this model becomes a standard hydrodynamical Riemann problem. The solution is presented at $t = .012$, $x \in [-1, 1]$, with 200 grid points and with CFL number 0.9; consult (4.3).

The solution of this second Riemann problem consists of a left-moving fast rarefaction wave (FR), followed by a tangential discontinuity (TD), and a right-moving fast shock (FS) with Mach number 15.5. Across the tangential discontinuity, the density, the magnetic field, and the pressure can change, but both the fluid velocity and the total pressure, $p + \frac{B^2}{2}$, are continuous.

As in the previous problem, our results in Figure 4.2 are comparable to those reported in [2], [3], and [11] with second-order central and second- and fifth-order upwind schemes and demonstrate the robustness of the schemes described in section 2.

5. Two-dimensional numerical results. In two space dimensions, equations (1.1)–(1.4) admit the conservative form (2.19) with

$$u = (\rho, \rho v_x, \rho v_y, \rho v_z, B_x, B_y, B_z, e)^\top, \tag{5.1}$$

$$f(u) = (\rho v_x, \rho v_x^2 + p^* - B_x^2, \rho v_x v_y - B_x B_y, \rho v_x v_z - B_x B_z, 0, B_y v_x - B_x v_y, B_z v_x - B_x v_z, (e + p^*)v_x - B_x(B_x v_x + B_y v_y + B_z v_z))^\top, \tag{5.2}$$

$$g(u) = (\rho v_z, \rho v_z v_x - B_z B_x, \rho v_z v_y - B_z B_y, \rho v_z^2 + p^* - B_z^2, B_x v_z - B_z v_x, B_y v_z - B_z v_y, 0, (e + p^*)v_z - B_z(B_x v_x + B_y v_y + B_z v_z))^\top. \tag{5.3}$$

In this section we present the solution of four prototype problems for two-dimensional MHD equations. We begin testing our two-dimensional numerical schemes by solving the one-dimensional Brio–Wu shock tube problem described in section 4.1 on a two-dimensional domain with the flow rotated 45°. For the second problem—the Kelvin–Helmholtz instability with transverse magnetic field configuration—we consider two different sets of boundary conditions in the x-direction: periodic in the first case and a free outflow boundary in the second convective setup. The third problem was introduced by Orszag and Tang in [21] as a simple model to study MHD turbulence. The last problem that we present simulates the interaction between a strong shock and a high-density cloud. These four problems appear extensively in the literature as they have become—in the absence of a rigorous stability condition for numerical schemes for MHD equations—standard tests to validate numerical algorithms. In all cases, the time scale Δt is determined dynamically according to the CFL restriction

$$\Delta t = \frac{c}{\sqrt{(\max_k |a_k(u)|/\Delta x)^2 + (\max_k |b_k(u)|/\Delta z)^2}}, \tag{5.4}$$

where $c < 1$ is a positive constant and $\{a_k(u)\}_k$ and $\{b_k(u)\}_k$ represent the eigenvalues of the Jacobian matrices of $f(u)$ and $g(u)$, respectively.

It should be emphasized that the results presented below were obtained with the original formulation of semidiscrete central schemes in section 2, namely, they were implemented "as is," without any enforcement of the solenoidal constraint (1.5). The $\nabla \cdot \mathbf{B} = 0$-constraint was [illegible] preserved to machine round-off error. In fact, the same test problems displayed below were first solved using the (global) projection method as a way to enforce the $\nabla \cdot \mathbf{B} = 0$-constraint. We observed that the divergence-free results displayed below avoid undesirable effects such as loss of conservation and monotonicity, often induced by global projection solvers.

5.1. Brio–Wu rotated flow. The initial conditions described by (4.4) are extended to the two-dimensional domain $[-1, 1] \times [-1, 1]$ and rotated 45° so that the fluid does not flow in the direction of the grid axis. Our results are displayed in

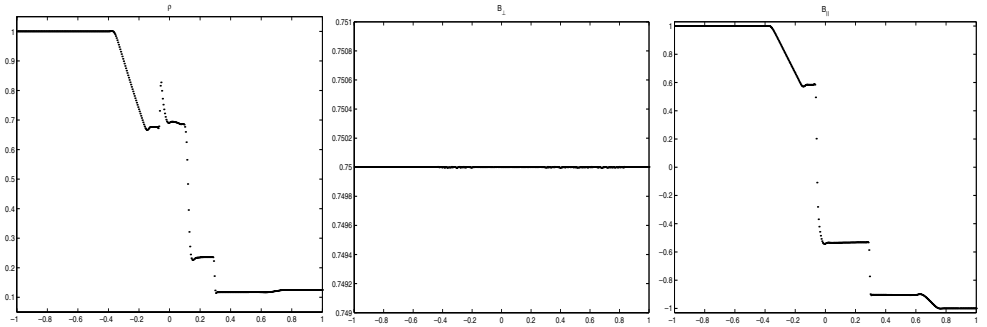


FIG. 5.1. *Results of rotated Brio–Wu shock tube problem at* $t = 0.2$ *computed over a* 600×600 *mesh using third-order dimension-by-dimension CWENO reconstruction,* (3.1)*, and Shu's SSP Runge–Kutta solver,* (3.19)*. From left to right: density, transverse, and parallel components of magnetic field (bottom)*

Figure 5.1, and they confirm the ability of central schemes to simulate discontinuous flows independently of the mesh orientation.

We observe that the small deviations from the constant value of the transverse component of the magnetic that can be observed in Figure 5.1 are well within the order of the scheme, and the numerical value of $\nabla \cdot \mathbf{B}$ is in the order of 10^{-13}.

5.2. Transverse Kelvin–Helmholtz instability. The Kelvin–Helmholtz instability arises when two superposed fluids flow one over the other with a relative velocity. It models, for example, the important mechanism for the momentum transfer at the Earth's magnetopause boundary, which separates the solar wind from the Earth's magnetosphere [11]. We approximate the solution of the two-dimensional periodic and convective models with transverse magnetic field configuration with the semidiscrete scheme of Kurganov and Tadmor, (2.24)–(2.26), implemented along with the reconstruction of section 3.1 and the SSP Runge–Kutta solver (3.19). In both cases, the governing equations are (5.1)–(5.3) are subject to initial conditions

$$(\rho, v_x, v_y, v_z, B_x, B_y, B_z, p)^\top = (1.0, v_{x0} + \tilde{v}_{x0}, 0, 0, 0, 1.0, 0, 0.5)^\top, \tag{5.5}$$

where

$$v_{x0} = \frac{v_0}{2} \tanh\left(\frac{z}{a}\right) \quad \text{and} \tag{5.6}$$

$$\tilde{v}_{x0} = \begin{cases} -\tilde{v}_0 \sin\left(\frac{2\pi x}{\lambda}\right) \frac{1}{1+z^2} & \text{if } -\frac{\lambda}{2} < x < \frac{\lambda}{2} \\ 0 & \text{otherwise} \end{cases}, \tag{5.7}$$

with $v_0 = 2$, $\tilde{v}_0 = -0.008$, $\lambda = 5\pi$, and $a = 1$. Also, the grids are stretched in the z-direction with a Roberts transformation [11],

$$z \leftarrow \frac{H \sinh(\tau z/2H)}{\sinh(\tau/2)}, \quad \tau = 6, \tag{5.8}$$

which renders a denser grid near $z = 0$, where the effect of the small initial perturbation $\tilde{v}_{x0}$ is more noticeable, and a coarser grid near $z = \pm H$, where little action takes place. The time step, Δt, is determined according to (5.4) with $c = 0.9$.

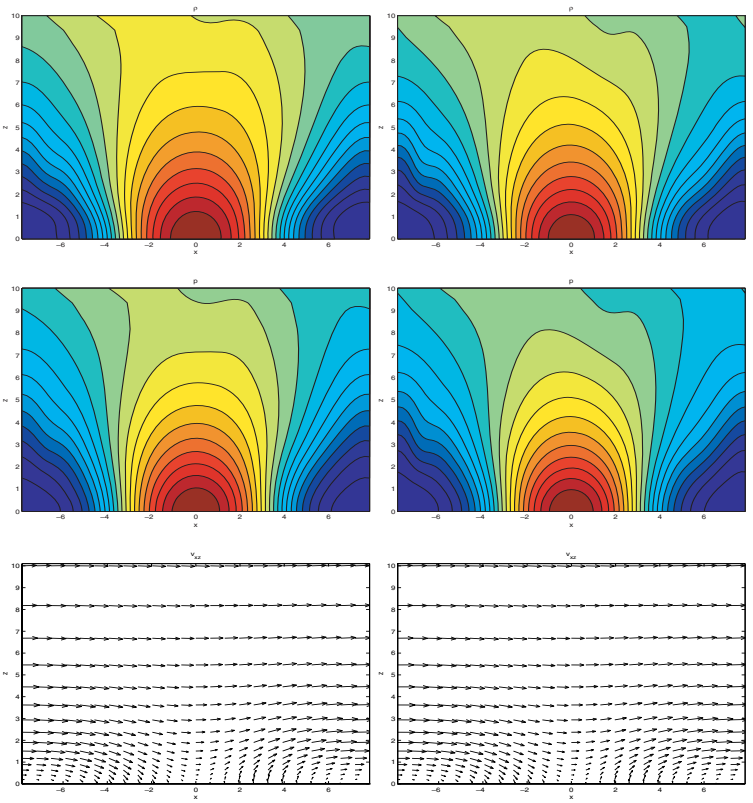


FIG. 5.2. *Results of transverse Kelvin–Helmholtz instability with periodic x-boundary conditions computed with third-order scheme. Left column uses 72×45 points and right column uses 96×60 grids, respectively. There are 20 contours for density and pressure, where darker colors indicate higher values. The density ranges from 0.79 to 1.2, the pressure ranges from 0.32 to 0.71, and the maximum value for the velocity is 1.25.*

In the periodic case, the computational domain is $[-\frac{L}{2}, \frac{L}{2}] \times [0, H]$, with $L = 5\pi$ and $H = 10$. The free outflow condition is applied at the top boundary, $z = H$, and the bottom boundary values are recovered by symmetry, since ρ, p, and B_y are symmetric, and v_x and v_z are antisymmetric under the transformation $(x, z) \to (-x, -z)$. In Figure 5.2, we present solutions at $t = 144$ with 72×45 and 96×60 grid points. The resolution and accuracy of our results are comparable to those reported in [2] obtained over two grids of sizes 96×96 and a 192×192, respectively, with the second-order staggered central scheme of Jiang and Tadmor; the gas kinetic scheme of Tang and Xu, [26], using a 200×200 uniform grid; and to those obtained with Jiang and Wu's fifth-order WENO scheme [11] over two grids of sizes 48×30 and 96×60, respectively. The higher computational cost of the third-order reconstruction as compared to that of the fully discrete second-order scheme we used in [2] is compensated by the smaller number of grid points needed to resolve accurately the steep gradients that characterize the solution.

In the convective setup, the initial conditions and perturbation are the same as in the periodic setup, (5.5)–(5.7). In this case, the free outflow condition is applied also in the x-direction over the computational domain $[-\frac{L}{2}, \frac{L}{2}] \times [0, H]$, where $H = 20$ and $L = 55\pi$ with $L >> \lambda$—so chosen to allow the excitation to convect freely without disturbing the x-boundaries. The values of the bottom boundary of the computational

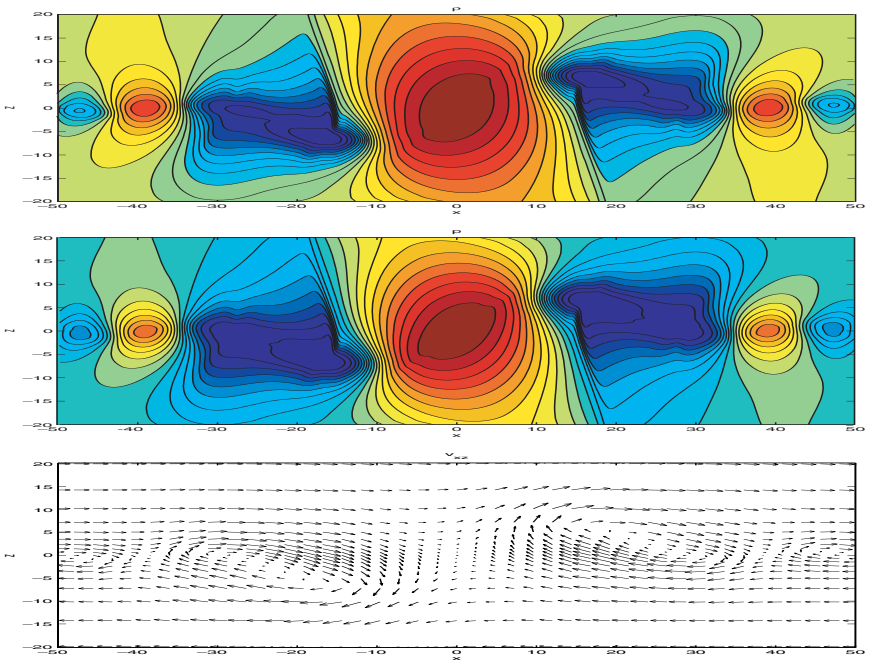


FIG. 5.3. *Solution of convective Kelvin–Helmholtz instability at* $t = 120$*, computed with third-order scheme on a* 792×72 *grid with* $c = 0.9$*. There are* 20 *contours for density and pressure, where darker colors indicate higher values. The density ranges from* 0.63 *to* 1.3*, the pressure ranges from* 0.20 *to* 0.85*, and the maximum value for the velocity is* 1.54.

domain, $z = 0$, are recovered by symmetry as in the periodic configuration. Figures 5.3 and 5.4 display the solution over the region $[-50, 50] \times [-20, 20]$ computed with 792×92 grid points over computational domain at $t = 120$ and $t = 145$, respectively. The values of the bottom half of the solution domain are recovered using the same symmetry conditions as used to reconstruct the bottom boundary.

5.3. Orszag–Tang MHD turbulence problem. This model considers the evolution of a compressible Orszag–Tang vortex system. The evolution of the vortex system involves the interaction between several shock waves traveling at various speed regimes [11, 28], which makes the problem useful to validate the robustness of numerical schemes. The initial data is given by

$$\rho(x, z, 0) = \gamma^2, v_x(x, z, 0) = -\sin z,\ v_z(x, z, 0) = \sin x,$$
$$p(x, z, 0) = \gamma, B_x(x, z, 0) = -\sin z,\ B_z(x, z, 0) = \sin 2x,$$

where $\gamma = 5/3$. With these data, the RMS values of the velocity and magnetic fields are both 1; the initial average Mach number is 1; and the average plasma beta is $10/3$. We solve the problem in $[0, 2\pi] \times [0, 2\pi]$ with periodic boundary conditions in both x- and z-directions. For this problem we implement the fourth-order reconstruction of section 3.2 and the SSP Runge–Kutta fourth-order ODE solver (3.20) using a uniform grid with 192×192 points and $c = 0.9$ in (5.4).

Figures 5.6, 5.7, and 5.8 display the solution of the Orszag–Tang vortex system at $t = 0.5, t = 2$, and $t = 3$, respectively. These simulations were performed with 192×192 grid points using the fourth-order, genuinely multidimensional, CWENO reconstruction of section 3.2 SSP Runge–Kutta solver (3.18)–(3.20). These results

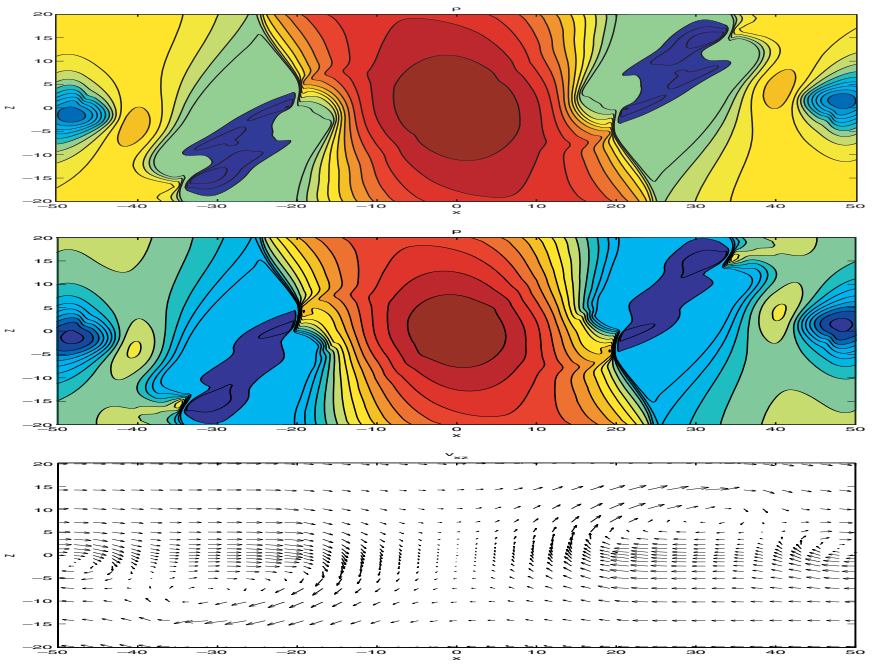


FIG. 5.4. *Solution of convective Kelvin–Helmholtz instability at* $t = 145$*, computed with third-order scheme on a* 792×72 *grid. There are* 20 *contours for density and pressure, where darker colors indicate higher values. The density ranges from* 0.43 *to* 1.3*, the pressure ranges from* 0.10 *to* 0.86*, and the maximum value for the velocity is* 1.94.

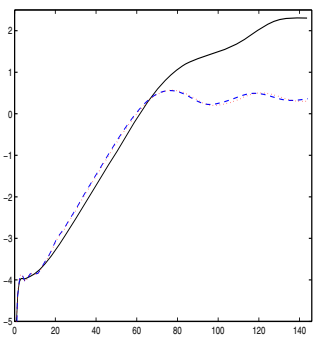


FIG. 5.5. *Time evolution of the total transverse kinetic energy,* $\log(\frac{1}{2}\int \rho v_z^2 dxdz)$*, integrated over* $[-L/2, L/2] \times [-H, H]$*, for both periodic and convective Kelvin–Helmholtz instability. The results for the periodic case with* 96×96 *and* 192×192 *grid points are represented by a dashed and a dotted curve, respectively. The convective configuration is represented by a solid line.*

demonstrate the ability of higher-order central schemes to resolve the shocks that the vortex system develops while maintaining the simplicity and ease of implementation typical of this [illegible] type of finite-difference schemes. We also note that similar results were obtained with the dimension-by-dimension third-order reconstruction and the Runge–Kutta solver (3.19) using a 288×288 mesh. The improved resolution of the fourth-order scheme allows us to compute accurate approximations using coarser grids than those required by lower-order schemes—consult Figures 5.5 and 5.9, for

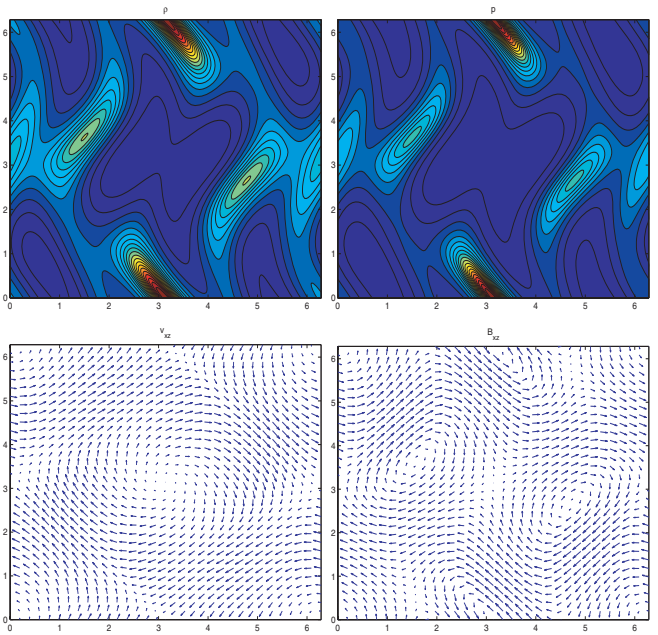


FIG. 5.6. *Fourth-order solution of Orszag–Tang MHD turbulence problem with a* 192×192 *uniform grid at* $t = 0.5$. *There are* 20 *contours for density and pressure, where darker colors indicate higher values. Density ranges from* 2.1 *to* 5.9, *pressure ranges from* 1.0 *to* 5.8. *The maximum values of* $|v|$ *and* $|B|$ *are* 1.6 *and* 1.6, *respectively.*

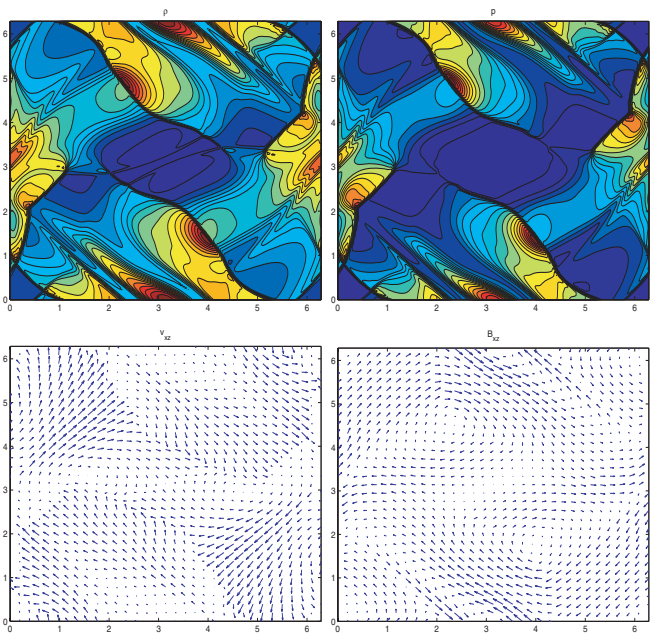


FIG. 5.7. *Fourth-order solution of Orszag–Tang MHD turbulence problem with a* 192×192 *uniform grid at* $t = 2$. *There are* 20 *contours for density and pressure, where darker colors indicate higher values. Density ranges from* 0.62 *to* 6.3, *pressure ranges from* 0.14 *to* 7.0. *The maximum values of* $|v|$ *and* $|B|$ *are* 1.6 *and* 2.8, *respectively.*

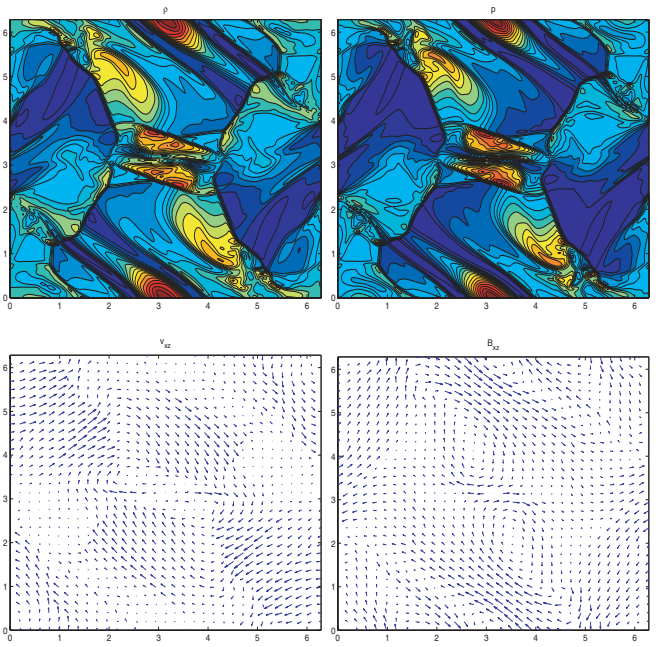


FIG. 5.8. *Fourth-order solution of Orszag–Tang MHD turbulence problem with a* 192×192 *uniform grid at* $t = 2$. *There are* 20 *contours for density and pressure, where darker colors indicate higher values. Density ranges from* 0.62 *to* 6.3, *pressure ranges from* 0.14 *to* 7.0. *The maximum values of* $|v|$ *and* $|B|$ *are* 1.6 *and* 2.8, *respectively.*

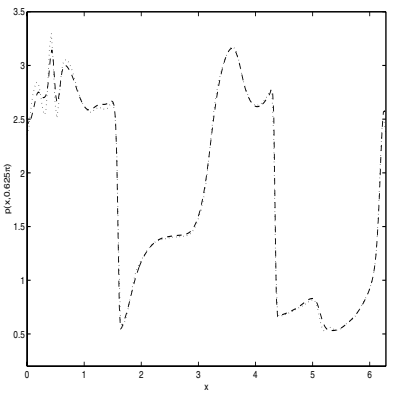


FIG. 5.9. *Pressure distribution along the line* $z = 0.625\pi$ *at* $t = 3$. *Dashed line corresponds to third-order approximation with* 288×288 *grid points and dotted line corresponds to fourth-order results computed with* 192×192 *gridpoints.*

example, thus reducing the computational cost of this type of simulation, an advantage that is more evident in higher space dimensions.

5.4. Interaction between a strong shock and a high-density cloud. This problem, introduced in [4], describes the disruption of a high-density cloud by a strong shock wave. The flow is simulated over the computational domain $[0, 1] \times [0, 1]$ with 256×256 cells and open boundary conditions. The ratio of specific heats is $\gamma = 5/3$

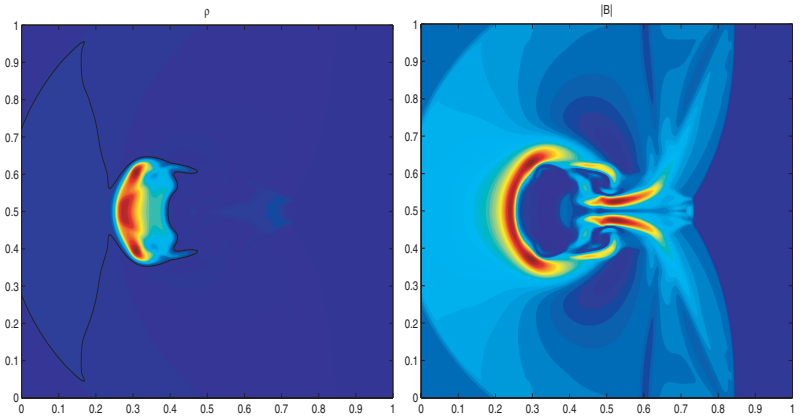


FIG. 5.10. *Third-order solution of the interaction between a strong shock and a high-density cloud obtained with a* 256×256 *uniform grid at* $t = 0.06$, $c = 0.5$. *Left: density; right: magnetic field,* $|B|$.

and the initial conditions are given by the two states,

(5.9)
$$(\rho, v_x, v_y, v_z, B_x, B_y, B_z, p)^\top = \begin{cases} (3.86859, 0, 0, 0, 0, -2.1826182, 2.1826182, 167.34)^\top & \text{for } x < 0.6, \\ (1, -11.2536, 0, 0, 0, 0.56418958, 0.56418958, 1)^\top & \text{for } x > 0.6, \end{cases}$$

separated by a discontinuity parallel to the z-axis at $x = 0.6$. To the right of the discontinuity there is a circular cloud of radius 0.15, centered at $x = 0.8$ and $z = 0.5$ with density $\rho = 10$ and $p = 1$. The solution we present in Figure 5.10 at $t = 0.06$ confirms the robustness of central schemes for highly superfast flows.

As in the case of the shock tube problem, the presence of discontinuities that affect only a particular component of the system—in this case a rotational discontinuity in the y component of the magnetic field—requires the careful choice of the smoothness indicators used in the calculation of the weights of the nonoscillatory reconstruction so as to prevent oscillations. Our numerical experiments suggest the use of the average of the smoothness indicators corresponding to the y-magnetic field and the density to determine the least oscillatory stencil for the evolution of all the conserved quantities.

6. Conclusions. The numerical schemes presented in this work and the results obtained with them complement those introduced in [2] and references therein and confirm the ability of this type of black-box solver to approximate the discontinuous solutions of ideal MHD equations accurately. The high-order semidiscrete schemes described in sections 2 and 3 retain the simplicity and ease of implementation of the fully discrete [illegible] methods while adding further advantages. As the limit of the reprojection of the staggered evolution of the cell averages, the semidiscrete formulation (2.17) requires only an estimate of the maximum local speeds of propagation to evolve the solution, avoiding any detailed knowledge of the eigen structure of (1.1)–(1.4) and, therefore, eliminating the need for Riemann solvers. The efficiency of the semidiscrete formulation is further enhanced by the larger time step that the implementation over nonstaggered grids yields and the wide range of reconstruction algorithms and ODE solvers available for its implementation. The use of these higher-order routines offers advantages that extend beyond the work presented here, in particular, the ability of

these schemes to compute accurate solutions over coarser grids and the possibility of applying the reconstruction algorithm dimension by dimension suggest these as an attractive alternative for schemes in higher space dimensions.

Our results also suggest a remarkable ability of central schemes to avoid any instability caused by the failure to satisfy the constraint $\nabla \cdot \mathbf{B} = 0$. All the results reported here have been calculated without any additional treatment for this constraint. Indeed, it can be shown that the magnetic field computed with the second-order staggered fully discrete schemes that lead to the semidiscrete formulation presented here satisfy the condition

$$\nabla \cdot \overline{\mathbf{B}}^{\,n+1}_{j+\frac{1}{2},k+\frac{1}{2}} = \nabla \cdot \overline{\mathbf{B}}^{\,n}_{j+\frac{1}{2},k+\frac{1}{2}}, \tag{6.1}$$

where $\overline{\mathbf{B}}^{\,n}_{j+\frac{1}{2},k+\frac{1}{2}}$ represents the cell average of the nonoscillatory polynomial reconstruction of the cell averages $\{\overline{\mathbf{B}}^{\,n}_{j+p,k+q}\}_{p,q=0,1}$.

Animations of the simulations presented in sections 4 and 5 and sample MHD codes are available online at http://www.math.lsa.umich.edu/∼jbalbas/MHD.

Acknowledgments. We thank C. Anderson of the UCLA Department of Mathematics for his advice in the computational aspects of this work and Dr. C-.C. Wu of the UCLA Department of Physics for his comments and suggestions.

REFERENCES

[1] J. Balbás, *Non-oscillatory Central Schemes for the Equations of Ideal Magnetohydrodynamics in One- and Two-Space Dimensions*, Ph.D. thesis, University of California, Los Angeles, May 2004.

[2] J. Balbás, E. Tadmor, and C.-C. Wu, *Non-oscillatory central schemes for one- and two-dimensional MHD equations.* I, J. Comput. Phys., 201 (2004), pp. 261–285.

[3] M. Brio and C. C. Wu, *An upwind differencing scheme for the equations of ideal magnetohydrodynamics*, J. Comput. Phys., 75 (1988), pp. 400–422.

[4] W. Dai and P. R. Woodward, *A simple finite difference scheme for multidimensional magnetohydrodynamical equations*, J. Comput. Phys., 142 (1998), pp. 331–369.

[5] S. Gottlieb, C.-W. Shu, and E. Tadmor, *Strong stability-preserving high-order time discretization methods*, SIAM Rev., 43 (2001), pp. 89–112.

[6] A. Harten, B. Engquist, S. Osher, and S. R. Chakravarthy, *Uniformly high-order accurate essentially nonoscillatory schemes.* III, J. Comput. Phys., 71 (1987), pp. 231–303.

[7] A. Harten, *High resolution schemes for hyperbolic conservation laws*, J. Comput. Phys., 49 (1983), pp. 357–393.

[8] G.-S. Jiang, D. Levy, C.-T. Lin, S. Osher, and E. Tadmor, *High-resolution nonoscillatory central schemes with nonstaggered grids for hyperbolic conservation laws*, SIAM J. Numer. Anal., 35 (1998), pp. 2147–2168.

[9] G.-S. Jiang and C.-W. Shu, *Efficient implementation of weighted ENO schemes*, J. Comput. Phys., 126 (1996), pp. 202–228.

[10] G.-S. Jiang and E. Tadmor, *Nonoscillatory central schemes for multidimensional hyperbolic conservation laws*, SIAM J. Sci. Comput., 19 (1998), pp. 1892–1917.

[11] G.-S. Jiang and C.-C. Wu, *A high-order WENO finite difference scheme for the equations of ideal magnetohydrodynamics*, J. Comput. Phys., 150 (1999), pp. 561–594.

[12] A. Kurganov and D. Levy, *A third-order semidiscrete central scheme for conservation laws and convection-diffusion equations*, SIAM J. Sci. Comput., 22 (2000), pp. 1461–1488.

[13] A. Kurganov and G. Petrova, *A third-order semi-discrete genuinely multidimensional central scheme for hyperbolic conservation laws and related problems*, Numer. Math., 88 (2001), pp. 683–729.

[14] A. Kurganov and E. Tadmor, *New high-resolution central schemes for nonlinear conservation laws and convection-diffusion equations*, J. Comput. Phys., 160 (2000), pp. 241–282.

[15] P. D. Lax, *Weak solutions of nonlinear hyperbolic equations and their numerical computation*, Comm. Pure Appl. Math., 7 (1954), pp. 159–193.

[16] D. Levy, G. Puppo, and G. Russo, *Central WENO schemes for hyperbolic systems of conservation laws*, Math. Model. Numer. Anal., 33 (1999), pp. 547–571.

[17] D. Levy, G. Puppo, and G. Russo, *A third order central WENO scheme for 2D conservation laws*, Appl. Numer. Math., 33 (2000), pp. 415–421.

[18] D. Levy, G. Puppo, and G. Russo, *A fourth-order central WENO scheme for multidimensional hyperbolic systems of conservation laws*, SIAM J. Sci. Comput., 24 (2002), pp. 480–506 (electronic).

[19] X.-D. Liu and E. Tadmor, *Third order nonoscillatory central scheme for hyperbolic conservation laws*, Numer. Math., 79 (1998), pp. 397–425.

[20] H. Nessyahu and E. Tadmor, *Nonoscillatory central differencing for hyperbolic conservation laws*, J. Comput. Phys., 87 (1990), pp. 408–463.

[21] S. A. Orszag and C. M. Tang, *Small-scale structure of two-dimensional magnetohydrodynamic turbulence*, J. Fluid Mech., 90 (1979), pp. 129–143.

[22] K. G. Powell, P. L. Roe, R. S Myong, T. Gombosi, and D. de Zeeuw, *An upwind scheme for magnetohydrodynamics*, in Proceedings of the AIAA 12th Computational Fluid Dynamics Conference, San Diego, 1995, pp. 661–674.

[23] C.-W. Shu, *Total-variation-diminishing time discretizations*, SIAM J. Sci. Statist. Comput., 9 (1988), pp. 1073–1084.

[24] C.-W. Shu, *Essentially non-oscillatory and weighted essentially non-oscillatory schemes for hyperbolic conservation laws*, in Advanced Numerical Approximation of Nonlinear Hyperbolic Equations, Lecture Notes in Math. 1697, Springer, Berlin, 1998, pp. 325–432.

[25] C.-W. Shu and S. Osher, *Efficient implementation of essentially nonoscillatory shock-capturing schemes.* II, J. Comput. Phys., 83 (1989), pp. 32–78.

[26] H.-Z. Tang and K. Xu, *A high-order gas-kinetic method for multidimensional ideal magnetohydrodynamics*, J. Comput. Phys., 165 (2000), pp. 69–88.

[27] B. van Leer, *Towards the ultimate conservative difference scheme.* V. *Second order sequel to Godunov method*, J. Comput. Phys., 32 (1979), pp. 101–136.

[28] C.-C. Wu and T. Chang, *Further study of the dynamics of two-dimensional MHD coherent structures—a large scale simulation*, J. Atmospheric Solar-Terrestrial Phys., 95 (2001), pp. 1447–1453.

SIAM J. SCI. COMPUT.
Vol. 28, No. 2, pp. 561–582
© 2006 Society for Industrial and Applied Mathematics

NUMERICAL ALGORITHMS FOR FORWARD-BACKWARD STOCHASTIC DIFFERENTIAL EQUATIONS*

G. N. MILSTEIN† AND M. V. TRETYAKOV‡

Abstract. Efficient numerical algorithms are proposed for a class of forward-backward stochastic differential equations (FBSDEs) connected with semilinear parabolic partial differential equations. As in [J. Douglas, Jr., J. Ma, and P. Protter, *Ann. Appl. Probab.*, 6 (1996), pp. 940–968], the algorithms are based on the known four-step scheme for solving FBSDEs. The corresponding semilinear parabolic equation is solved by layer methods which are constructed by means of a probabilistic approach. The derivatives of the solution u of the semilinear equation are found by finite differences. The forward equation is simulated by mean-square methods of order 1/2 and 1. Corresponding convergence theorems are proved. Along with the algorithms for FBSDEs on a fixed finite time interval, we also construct algorithms for FBSDEs with random terminal time. The results obtained are supported by numerical experiments.

Key words. forward-backward stochastic differential equations, numerical integration, mean-square convergence, semilinear partial differential equations of parabolic type

AMS subject classifications. Primary, 60H35; Secondary, 65C30, 60H10, 62P05

DOI. 10.1137/040614426

1. Introduction.

Forward-backward stochastic differential equations (FBSDEs) have numerous applications in stochastic control theory and mathematical finance (see, e.g., [7, 5, 9, 25, 12] and references therein). Consider an FBSDE of the form

$$dX = a(t,X,Y)dt + \sigma(t,X,Y)dw(t), \quad X(t_0) = x, \tag{1.1}$$

$$dY = -g(t,X,Y)dt - f^{\intercal}(t,X,Y)Zdt + Z^{\intercal}dw(t), \quad Y(T) = \varphi(X(T)). \tag{1.2}$$

Here $X = X(t)$ and $a = a(t,x,y)$ are d-dimensional vectors; $\sigma = \sigma(t,x,y)$ is a $d \times n$-matrix; $Y = Y(t)$, $g = g(t,x,y)$, and $\varphi = \varphi(x)$ are scalars; $Z = Z(t)$ and $f = f(t,x,y)$ are n-dimensional vectors; and $w(t)$ is an n-dimensional $\{\mathcal{F}_t\}_{t\ge 0}$-adapted Wiener process, where $(\Omega, \mathcal{F}, \mathcal{F}_t, P)$, $t_0 \le t \le T$, is a filtered probability space. It is known (see, e.g., [1, 11, 19, 25, 12] and also references therein) that there exists a unique $\{\mathcal{F}_t\}_{t\ge 0}$-adapted solution $(X(t), Y(t), Z(t))$ of the system (1.1)–(1.2) under some appropriate smoothness and boundedness conditions on its coefficients.

Due to the four-step scheme from [11] (we recall it in the next section), the solution of (1.1)–(1.2) is connected with the Cauchy problem for the semilinear partial

*Received by the editors September 3, 2004; accepted for publication (in revised form) November 16, 2005; published electronically April 21, 2006. This research received partial support from EPSRC research grant GR/R90017/01 and from the Royal Society International Joint Project-2004/R2-FS grant.

http://www.siam.org/journals/sisc/28-2/61442.html

†Department of Mathematics, Ural State University, Lenin Str. 51, 620083 Ekaterinburg, Russia (Grigori.Milstein@usu.ru).

‡Department of Mathematics, University of Leicester, Leicester LE1 7RH, UK (M.Tretiakov@le.ac.uk).

differential equation (PDE):

$$\frac{\partial u}{\partial t}+\sum_{i=1}^{d}a^i(t,x,u)\frac{\partial u}{\partial x^i}+\frac{1}{2}\sum_{i,j=1}^{d}a^{ij}(t,x,u)\frac{\partial^2 u}{\partial x^i\partial x^j} \tag{1.3}$$

$$=-g(t,x,u)-\sum_{k=1}^{n}f^k(t,x,u)\sum_{i=1}^{d}\sigma^{ik}(t,x,u)\frac{\partial u}{\partial x^i},\quad t<T,\quad x\in\mathbf{R}^d,$$

$$u(T,x)=\varphi(x), \tag{1.4}$$

where

$$a^{ij}:=\sum_{k=1}^{d}\sigma^{ik}\sigma^{jk}.$$

In turn the corresponding solution of the semilinear PDE has a probabilistic representation using the FBSDE (1.1)–(1.2), which is a generalization of the Feynman–Kac formula (see, e.g., [22, 20, 19, 21, 25, 12]).

Our aim is to find an effective numerical algorithm for solving (1.1)–(1.2) in the mean-square sense. Not many works have been devoted to numerical integration of FBSDEs, but let us mention [4, 6, 2, 3]. Among these works, the present paper is most closely connected with [6]. As in [6], we exploit the four-step scheme for numerical solution of (1.1)–(1.2). Unlike [6], we use another approach for numerical solution of the corresponding semilinear PDE (1.3)–(1.4), which is developed in [14] (see also [16, 17, 18]).

The most significant distinction between our paper and [6] consists of numerical evaluation of Z. According to the four-step scheme, Z is expressed through first derivatives of the solution $u(t,x)$ to (1.3)–(1.4) with respect to x^i, $i=1,\dots,d$, while finding X and Y of (1.1)–(1.2) requires knowledge of u only. In [6] the authors write a system of semilinear PDEs for u and $\partial u/\partial x^i$, $i=1,\dots,d$, and solve it numerically in order to simulate then X, Y, and Z. This approach is rather expensive from the computational point of view. We approximate the derivatives by finite differences; thus we need to simulate u only. As it is proved in section 3, this approximation gives quite accurate results.

Along with the algorithms for FBSDEs on a fixed finite time interval, we also construct algorithms for FBSDEs with random terminal time. To the authors' best knowledge, numerical solution of FBSDEs with random terminal time is considered for the first time. Let us note that in this case the approach of [6] leads to a complicated system of boundary value problems whose numerical solution is less effective than the algorithms proposed here.

For clarity of exposition, we state and prove our results for the one-dimensional case ($d=1$, $n=1$) although it is not difficult to generalize them to any dimension. However, since the obtained algorithms require simulation of nonlinear PDEs, they can realistically be used in practice for solving FBSDEs with forward equation of dimension three or lower ($d\le 3$). At the same time, the variable Y can be of a high dimension; in such a case we shall deal with a system of semilinear parabolic PDEs instead of (1.3)–(1.4).

Let us note that in [6] a more general FBSDE than (1.1)–(1.2) is considered; it is connected with quasi-linear parabolic equations. We will consider this case in a separate publication.

In section 2 we recall the four-step scheme due to [11] and the probabilistic approach to numerical solution of the Cauchy problem (1.3)–(1.4) from [14] (see also [18]). In section 3 we obtain some results concerning accuracy of approximating derivatives $\partial u/\partial x^i$ by finite differences. In section 4 we prove the mean-square convergence of the Euler method for FBSDEs when using the approximations of u and $\partial u/\partial x$ given in the previous two sections. Sections 5 and 6 are devoted to FBSDEs with random terminal time (see e.g., [19, 25] and references therein). In these sections we have to consider the Dirichlet boundary value problem for semilinear parabolic PDEs instead of (1.3)–(1.4). For solving the Dirichlet problem we use the probabilistic approach again [16, 18], and for simulating solutions of FBSDEs with random terminal time we use the approximations of SDEs in space-time bounded domains [15, 18]. We also consider the case of unbounded random terminal time, which is connected with the Dirichlet boundary value problem for semilinear elliptic PDEs. The results obtained are supported by numerical experiments which are presented in section 7.

2. Preliminaries.

2.1. Four-step scheme for solving FBSDEs. First we recall the four-step scheme for solving FBSDE (1.1)–(1.2) [11]. Assume that the solution $u(t,x)$ of the Cauchy problem (1.3)–(1.4) is known. Consider the following SDE:

$$dX = a(t,X,u(t,X))dt + \sigma(t,X,u(t,X))dw(t), \quad X(t_0) = x. \tag{2.1}$$

Let $X(t) = X_{t_0,x}(t)$ be a solution of the Cauchy problem (2.1). Introduce

$$\begin{aligned} Y(t) &= u(t, X_{t_0,x}(t)), \\ Z^j(t) &= \sum_{i=1}^{d} \sigma^{ij}(t, X_{t_0,x}(t), Y(t)) \frac{\partial u}{\partial x^i}(t, X_{t_0,x}(t)), \qquad j = 1, \dots, n. \end{aligned} \tag{2.2}$$

It turns out that $(X(t), Y(t), Z(t))$ defined by (2.1)–(2.2) is the solution of the FBSDE (1.1)–(1.2). Indeed, it is $\{\mathcal{F}_t\}_{t\geq 0}$-adapted and (1.1) is evidently satisfied. To verify (1.2), it suffices to apply Ito's formula to $Y(t) = u(t, X_{t_0,x}(t))$.

2.2. Layer methods for PDEs. Now we recall layer methods for solving the problem (1.3)–(1.4) due to [14] (see also [18]). For simplicity in writing, we restrict ourselves to a one-dimensional version of the problem ($d = 1$, $n = 1$). Introducing

$$b(t,x,y) := a(t,x,y) + f(t,x,y)\sigma(t,x,y),$$

we get

$$\frac{\partial u}{\partial t} + b(t,x,u)\frac{\partial u}{\partial x} + \frac{1}{2}\sigma^2(t,x,u)\frac{\partial^2 u}{\partial x^2} + g(t,x,u) = 0, \quad t < T,\ x \in \mathbf{R}, \tag{2.3}$$

$$u(T,x) = \varphi(x)\,. \tag{2.4}$$

The solution to this problem is supposed to exist, be unique, be sufficiently smooth, and satisfy some conditions on boundedness. One can find theoretical results on this topic in [10, 23, 8, 24] (see also references therein). For convenience, we shall assume throughout the paper that the standing assumptions of [6] are fulfilled. We prefer to state them here in a less specific way since different numerical methods will require, e.g., different levels of smoothness of the coefficients.

ASSUMPTION 2.1. [illegible] b, σ, g [illegible] φ [illegible] $[t_0, T] \times \mathbf{R} \times \mathbf{R}$. [illegible] σ [illegible] $\sigma \geq \sigma_0$, [illegible] σ_0 [illegible]

Assumption 2.1 ensures the existence of a unique bounded solution $u(t, x)$ with bounded derivatives up to some order. We note that these assumptions are more than necessary (for instance, other types of assumptions are given in [14] (see also [18, pp. 419, 422])), and the methods constructed in this paper can be used under broader conditions.

We introduce a time discretization; to be definite let us take the equidistant one:

$$T = t_N > t_{N-1} > \cdots > t_0, \quad h := \frac{T - t_0}{N}.$$

Layer methods proposed in [14] (see also [18]) give an approximation $\bar{u}(t_k, x)$ of the solution $u(t_k, x)$, $k = N, \ldots, 0$, to (2.3)–(2.4). These methods are based on the local probabilistic representation of the solution to (2.3)–(2.4):

$$u(t_k, x) = E\left(u(t_{k+1}, X_{t_k,x}(t_{k+1})) + \int_{t_k}^{t_{k+1}} g(s, X_{t_k,x}(s), u(s, X_{t_k,x}(s)))ds\right), \tag{2.5}$$

where $X_{t_k,x}(s)$ is the solution of the Cauchy problem for the SDE

$$dX = b(s, X, u(s, X))ds + \sigma(s, X, u(s, X))dw(s), \quad X(t_k) = x. \tag{2.6}$$

Exploiting the weak Euler scheme, the following layer method is constructed [14, 18]:

$$\tilde{u}(t_N, x) = \varphi(x), \tag{2.7}$$

$$\begin{aligned}\tilde{u}(t_k, x) = {} & \frac{1}{2}\tilde{u}(t_{k+1}, x + hb(t_k, x, \tilde{u}(t_{k+1}, x)) - \sqrt{h}\sigma(t_k, x, \tilde{u}(t_{k+1}, x))) \\ & + \frac{1}{2}\tilde{u}(t_{k+1}, x + hb(t_k, x, \tilde{u}(t_{k+1}, x)) + \sqrt{h}\sigma(t_k, x, \tilde{u}(t_{k+1}, x))) \\ & + hg(t_k, x, \tilde{u}(t_{k+1}, x)), \quad k = N-1, \ldots, 1, 0.\end{aligned}$$

It is proved (see either [14] or [18, p. 420]) that this method is of order one; i.e.,

$$|\tilde{u}(t_k, x) - u(t_k, x)| \leq Kh, \tag{2.8}$$

where K does not depend on x, h, k.

To obtain a numerical algorithm, we need to discretize (2.7) in the variable x. Consider the equidistant space discretization

$$x_j = x_0 + j\varkappa h, \quad j = 0, \pm 1, \pm 2, \ldots,$$

where x_0 is a point on $\mathbf{R}$ and $\varkappa$ is a positive number; i.e., the step h_x of space discretization is equal to $\varkappa h$, where $h = h_t$ is the step of time discretization. Using, for instance, linear interpolation, we construct the following algorithm on the basis of the layer method (2.7).

ALGORITHM 2.2.

$$\bar{u}(t_N, x) = \varphi(x), \tag{2.9}$$

$$\bar{u}(t_k,x_j)=\frac{1}{2}\bar{u}(t_{k+1},x_j+hb(t_k,x_j,\bar{u}(t_{k+1},x_j))-\sqrt{h}\sigma(t_k,x_j,\bar{u}(t_{k+1},x_j)))$$
$$+\frac{1}{2}\bar{u}(t_{k+1},x_j+hb(t_k,x_j,\bar{u}(t_{k+1},x_j))+\sqrt{h}\sigma(t_k,x_j,\bar{u}(t_{k+1},x_j)))$$
$$+hg(t_k,x_j,\bar{u}(t_{k+1},x_j)),\quad j=0,\pm1,\pm2,\dots,$$

$$\bar{u}(t_k,x)=\frac{x_{j+1}-x}{\varkappa h}\bar{u}(t_k,x_j)+\frac{x-x_j}{\varkappa h}\bar{u}(t_k,x_{j+1}),\quad x_j\le x\le x_{j+1}, \tag{2.10}$$

$$k=N-1,\dots,1,0.$$

Algorithm 2.2 is of order one; i.e.,

$$|\bar{u}(t_k,x)-u(t_k,x)|\le Kh\,, \tag{2.11}$$

where K does not depend on x, h, k (see either [14] or [18, p. 423] for a proof).

Along with linear interpolation, a spline approximation is also considered and the cubic interpolation with step $h_x=\varkappa\sqrt{h}$ is used to reduce the number of nodes x_j (see [14, 18]). Clearly, both the method and algorithm can be considered with variable time and space steps. Algorithms for the multidimensional parabolic problems (such as (1.3)–(1.4) with $d>1$) are available in [14, 18] as well.

3. Approximation of the derivative $\partial u/\partial x$ by finite differences. Consider the solution $u(t,x)$ of the Cauchy problem for semilinear parabolic equation (2.3)–(2.4) and its approximations $\tilde{u}(t_k,x)$ by the layer method (2.7) and $\bar{u}(t_k,x)$ by Algorithm 2.2.

PROPOSITION 3.1. [illegible]

$$\frac{\partial u}{\partial x}(t_k,x)=\frac{\bar{u}(t_k,x+\gamma\sqrt{h})-\bar{u}(t_k,x-\gamma\sqrt{h})}{2\gamma\sqrt{h}}+O(h^{1/2}), \tag{3.1}$$

[illegible] (2.9) (2.10) [illegible] $\bar{u}$ [illegible] $\tilde{u}$.

[illegible]. Since the solution u has bounded third derivatives with respect to x (we assume that the functions from Assumption 2.1 have continuous bounded first derivative with respect to t and second derivatives with respect to x and u including the mixed ones), we have

$$\frac{\partial u}{\partial x}(t_k,x)=\frac{u(t_k,x+\gamma\sqrt{h})-u(t_k,x-\gamma\sqrt{h})}{2\gamma\sqrt{h}}+O(h).$$

Now (3.1) immediately follows from the inequality (2.11). □

[illegible] 3.2. Analogous to (3.1), we also get

$$\frac{\partial u}{\partial x}(t_k,x)=\frac{\bar{u}(t_k,x+\gamma h^{1/3})-\bar{u}(t_k,x-\gamma h^{1/3})}{2\gamma h^{1/3}}+O(h^{2/3}). \tag{3.2}$$

In fact, it is possible to prove a more accurate result than (3.1) or (3.2) for the layer approximation $\tilde{u}$ from (2.7).

THEOREM 3.3. [illegible]

$$\frac{\partial u}{\partial x}(t_k,x)=\frac{\tilde{u}(t_k,x+\gamma\sqrt{h})-\tilde{u}(t_k,x-\gamma\sqrt{h})}{2\gamma\sqrt{h}}+O(h), \tag{3.3}$$

[illegible] γ [illegible] h [illegible] (2.7) [illegible]. Clearly, the pair of functions $u(t,x)$ and $v(t,x):=\frac{\partial u}{\partial x}(t,x)$ satisfy the Cauchy problem for two parabolic equations consisting of (2.3)–(2.4) and

$$\begin{aligned}\frac{\partial v}{\partial t}+b(t,x,u)\frac{\partial v}{\partial x}+\frac{1}{2}\sigma^2(t,x,u)\frac{\partial^2 v}{\partial x^2}+\left(\sigma\frac{\partial\sigma}{\partial x}+\sigma\frac{\partial\sigma}{\partial u}v\right)\frac{\partial v}{\partial x}\\ +\left(\frac{\partial b}{\partial x}+\frac{\partial b}{\partial u}v\right)v+\frac{\partial g}{\partial x}+\frac{\partial g}{\partial u}v=0,\quad t<T,\ x\in\mathbf{R},\end{aligned} \tag{3.4}$$

$$v(T,x)=\varphi'(x)\,. \tag{3.5}$$

To solve the problem (2.3)–(2.4), (3.4)–(3.5), we use a layer method based on a local probabilistic representation. To this aim, introduce the system of SDEs with respect to X and scalars P, Q, R:

$$\begin{aligned}&dX=b(s,X,u(s,X))ds+\sigma(s,X,u(s,X))dw(s),\quad X(t_k)=x,\\ &dP=g(s,X,u(s,X))ds,\quad P(t_k)=0,\\ &dQ=\left(\frac{\partial b}{\partial x}+\frac{\partial b}{\partial u}v(s,X)\right)Qds+\left(\frac{\partial\sigma}{\partial x}+\frac{\partial\sigma}{\partial u}v(s,X)\right)Qdw(s),\quad Q(t_k)=1,\\ &dR=\left(\frac{\partial g}{\partial x}+\frac{\partial g}{\partial u}v(s,X)\right)Qds,\quad R(t_k)=0,\end{aligned} \tag{3.6}$$

where $\partial b/\partial x$, $\partial b/\partial u$, and the other derivatives are known functions of s, X, $u(s,X)$. One can verify that the following local probabilistic representation holds (cf. (2.5)–(2.6)):

$$\begin{aligned}u(t_k,x)&=E\left[u(t_{k+1},X_{t_k,x}(t_{k+1}))+P_{t_k,x,0}(t_{k+1})\right],\\ v(t_k,x)&=E\left[v(t_{k+1},X_{t_k,x}(t_{k+1}))Q_{t_k,x,1}(t_{k+1})+R_{t_k,x,1,0}(t_{k+1})\right].\end{aligned} \tag{3.7}$$

The corresponding layer method $\tilde{u}(t_k,x)$, $\tilde{v}(t_k,x)$ is given by the formulas (2.7) for $\tilde{u}$ while $\tilde{v}$ is found from

$$\tilde{v}(t_N,x)=\varphi'(x), \tag{3.8}$$

$$\begin{aligned}\tilde{v}(t_k,x)=&\frac{1}{2}\tilde{v}(t_{k+1},x+h\tilde{b}_k-\sqrt{h}\tilde{\sigma}_k)\\ &\times\left[1+h\left(\frac{\partial\tilde{b}_k}{\partial x}+\frac{\partial\tilde{b}_k}{\partial u}\tilde{v}(t_{k+1},x)\right)-h^{1/2}\left(\frac{\partial\tilde{\sigma}_k}{\partial x}+\frac{\partial\tilde{\sigma}_k}{\partial u}\tilde{v}(t_{k+1},x)\right)\right]\\ &+\frac{1}{2}\tilde{v}(t_{k+1},x+h\tilde{b}_k+\sqrt{h}\tilde{\sigma}_k)\\ &\times\left[1+h\left(\frac{\partial\tilde{b}_k}{\partial x}+\frac{\partial\tilde{b}_k}{\partial u}\tilde{v}(t_{k+1},x)\right)+h^{1/2}\left(\frac{\partial\tilde{\sigma}_k}{\partial x}+\frac{\partial\tilde{\sigma}_k}{\partial u}\tilde{v}(t_{k+1},x)\right)\right]\\ &+h\left(\frac{\partial\tilde{g}_k}{\partial x}+\frac{\partial\tilde{g}_k}{\partial u}\tilde{v}(t_{k+1},x)\right),\quad k=N-1,\dots,1,0,\end{aligned}$$

where $\tilde{b}_k := b(t_k, x, \tilde{u}(t_{k+1}, x))$, $\tilde{\sigma}_k := \sigma(t_k, x, \tilde{u}(t_{k+1}, x))$, and the notation $\partial\tilde{b}_k/\partial x$ means $\frac{\partial\tilde{b}_k}{\partial x} := \frac{\partial b}{\partial x}(t_k, x, \tilde{u}(t_{k+1}, x))$ and so on. The layer method (2.7), (3.8) for the system (2.3)–(2.4), (3.4)–(3.5) is of order one. The order of convergence can be proved due to [18] if we assume that the functions from Assumption 2.1 have continuous bounded second mixed derivatives with respect to t, x and t, u and third derivatives with respect to x and u including the mixed ones. This also implies that $\tilde{u}(t_k, x)$ has, in particular, a continuous bounded third derivative with respect to x.

Further, it is straightforward to verify

$$\frac{\partial\tilde{u}}{\partial x}(t_k, x) = \tilde{v}(t_k, x).$$

Thus we get

$$\frac{\partial u}{\partial x}(t_k, x) = v(t_k, x) = \tilde{v}(t_k, x) + O(h) = \frac{\partial\tilde{u}}{\partial x}(t_k, x) + O(h). \tag{3.9}$$

Since $\tilde{u}(t_k, x)$ has a bounded third derivative with respect to x, we obtain

$$\frac{\partial\tilde{u}}{\partial x}(t_k, x) = \frac{\tilde{u}(t_k, x + \gamma\sqrt{h}) - \tilde{u}(t_k, x - \gamma\sqrt{h})}{2\gamma\sqrt{h}} + O(h). \tag{3.10}$$

The formulas (3.9) and (3.10) imply (3.3). □

REMARK 3.4. We have not succeeded in a rigorous proof of the relation

$$\frac{\partial u}{\partial x}(t_k, x) = \frac{\bar{u}(t_k, x + \gamma\sqrt{h}) - \bar{u}(t_k, x - \gamma\sqrt{h})}{2\gamma\sqrt{h}} + O(h) \tag{3.11}$$

for $\bar{u}(t_k, x)$ from Algorithm 2.2. At the same time, we have some heuristic arguments justifying (3.11). It is possible to obtain the following representation for $\bar{u}(t_0, x)$:

$$\bar{u}(t_0, x) = \tilde{u}(t_0, x) + R(x, h)h + \sum_{k=1}^{N} \zeta_k h^2 + O(h^{3/2}). \tag{3.12}$$

Moreover, it is reasonable to consider $R(x, h)$ a function which changes in x slowly or, more exactly, that R satisfies the inequality

$$R(x + \gamma\sqrt{h}, h) - R(x - \gamma\sqrt{h}, h) = O(\sqrt{h}). \tag{3.13}$$

In (3.12) ζ_k are related to the distances between $x_j + hb(t_k, x_j, \bar{u}(t_{k+1}, x_j)) \pm \sqrt{h}\sigma(t_k, x_j, \bar{u}(t_{k+1}, x_j))$ and the nearest node x_i. These distances are random, in a sense, and it is natural to assume that ζ_k are independent and identically distributed (i.i.d.) uniformly bounded random variables with zero mean and variance $Var\zeta$. Then, due to the central limit theorem, we get

$$\sum_{k=1}^{N} \zeta_k h^2 \doteq \eta \sqrt{Var\zeta} \cdot \sqrt{N}\, h^2, \tag{3.14}$$

where η is a standard Gaussian random variable. The relation (3.11) follows from (3.12)–(3.14). We also note here in passing that if in Algorithm 2.2 we would put $h_x = \varkappa h^{5/4}$ instead of $h_x = \varkappa h$, then it would not be difficult to prove (3.11) rigorously.

In the multidimensional case ((1.3)–(1.4) with $d > 1$) we use the approximation

(3.15)
$$\frac{\partial u}{\partial x^i}(t_k, x) \approx \frac{\bar{u}(t_k, x^1, \dots, x^i + \gamma\sqrt{h}, \dots, x^d) - \bar{u}(t_k, x^1, \dots, x^i - \gamma\sqrt{h}, \dots, x^d)}{2\gamma\sqrt{h}},$$
$$i = 1, \dots, d,$$

where $\bar{u}$ is found by a multidimensional algorithm analogous to Algorithm 2.2 [14, 18]. Proposition 3.1 and Remark 3.4 are valid for (3.15) as well.

4. Numerical integration of FBSDEs. Let $\bar{u}(t_k, x)$ be defined by Algorithm 2.2 and introduce the notation

$$\frac{\Delta\bar{u}}{\Delta x}(t_k, x) := \frac{\bar{u}(t_k, x + \gamma\sqrt{h}) - \bar{u}(t_k, x - \gamma\sqrt{h})}{2\gamma\sqrt{h}} \quad \text{for some } \gamma > 0 \tag{4.1}$$

and also $\Delta_k w := w(t_k + h) - w(t_k)$. In practice it is advisable to choose the parameter γ close to the diffusion σ at the point (t_k, x).

Consider two numerical schemes for the FBSDE (1.1)–(1.2) with $d = n = 1$:

(4.2)
$$X_0 = x,$$
$$X_{k+1} = X_k + a(t_k, X_k, \bar{u}(t_k, X_k))h + \sigma(t_k, X_k, \bar{u}(t_k, X_k))\Delta_k w, \quad k = 0, \dots, N-1,$$

and

(4.3)
$$X_0 = x,$$
$$X_{k+1} = X_k + a(t_k, X_k, \bar{u}(t_k, X_k))h + \sigma(t_k, X_k, \bar{u}(t_k, X_k))\Delta_k w$$
$$+ \frac{1}{2}\sigma(t_k, X_k, \bar{u}(t_k, X_k)) \left(\frac{\partial\sigma}{\partial x}(t_k, X_k, \bar{u}(t_k, X_k)) \right.$$
$$\left. + \frac{\partial\sigma}{\partial u}(t_k, X_k, \bar{u}(t_k, X_k)) \frac{\Delta\bar{u}}{\Delta x}(t_k, X_k) \right) \times \left(\Delta_k^2 w - h\right), \quad k = 0, \dots, N-1;$$

Y and Z of the solution to (1.1)–(1.2) are approximated as

$$Y_k = \bar{u}(t_k, X_k), \quad Z_k = \sigma(t_k, X_k, Y_k) \frac{\Delta\bar{u}}{\Delta x}(t_k, X_k), \tag{4.4}$$

where X_k is either from (4.2) or (4.3).

Let us note that the first-order scheme (4.3) becomes the Euler scheme in the case of additive noise in (1.1).

THEOREM 4.1. (i) (4.2) (4.4) 1/2.

(4.5)
$$\left[E \left[(X(t_k) - X_k)^2 + (Y(t_k) - Y_k)^2 + (Z(t_k) - Z_k)^2 \right] \right]^{1/2} \le K (1 + x^2)^{1/2} h^{1/2},$$

K x, k, h

(ii) (4.3) (4.4) X Y.

$$\left[E \left[(X(t_k) - X_k)^2 + (Y(t_k) - Y_k)^2 \right] \right]^{1/2} \le K (1 + x^2)^{1/2} h, \tag{4.6}$$

$$\left|\frac{\Delta \bar{u}}{\Delta x}(t_k, x) - \frac{\partial u}{\partial x}(t_k, x)\right| \le Ch, \tag{4.7}$$

$$\left[E\left(Z(t_k) - Z_k\right)^2\right]^{1/2} \le K\left(1 + x^2\right)^{1/2} h, \tag{4.8}$$

K x, k, h

With respect to the assumption (4.7), see Remark 3.4.

(i) Let us prove that the Euler scheme satisfies

$$\left[E\left(X(t_k) - X_k\right)^2\right]^{1/2} \le K\left(1 + x^2\right)^{1/2} h^{1/2}. \tag{4.9}$$

Assume for a while that the solution $u(t, x)$ to (1.3)–(1.4) is known exactly. Then the coefficients in (1.1) are known functions of t and x and we can apply the standard mean-square Euler scheme

$$\hat{X}_{k+1} = \hat{X}_k + a(t_k, \hat{X}_k, u(t_k, \hat{X}_k))h + \sigma(t_k, \hat{X}_k, u(t_k, \hat{X}_k))\Delta_k w, \quad k = 0, \ldots, N-1, \tag{4.10}$$

which is of mean-square order $1/2$; i.e., $\hat{X}_k$ from (4.10) satisfies a relation like (4.9).

Now we compare X_k and $\hat{X}_k$. To this end, we exploit the (see [13] or [18, p. 4]). It states that if a one-step approximation $\bar{X}_{t,x}(t+h)$ of the solution $X_{t,x}(t+h)$ satisfies the conditions

$$|E(X_{t,x}(t+h) - \bar{X}_{t,x}(t+h))| \le K(1 + |x|^2)^{1/2} h^{p_1}, \tag{4.11}$$

$$\left[E|X_{t,x}(t+h) - \bar{X}_{t,x}(t+h)|^2\right]^{1/2} \le K(1 + |x|^2)^{1/2} h^{p_2} \tag{4.12}$$

with $p_2 > 1/2$ and $p_1 \ge p_2 + 1/2$, then the corresponding mean-square method X_k has order of convergence $p_2 - 1/2$; i.e.,

$$\left[E\left(X(t_k) - X_k\right)^2\right]^{1/2} \le K\left(1 + x^2\right)^{1/2} h^{p_2 - 1/2}.$$

Introduce the one-step approximations corresponding to X_k and $\hat{X}_k$:

$$X(t+h) \approx \bar{X} = x + a(t, x, \bar{u}(t, x))h + \sigma(t, x, \bar{u}(t, x))\Delta w \tag{4.13}$$

and

$$X(t+h) \approx \hat{X} = x + a(t, x, u(t, x))h + \sigma(t, x, u(t, x))\Delta w. \tag{4.14}$$

It is known [13, 18] that $\hat{X}$ from (4.14) satisfies (4.11) with $p_1 = 2$ and (4.12) with $p_2 = 1$. Due to Assumption 2.1 and the relation (2.11), we get

$$E(\hat{X} - \bar{X}) = a(t, x, u(t, x))h - a(t, x, \bar{u}(t, x))h = O(h^2)$$

and

$$E(\hat{X}-\bar{X})^2 = \left[a(t,x,u(t,x)) - a(t,x,\bar{u}(t,x))\right]^2 h^2 + \left[\sigma(t,x,u(t,x)) - \sigma(t,x,\bar{u}(t,x))\right]^2 h = O(h^3),$$

whence it follows that $\bar{X}$ from (4.13) also satisfies (4.11) with $p_1 = 2$ and (4.12) with $p_2 = 1$. Then, applying the fundamental convergence theorem, we prove (4.9). Now the rest of (4.5) follows from (2.2), (4.9), (2.11), and (3.1).

(ii) To prove (4.6), we just repeat all the arguments as above. The only difference is that this time we compare the one-step approximation corresponding to (4.3) with the one corresponding to the standard first-order mean-square scheme [18] applied to (1.1), assuming again that $u(t,x)$ is known exactly. We note that the estimate (3.1) is enough to obtain (4.6) but it would imply the mean-square order 1/2 for Z instead of (4.8). At the same time, it is clear that by (2.2), (4.6) together with (4.7) we obtain (4.8). □

Remark 4.2. Theorem 4.1 is also valid for other Euler-type methods (e.g., for the implicit Euler method) as well as for other first-order mean-square methods. Further, in the case of additive noise this theorem can be extended to constructive mean-square methods of order 3/2 [18] using second-order methods from [14, 18] for the semilinear parabolic problem (2.3)–(2.4).

Remark 4.3. Let us consider the weak Euler scheme

$$X_0 = x,$$
$$X_{k+1} = X_k + a(t_k, X_k, \bar{u}(t_k, X_k))h + \sigma(t_k, X_k, \bar{u}(t_k, X_k))\xi_k, \quad k = 0, \dots, N-1, \tag{4.15}$$

where ξ_k, $k = 0, \dots, N-1$, are i.i.d. random variables with the distribution $P(\xi = \pm 1) = 1/2$ and $\bar{u}(t_k, x)$ is defined by Algorithm 2.2. It is possible to prove that for a sufficiently smooth function $F(x,y)$ satisfying some boundedness conditions, the Euler method (4.15) and (4.4) is of weak order one; i.e.,

$$EF(X(t_k), Y(t_k)) - EF(X_k, Y_k) = O(h).$$

The proof is based on the main theorem on convergence of weak approximations [18, p. 100].

Remark 4.4. It is not difficult to generalize both the numerical algorithm considered and Theorem 4.1 to the case $d > 1$ using the multidimensional version of the Euler method (4.2) and (4.4), a multidimensional algorithm analogous to Algorithm 2.2 (see [14, 18]) to solve (1.3)–(1.4), and (3.15) to approximate the derivatives.

5. FBSDEs with random terminal time.

5.1. The parabolic case. Let G be a bounded domain in $\mathbf{R}^d$, let $Q = [t_0, T) \times G$ be a cylinder in $\mathbf{R}^{d+1}$, and let $\Gamma = \overline{Q} \setminus Q$. The set Γ is a part of the boundary of the cylinder Q consisting of the upper base and the lateral surface. Let $\varphi(t,x)$ be a function defined on Γ.

Consider the FBSDE with random terminal time (see e.g., [19, 25]):

$$dX = a(t,X,Y)dt + \sigma(t,X,Y)dw(t), \quad X(t_0) = x \in G, \tag{5.1}$$
$$dY = -g(t,X,Y)dt - f^{\intercal}(t,X,Y)Zdt + Z^{\intercal}dw(t), \quad Y(\tau) = \varphi(\tau, X(\tau)), \tag{5.2}$$

where $\tau = \tau_{t_0,x}$ is the first exit time of the trajectory $(t, X_{t_0,x}(t))$ from the domain Q; i.e., the point $(\tau, X(\tau))$ belongs to Γ. A solution to (5.1)–(5.2) is defined as an $\{\mathcal{F}_t\}_{t\geq 0}$-adapted vector $(X(t), Y(t), Z(t))$ together with the Markov moment τ, which satisfy

(5.1)–(5.2). This solution is connected with the Dirichlet boundary value problem for the semilinear parabolic equation

$$\frac{\partial u}{\partial t} + \sum_{i=1}^{d} a^i(t,x,u)\frac{\partial u}{\partial x^i} + \frac{1}{2}\sum_{i,j=1}^{d} a^{ij}(t,x,u)\frac{\partial^2 u}{\partial x^i \partial x^j} = -g(t,x,u) - \sum_{k=1}^{n} f^k(t,x,u)\sum_{i=1}^{d}\sigma^{ik}(t,x,u)\frac{\partial u}{\partial x^i}, \quad (t,x)\in Q, \tag{5.3}$$

$$u(t,x)|_{\Gamma} = \varphi(t,x). \tag{5.4}$$

Let $u(t,x)$ be the solution of (5.3)-(5.4), which is supposed to exist, be unique, and be sufficiently smooth. One can find many theoretical results on this topic in [10] (see also references therein and in [16, 18]). To be definite, we assume here that the conditions like Assumption 2.1 together with sufficient smoothness of the boundary ∂G and of the function φ are fulfilled.

Consider the following SDE in Q:

$$dX = a(t,X,u(t,X))dt + \sigma(t,X,u(t,X))dw(t), \quad X(t_0) = x, \tag{5.5}$$

with random terminal time τ which is defined as the first exit time of the trajectory $(t, X_{t_0,x}(t))$ of (5.5) from the domain Q. Introduce

$$\begin{aligned} Y(t) &= u(t, X_{t_0,x}(t)), \quad t_0 \le t \le \tau, \\ Z^j(t) &= \sum_{i=1}^{d}\sigma^{ij}(t, X_{t_0,x}(t), Y(t))\frac{\partial u}{\partial x^i}(t, X_{t_0,x}(t)), \quad j = 1,\dots,n, \quad t_0 \le t \le \tau. \end{aligned} \tag{5.6}$$

Clearly, the four-tuple $(X_{t_0,x}(t), Y(t), Z(t), \tau)$ is a solution of (5.1)–(5.2).

In what follows we restrict ourselves to the one-dimensional version of (5.1)–(5.2) ($d = 1$, $n = 1$). Introducing

$$b(t,x,y) := a(t,x,y) + f(t,x,y)\sigma(t,x,y),$$

we get

$$\frac{\partial u}{\partial t} + b(t,x,u)\frac{\partial u}{\partial x} + \frac{1}{2}\sigma^2(t,x,u)\frac{\partial^2 u}{\partial x^2} + g(t,x,u) = 0, \quad (t,x)\in Q, \tag{5.7}$$

$$u(t,x)|_{\Gamma} = \varphi(t,x). \tag{5.8}$$

In this case Q is the partly open rectangle $Q = [t_0, T)\times(\alpha,\beta)$, and Γ consists of the upper base $\{T\}\times[\alpha,\beta]$ and two vertical intervals, $[t_0,T)\times\{\alpha\}$ and $[t_0,T)\times\{\beta\}$.

In [16] (see also [18]) we propose a number of algorithms for solving the problem (5.7)–(5.8). As an example, let us recall one of them. Consider an equidistant space discretization with a space step h_x (recall that the notation for time step is h): $x_j = \alpha + jh_x$, $j = 0,1,2,\dots,M$, $h_x = (\beta-\alpha)/M$. The algorithm has the following form.

ALGORITHM 5.1.

$$\bar{u}(t_N, x) = \varphi(t_N, x), \quad x\in[\alpha,\beta], \tag{5.9}$$

$$\bar{u}(t_k, x_j) = \frac{1}{2}\bar{u}(t_{k+1}, x_j + \bar{b}_{k,j} \cdot h - \bar{\sigma}_{k,j} \cdot \sqrt{h}) + \frac{1}{2}\bar{u}(t_{k+1}, x_j + \bar{b}_{k,j} \cdot h + \bar{\sigma}_{k,j} \cdot \sqrt{h})$$
$$+ \bar{g}_{k,j} \cdot h \quad \text{if } x_j + \bar{b}_{k,j} \cdot h \pm \bar{\sigma}_{k,j} \cdot \sqrt{h} \in [\alpha, \beta];$$

$$\bar{u}(t_k, x_j) = \frac{1}{1 + \sqrt{\bar{\lambda}_{k,j}}}\varphi(t_{k+\bar{\lambda}_{k,j}}, \alpha) + \frac{\sqrt{\bar{\lambda}_{k,j}}}{1 + \sqrt{\bar{\lambda}_{k,j}}}\bar{u}(t_{k+1}, x_j + \bar{b}_{k,j} \cdot h + \bar{\sigma}_{k,j} \cdot \sqrt{h})$$
$$+ \bar{g}_{k,j} \cdot \sqrt{\bar{\lambda}_{k,j}} h \quad \text{if } x_j + \bar{b}_{k,j} \cdot h - \bar{\sigma}_{k,j} \cdot \sqrt{h} < \alpha;$$

$$\bar{u}(t_k, x_j) = \frac{1}{1 + \sqrt{\bar{\mu}_{k,j}}}\varphi(t_{k+\bar{\mu}_{k,j}}, \beta) + \frac{\sqrt{\bar{\mu}_{k,j}}}{1 + \sqrt{\bar{\mu}_{k,j}}}\bar{u}(t_{k+1}, x_j + \bar{b}_{k,j} \cdot h - \bar{\sigma}_{k,j} \cdot \sqrt{h})$$
$$+ \bar{g}_{k,j} \cdot \sqrt{\bar{\mu}_{k,j}} h, \text{ if } x_j + \bar{b}_{k,j} \cdot h + \bar{\sigma}_{k,j} \cdot \sqrt{h} > \beta,$$
$$j = 1, 2, \ldots, M-1;$$

$$\bar{u}(t_k, x) = \frac{x_{j+1} - x}{h_x}\,\bar{u}(t_k, x_j) + \frac{x - x_j}{h_x}\,\bar{u}(t_k, x_{j+1}), \quad x_j < x < x_{j+1}, \tag{5.10}$$
$$j = 0, 1, 2, \ldots, M-1, \; k = N-1, \ldots, 1, 0,$$

$\bar{b}_{k,j}$, $\bar{\sigma}_{k,j}$, $\bar{g}_{k,j}$ [illegible] $b(t,x,u)$, $\sigma(t,x,u)$, $g(t,x,u)$ [illegible] $(t_k, x_j, \bar{u}(t_{k+1}, x_j))$, $t_{k+\bar{\lambda}_{k,j}} := t_k + h\bar{\lambda}_{k,j}$, $t_{k+\bar{\mu}_{k,j}} := t_k + h\bar{\mu}_{k,j}$, [illegible] $0 < \bar{\lambda}_{k,j}, \bar{\mu}_{k,j} \le 1$ [illegible]

$$\alpha = x_j + \bar{b}_{k,j} \cdot \bar{\lambda}_{k,j} h - \bar{\sigma}_{k,j} \cdot \sqrt{\bar{\lambda}_{k,j} h}, \qquad \beta = x_j + \bar{b}_{k,j} \cdot \bar{\mu}_{k,j} h + \bar{\sigma}_{k,j} \cdot \sqrt{\bar{\mu}_{k,j} h}.$$

It is proved in [16] (see also [18, p. 475]) that if the value of h_x is taken equal to $\varkappa h$ with $\varkappa$ being a positive constant, then

$$|\bar{u}(t_k, x) - u(t_k, x)| \le Kh\,, \tag{5.11}$$

where K does not depend on x, h, k.

To construct Z due to (5.6), we need an approximation of $\partial u/\partial x$. To this end, we propose to use the formulas (cf. (3.1))

$$\frac{\partial u}{\partial x}(t_k, x) = \frac{\bar{u}(t_k, x + \gamma\sqrt{h}) - \bar{u}(t_k, x - \gamma\sqrt{h})}{2\gamma\sqrt{h}} + O(h^{1/2}) := \frac{\Delta\bar{u}}{\Delta x}(t_k, x) + O(h^{1/2}) \tag{5.12}$$
$$\text{if } x \pm \gamma\sqrt{h} \in [\alpha, \beta],$$

$$\frac{\partial u}{\partial x}(t_k,x)=\frac{4\bar u(t_k,x+\gamma\sqrt{h})-3\bar u(t_k,x)-\bar u(t_k,x+2\gamma\sqrt{h})}{2\gamma\sqrt{h}}+O(h^{1/2}) \tag{5.13}$$
$$:=\frac{\Delta\bar u}{\Delta x}(t_k,x)+O(h^{1/2})$$
$$\text{if } x-\gamma\sqrt{h}<\alpha,$$

and

$$\frac{\partial u}{\partial x}(t_k,x)=\frac{\bar u(t_k,x-2\gamma\sqrt{h})-4\bar u(t_k,x-\gamma\sqrt{h})+3\bar u(t_k,x)}{2\gamma\sqrt{h}}+O(h^{1/2}) \tag{5.14}$$
$$:=\frac{\Delta\bar u}{\Delta x}(t_k,x)+O(h^{1/2})$$
$$\text{if } x+\gamma\sqrt{h}>\beta.$$

Most probably, the accuracy in (5.12)–(5.14) is $O(h)$ rather than $O(h^{1/2})$, but we have not investigated this issue in detail.

Let us note that if we apply the method of differentiation to the boundary value problem (5.7)–(5.8), we obtain (3.4) for $v=\partial u/\partial x$ in Q and the Neumann boundary condition. Namely, this boundary condition is of the form

$$\text{on the upper base of } Q\text{: } v(T,x)=\frac{\partial\varphi}{\partial x}(T,x)$$

and, for example, on the vertical interval $[t_0,T]\times\{\alpha\}$:

$$b(t,\alpha,\varphi(t,\alpha))v(t,\alpha)+\frac{1}{2}\sigma^2(t,\alpha,\varphi(t,\alpha))\frac{\partial v}{\partial x}(t,\alpha)=-\frac{\partial\varphi}{\partial t}(t,\alpha)-g(t,\alpha,\varphi(t,\alpha)).$$

Thus, in the case of FBSDEs with random terminal time the approach of [6] leads to a complicated system of boundary value problems.

5.2. The elliptic case. The random terminal time in FBSDE (5.1)–(5.2) is bounded by the time T. Now we consider FBSDEs with unbounded random terminal time. Let G be a bounded domain in $\mathbf{R}^d$ and $Q=[0,\infty)\times G$ be a cylinder in $\mathbf{R}^{d+1}$.

Consider the FBSDE with random terminal time (see e.g., [19, 25]):

$$dX=a(X,Y)dt+\sigma(X,Y)dw(t),\quad X(0)=x\in G, \tag{5.15}$$
$$dY=-g(X,Y)dt-f^{\intercal}(X,Y)Zdt+Z^{\intercal}dw(t),\quad Y(\tau)=\varphi(X(\tau)), \tag{5.16}$$

where $\tau=\tau_x$ is the first exit time of the trajectory $X_x(t)$ from the domain G; i.e., the point $X(\tau)$ belongs to the boundary ∂G of G. A solution to (5.15)–(5.16) is defined as an $\{\mathcal{F}_t\}_{t\ge0}$-adapted vector $(X(t),Y(t),Z(t))$ together with the Markov moment τ, which satisfy (5.15)–(5.16). This solution is connected with the Dirichlet boundary value problem for the semilinear elliptic equation

$$\sum_{i=1}^{d}a^i(x,u)\frac{\partial u}{\partial x^i}+\frac{1}{2}\sum_{i,j=1}^{d}a^{ij}(x,u)\frac{\partial^2u}{\partial x^i\partial x^j} \tag{5.17}$$
$$=-g(x,u)-\sum_{k=1}^{n}f^k(x,u)\sum_{i=1}^{d}\sigma^{ik}(x,u)\frac{\partial u}{\partial x^i},\quad x\in G,$$

$$u(x)|_{\partial G}=\varphi(x). \tag{5.18}$$

Let $u(x)$ be the solution of (5.17)–(5.18), which is supposed to exist, be unique, and be sufficiently smooth. Consider the following SDE in G:

$$dX = a(X, u(X))dt + \sigma(X, u(X))dw(t), \quad X(0) = x, \tag{5.19}$$

with random terminal time τ which is defined as the first exit time of the trajectory $X_x(t)$ of (5.19) from the domain G. Introduce

$$Y(t) = u(X_x(t)), \quad 0 \le t \le \tau, \tag{5.20}$$
$$Z^j(t) = \sum_{i=1}^{d} \sigma^{ij}(X_x(t), Y(t)) \frac{\partial u}{\partial x^i}(X_x(t)), \quad j = 1, \dots, n, \ 0 \le t \le \tau.$$

Clearly, the four-tuple $(X_x(t), Y(t), Z(t), \tau)$ is a solution of (5.15)–(5.16). Note that in the one-dimensional case the problem (5.17)–(5.18) is the boundary value problem just for a second-order ordinary differential equation whose numerical solution does not cause any problem. To solve (5.17)–(5.18) for $d > 1$, one can use finite-difference methods or apply a multidimensional layer method analogous to (5.9)–(5.10) (see [16, 18]) using ideas of the relaxation method.

6. Numerical integration of FBSDEs with random terminal time. Trajectories $(t, X(t))$ of the SDE (5.1) belong to the space-time bounded domain $\bar{Q}$, and a corresponding approximation (ϑ_k, X_k) should possess the same property. It is obvious that, e.g., the standard Euler scheme (4.2) does not satisfy this requirement and specific methods are needed. Such approximations were proposed in [15] (see also [18]). In the one-dimensional case they take a simpler form, which is presented below. We note that here the notation differs partly from that used in [15, 18].

Consider the one-dimensional SDE

$$dX = \chi_{\tau_{t,x}>s} b(s, X)ds + \chi_{\tau_{t,x}>s}\sigma(s, X)dw(s), \quad X(t) = X_{t,x}(t) = x, \tag{6.1}$$

in a space-time bounded domain $Q = [t_0, T) \times (\alpha, \beta)$; the Markov moment $\tau_{t,x}$ is the first-passage time of the process $(s, X_{t,x}(s))$, $s \ge t$, to $\Gamma = \bar{Q} \setminus Q$.

Let $I_r := [-r, r]$, $r > 0$, $\Pi := [0, l) \times I_1$ for some $l > 0$, and $\Pi_h := [0, lh) \times I_{\sqrt{h}}$. Take a point $(s, y) \in Q$ and introduce another interval $I(s, y; h) := [x + hb(s, y) - \sigma(s, y)\sqrt{h}, x + hb(s, y) + \sigma(s, y)\sqrt{h}]$ and also the space-time rectangle $\Pi(s, y; h) = [s, s + lh) \times I(s, y; h)$. Let Γ_δ be an intersection of a δ-neighborhood of the set Γ with the domain Q. Below we take δ equal to $\lambda h^{(1-\varepsilon)/2}$ with $0 < \varepsilon \le 1$ and $\lambda = 2 \max\limits_{(s,y)\in\bar{Q}} |\sigma(s, y)|$. Now we construct a random walk over small space-time rectangles.

Algorithm 6.1 (random walk over small space-time rectangles). Choose a time step $h > 0$ and numbers $0 < \varepsilon \le 1$ and $L > 0$.

Step 0. $X_0 = x$, $\vartheta_0 = t$, $(t, x) \in Q$, $k = 0$.

Step 1. If $(\vartheta_k, X_k) \in \Gamma_{\lambda h^{(1-\varepsilon)/2}}$ or $k \ge L/h$, then Stop and
(i) put $\nu = k$, $(\vartheta_\nu, X_\nu) = (\vartheta_k, X_k)$;
(ii) if $\vartheta_\nu \ge T - \lambda h^{(1-\varepsilon)/2}$, then $\bar{\tau}_{t,x} = T$ and $\xi_{t,x} = X_\nu \in (\alpha, \beta)$; otherwise $\bar{\tau}_{t,x} = \vartheta_\nu$ and $\xi_{t,x}$ is the end of the interval $[\alpha, \beta]$ nearest to X_ν.

Step 2. Put $k := k + 1$. Simulate the first exit point $(\theta_k, w(\vartheta_{k-1} + \theta_k) - w(\vartheta_{k-1}))$ of the process $(s - \vartheta_{k-1}, w(s) - w(\vartheta_{k-1}))$, $s > \vartheta_{k-1}$, from the rectangle Π_h. Put

$$\vartheta_k = \vartheta_{k-1} + \theta_k, \tag{6.2}$$
$$X_k = X_{k-1} + b(\vartheta_{k-1}, X_{k-1})\theta_k + \sigma(\vartheta_{k-1}, X_{k-1})(w(\vartheta_k) - w(\vartheta_{k-1})). \tag{6.3}$$

Go to Step 1.

The sequence (ϑ_k, X_k) obtained by Algorithm 6.1 is a Markov chain stopping at the Markov moment ν in the neighborhood $\Gamma_{\lambda h^{(1-\varepsilon)/2}}$ of the boundary Γ. At each step $(\vartheta_k, X_k) \in \partial\Pi(\vartheta_{k-1}, X_{k-1}; h)$ and $\overline{\Pi}(\vartheta_{k-1}, X_{k-1}; h) \subset Q$; i.e., the chain belongs to the space-time bounded domain Q with probability one. The simulated points (ϑ_k, X_k) are close in the mean-square sense to $(\vartheta_k, X(\vartheta_k))$, and the point $(\bar{\tau}_{t,x}, \xi_{t,x})$ is close to $(\tau_{t,x}, X(\tau_{t,x}))$. It is proved in [15, 18] that

$$\left(E\,|X(\vartheta_k) - X_k|^2\right)^{1/2} \le K\ (\sqrt{h} + e^{-c_h L}), \quad k = 1, \dots, \nu, \tag{6.4}$$

$$E|\tau_{t,x} - \bar{\tau}_{t,x}| \le K\ (h^{(1-\varepsilon)/2} + e^{-c_h L}),$$

$$\left(E\left|X_{t,x}(\tau_{t,x}) - \xi_{t,x}\right|^2\right)^{1/2} \le K\ (\,h^{(1-\varepsilon)/4} + e^{-c_h L/2}),$$

where the constant K is independent of h, k, t, x and c_h tends to a positive constant independent of L as $h \to 0$. We note that the accuracy of the algorithm depends on the choice of h, ε, and L. Clearly, we reach higher accuracy by decreasing h and/or ε and increasing L.

Algorithm 6.1 in its turn requires an algorithm, which is considered below, for simulating the first exit point $(\theta, w(\theta))$ of the process $(s, w(s))$, $s > 0$, from the rectangle Π_h.

Let $W(s)$ be a one-dimensional standard Wiener process and let τ be the first exit time of $W(s)$ from the interval $I_1 = [-1, 1]$. Then the following formulas for the distribution and density of τ take place:

$$\mathcal{P}(t) = 1 - \frac{4}{\pi}\sum_{k=0}^{\infty}\frac{(-1)^k}{2k+1}\cdot\exp\left(-\frac{1}{8}\pi^2(2k+1)^2 t\right), \quad t > 0, \tag{6.5}$$

and

$$\mathcal{P}(t) = 2\sum_{k=0}^{\infty}(-1)^k \operatorname{erfc}\frac{2k+1}{\sqrt{2t}}, \quad t > 0, \qquad \operatorname{erfc} x = \frac{2}{\sqrt{\pi}}\int_x^{\infty}\exp(-y^2)dy\,; \tag{6.6}$$

$$\mathcal{P}'(t) = \frac{\pi}{2}\sum_{k=0}^{\infty}(-1)^k(2k+1)\exp\left(-\frac{1}{8}\pi^2(2k+1)^2 t\right), \quad t > 0, \tag{6.7}$$

and

$$\mathcal{P}'(t) = \frac{2}{\sqrt{2\pi t^3}}\sum_{k=0}^{\infty}(-1)^k(2k+1)\exp\left(-\frac{1}{2t}(2k+1)^2\right), \quad t > 0. \tag{6.8}$$

The formulas (6.5) and (6.7) are suitable for calculations under big t, and the formulas (6.6) and (6.8) are suitable for small t. See further computational details in [15, 18].

For the conditional probability

$$\mathcal{Q}(\mu; t) := P(W(t) < \mu \,/\, |W(s)| < 1,\ 0 < s < t)\,,$$

where $-1 < \mu \le 1$, the following equalities hold [15, 18]:

$$\mathcal{Q}(\mu; t) = \frac{P(W(t) < \mu\,,\ \tau \ge t)}{P(\tau \ge t)} \tag{6.9}$$

$$= \frac{1}{1 - \mathcal{P}(t)}\cdot\frac{2}{\pi}\sum_{k=0}^{\infty}\frac{1}{2k+1}\left((-1)^k + \sin\frac{\pi(2k+1)\mu}{2}\right)\exp\left(-\frac{1}{8}\pi^2(2k+1)^2 t\right)$$

and

$$\mathcal{Q}(\mu;t)=\frac{1}{1-\mathcal{P}(t)} \times\sum_{k=0}^{\infty}\frac{(-1)^k}{2}\left(\operatorname{erfc}\frac{2k-1}{\sqrt{2t}}-\operatorname{erfc}\frac{2k+\mu}{\sqrt{2t}}-\operatorname{erfc}\frac{2k+2-\mu}{\sqrt{2t}}+\operatorname{erfc}\frac{2k+3}{\sqrt{2t}}\right). \tag{6.10}$$

Note that the series (6.9) and (6.10) are of the Leibniz type, the formula (6.9) is convenient for calculations under big t, and the formula (6.10) is convenient for small t. We draw attention to the denominator $(1-\mathcal{P}(t))$ in (6.9) which is close to zero for $t\gg 1$. But it is not difficult to transform (6.9) to a form suitable for calculations. See further computational details in [15, 18].

Algorithm 6.2 (simulating exit point of $(t,W(t))$ from space-time rectangle Π [15, 18]). Let ι, ν, and γ be independent random variables. Let ι be simulated by the law $P(\iota=-1)=\mathcal{P}(l)$, $P(\iota=1)=1-\mathcal{P}(l)$, let ν be simulated by the law $P(\nu=\pm 1)=\frac{1}{2}$, and let γ be uniformly distributed on $[0,1]$.

Then a random point (τ,ξ), distributed as the exit point $(\tau,W(\tau))$, is simulated as follows. If the simulated value of ι is equal to -1, then the point (τ,ξ) belongs to the lateral sides of Π and

$$\tau=\mathcal{P}^{-1}(\gamma\mathcal{P}(l)),\quad \xi=\nu;$$

otherwise, when $\iota=1$, the point (τ,ξ) belongs to the upper base of Π and

$$\tau=l,\quad \xi=\mathcal{Q}^{-1}(\gamma;l).$$

Corollary 6.3. [illegible] θ [illegible] $(s,w(s))$, $s>0$, [illegible] $\partial\Pi_h$. [illegible]

$$(\theta,w)=(h\tau,\sqrt{h}\,\xi),$$

[illegible] (τ,ξ) [illegible] 6.2 [illegible] $(\theta,w(\theta))$.

Algorithm 6.1 together with Algorithm 6.2 and Corollary 6.3 gives the constructive procedure for modeling the Markov chain (ϑ_k,X_k) which approximates trajectories $(t,X(t))$ of the SDE (6.1) in the space-time bounded domain Q.

[illegible] 6.4. In the one-dimensional case one can construct a random walk which terminates on the boundary Γ rather than in a boundary layer. Indeed, fix a sufficiently small $h>0$ and define the function $\rho(t,x;h)$, $(t,x)\in Q$, in the following way. If $\Pi(t,x;h)\in Q$, set $\rho\equiv\rho(t,x;h)=h$. Otherwise, find $\rho(t,x;h)<h$ such that $\Pi(t,x;\rho)$ touches the boundary Γ; i.e., either $t+\rho=T$ or one of the ends of the interval $I(t,x;\rho)$ coincides with α or β. At each iteration of the algorithm we find $h_k=\rho(\vartheta_{k-1},X_{k-1};h)$ and simulate the first exit point $(\theta_k,w(\vartheta_{k-1}+\theta_k)-w(\vartheta_{k-1}))$ of the process $(s-\vartheta_{k-1},w(s)-w(\vartheta_{k-1}))$, $s>\vartheta_{k-1}$, from the rectangle Π_{h_k}. Then we evaluate (ϑ_k,X_k) due to (6.2)–(6.3). We stop the algorithm when $(\vartheta_k,X_k)\in\Gamma$ and put $\nu=k$, $(\vartheta_\nu,X_\nu)=(\vartheta_k,X_k)$, $\bar\tau_{t,x}=\vartheta_\nu$, $\xi_{t,x}=X_\nu$. In comparison with Algorithm 6.1, the algorithm of this remark allows us to simulate a one-dimensional space-time Brownian motion $(s,w(s))$ exactly. We note that this algorithm cannot be generalized even to the two-dimensional case, while Algorithm 6.1 is available for any dimension [15, 18].

Now we are in position to propose a numerical algorithm for solving the FBSDE with random terminal time (5.1)–(5.2). Let $\bar u(t_k,x)$ be defined by Algorithm 5.1 and

$\frac{\Delta\bar{u}}{\Delta x}(t_k, x)$ by (5.12)–(5.14). Further, we define $\bar{u}(t, x)$ by linear interpolation as

$$\bar{u}(t,x) = \frac{t_k - t}{h}\bar{u}(t_{k-1}, x) + \frac{t - t_{k-1}}{h}\bar{u}(t_k, x), \quad t_{k-1} \le x \le t_k, \tag{6.11}$$

and analogously we define $\frac{\Delta\bar{u}}{\Delta x}(t, x)$. It is clear (see (5.11) and (5.12)–(5.14)) that

$$|u(t,x) - \bar{u}(t,x)| \le Kh \tag{6.12}$$

and

$$\left|\frac{\partial u}{\partial x}(t,x) - \frac{\Delta\bar{u}}{\Delta x}(t,x)\right| \le K\sqrt{h}\,. \tag{6.13}$$

We approximate $X(t)$ from (5.1)–(5.2) by Algorithm 6.1 in which (6.3) is replaced by

$$\begin{aligned} X_k = X_{k-1} &+ b(\vartheta_{k-1}, X_{k-1}, \bar{u}(\vartheta_{k-1}, X_{k-1}))\theta_k \\ &+ \sigma(\vartheta_{k-1}, X_{k-1}, \bar{u}(\vartheta_{k-1}, X_{k-1}))(w(\vartheta_k) - w(\vartheta_{k-1})). \end{aligned} \tag{6.14}$$

The algorithm also gives us the approximation $(\bar{\tau}_{t,x}, \xi_{t,x})$ for the first exit point $(\tau_{t,x}, X_{t,x}(\tau_{t,x}))$ of the trajectory $(s, X_{t,x}(s))$ from Q. Further, we compute the components Y and Z as

$$\begin{aligned} &Y_k = \bar{u}(\vartheta_k, X_k), \quad Z_k = \sigma(\vartheta_k, X_k, Y_k)\,\frac{\Delta\bar{u}}{\Delta x}(t_k, X_k), \quad k = 1, \dots, \nu, \\ &\bar{Y}_\nu = \bar{u}(\bar{\tau}_{t,x}, \xi_{t,x}), \quad \bar{Z}_\nu = \sigma(\bar{\tau}_{t,x}, \xi_{t,x}, \bar{Y}_\nu)\,\frac{\Delta\bar{u}}{\Delta x}(\bar{\tau}_{t,x}, \xi_{t,x})\,. \end{aligned} \tag{6.15}$$

It is possible to prove (cf. (6.4) and (6.12)–(6.13)) that

(6.16)

$$\Big[E\left[(X(\vartheta_k) - X_k)^2 + (Y(\vartheta_k) - Y_k)^2 + (Z(\vartheta_k) - Z_k)^2\right]\Big]^{1/2} \le K\ (\sqrt{h} + e^{-c_h L}),$$

$$k = 1, \dots, \nu,$$

$$\Big[E\left[\big(X_{t,x}(\tau_{t,x}) - \xi_{t,x}\big)^2 + \big(Y(\tau_{t,x}) - \bar{Y}_\nu\big)^2 + \big(Z(\tau_{t,x}) - \bar{Z}_\nu\big)^2\right]\Big]^{1/2}$$

$$\le K\ (h^{(1-\varepsilon)/4} + e^{-c_h L/2}),$$

$$E|\tau_{t,x} - \bar{\tau}_{t,x}| \le K\ (h^{(1-\varepsilon)/2} + e^{-c_h L}).$$

Using Algorithm 6.1, we can also simulate the FBSDE with unbounded terminal time (5.15)–(5.16) analogously to the approximation of the FBSDE (5.1)–(5.2) considered in this section.

7. Numerical tests.

7.1. Description of the test problems.

Consider the FBSDE

$$dX = \frac{X\left(1 + X^2\right)}{\left(2 + X^2\right)^3}dt + \frac{1 + X^2}{2 + X^2}\sqrt{\frac{1 + 2Y^2}{1 + Y^2 + \exp\left(-\dfrac{2X^2}{t+1}\right)}}\,dw(t), \tag{7.1}$$

$$X(0) = x,$$

$$dY = -g(t, X, Y)dt - f(t, X, Y)Z dt + Z\,dw(t), \tag{7.2}$$

$$Y(T) = \exp\left(-\frac{X^2(T)}{T+1}\right),$$

where

$$g(t,x,u) = \frac{1}{t+1}\exp\left(-\frac{x^2}{t+1}\right) \times \left[\frac{4x^2\left(1+x^2\right)}{\left(2+x^2\right)^3} + \left(\frac{1+x^2}{2+x^2}\right)^2\left(1-\frac{2x^2}{t+1}\right) - \frac{x^2}{t+1}\right], \tag{7.3}$$

$$f(t,x,u) = \frac{x}{\left(2+x^2\right)^2}\sqrt{\frac{1+u^2+\exp\left(-\frac{2x^2}{t+1}\right)}{1+2u^2}}.$$

Note that Assumption 2.1 is satisfied.

The corresponding Cauchy problem (see (2.3)–(2.4)) has the form

$$\frac{\partial u}{\partial t} + \frac{1}{2}\left(\frac{1+x^2}{2+x^2}\right)^2 \frac{1+2u^2}{1+u^2+\exp\left(-\frac{2x^2}{t+1}\right)}\frac{\partial^2 u}{\partial x^2} + \frac{2x\left(1+x^2\right)}{\left(2+x^2\right)^3}\frac{\partial u}{\partial x} \tag{7.4}$$

$$= \frac{1}{t+1}\exp\left(-\frac{x^2}{t+1}\right)\left[\frac{x^2}{t+1} - \frac{4x^2\left(1+x^2\right)}{\left(2+x^2\right)^3} - \left(\frac{1+x^2}{2+x^2}\right)^2\left(1-\frac{2x^2}{t+1}\right)\right],$$

$$t < T,\ x \in \mathbf{R},$$

$$u(T,x) = \exp\left(-\frac{x^2}{T+1}\right). \tag{7.5}$$

We use the problem (7.1)–(7.2) to test the numerical algorithms proposed in section 4. To this end, we need to know the exact solution of this problem. First, it can easily be verified that the solution of the problem (7.4)–(7.5) is the function

$$u(t,x) = \exp\left(-\frac{x^2}{t+1}\right). \tag{7.6}$$

Now we find the solution of (7.1)–(7.2). Substituting

$$Y(t) = u(t,X(t)) = \exp\left(-\frac{X(t)^2}{t+1}\right)$$

in (7.1), we get

$$dX = \frac{X\left(1+X^2\right)}{\left(2+X^2\right)^3}dt + \frac{1+X^2}{2+X^2}\,dw(t), \quad X(0) = x, \tag{7.7}$$

whose solution can be expressed by the formula

$$X(t) = \Lambda(x + \arctan x + w(t)), \tag{7.8}$$

where the function $\Lambda(z)$ is defined by the equation

$$\Lambda + \arctan\Lambda = z\,. \tag{7.9}$$

Indeed, $X(0) = \Lambda(x + \arctan x) = x$. Further, due to the Ito formula, we have

$$dX = \Lambda'(x + \arctan x + w(t))\,dw + \frac{1}{2}\Lambda''(x + \arctan x + w(t))\,dt$$

and by (7.9) we get

$$\Lambda' = \frac{1+\Lambda^2}{2+\Lambda^2}, \quad \Lambda'' = \frac{2\Lambda(1+\Lambda^2)}{(2+\Lambda^2)^3},$$

whence it follows that (7.8) satisfies (7.7).

Thus, the solution of (7.1)–(7.2) is

$$X(t) = \Lambda(x + \arctan x + w(t)), \quad Y(t) = \exp\left(-\frac{X(t)^2}{t+1}\right), \tag{7.10}$$

$$Z(t) = -\frac{2X(t)\left(1+X^2(t)\right)}{(t+1)\left(2+X^2(t)\right)} \exp\left(-\frac{X(t)^2}{t+1}\right),$$

where $\Lambda(z)$ is defined by (7.9).

Now consider the test problem for numerical algorithms for FBSDEs with random terminal time (cf. (7.1)–(7.2)):

$$dX = \frac{X\left(1+X^2\right)}{\left(2+X^2\right)^3}dt + \frac{1+X^2}{2+X^2}\sqrt{\frac{1+2Y^2}{1+Y^2+\exp\left(-\frac{2X^2}{t+1}\right)}}\,dw(t), \tag{7.11}$$

$$X(t_0) = x \in (0,\beta),$$

$$dY = -g(t,X,Y)dt - f(t,X,Y)Z dt + Z\,dw(t), \tag{7.12}$$

$$Y(\tau_x) = \exp\left(-\frac{X^2(\tau_{t_0,x})}{\tau_{t_0,x}+1}\right),$$

where $g(t,x,y)$ and $f(t,x,y)$ are from (7.3) and $\tau_{t_0,x}$ is the first exit time of the trajectory $(t, X_{t_0,x}(t))$, $t > t_0 > -1$, from the space-time rectangle $[t_0,T)\times(0,\beta)$; i.e., either $\tau_{t_0,x} = T$ or $X(\tau_{t_0,x})$ is equal to 0 or β. The corresponding Dirichlet problem (see (5.3)–(5.4) and also (7.4)–(7.5)) has the form

$$\frac{\partial u}{\partial t} + \frac{1}{2}\left(\frac{1+x^2}{2+x^2}\right)^2 \frac{1+2u^2}{1+u^2+\exp\left(-\frac{2x^2}{t+1}\right)}\frac{\partial^2 u}{\partial x^2} + \frac{2x\left(1+x^2\right)}{\left(2+x^2\right)^3}\frac{\partial u}{\partial x} \tag{7.13}$$

$$= \frac{1}{t+1}\exp\left(-\frac{x^2}{t+1}\right)\left[\frac{x^2}{t+1} - \frac{4x^2\left(1+x^2\right)}{\left(2+x^2\right)^3} - \left(\frac{1+x^2}{2+x^2}\right)^2\left(1-\frac{2x^2}{t+1}\right)\right],$$

$$t < T,\ x \in (0,\beta),$$

$$u(t,0) = 1, \quad u(t,\beta) = \exp\left(-\frac{\beta^2}{T+1}\right), \tag{7.14}$$

$$u(T,x) = \exp\left(-\frac{x^2}{T+1}\right). \tag{7.15}$$

Obviously, the solution of this problem is given by (7.6) again. The exact solution of (7.11)–(7.12) can be simulated using formulas (7.10).

7.2. Numerical experiments. We simulate (7.1)–(7.2) using the Euler scheme (4.2) and (4.4), where the solution u of (7.4)–(7.5) is approximated by Algorithm 2.2. Of course, practical realization of such algorithms always requires a truncation of the infinite space domain using the knowledge of behavior of solutions at infinity. In

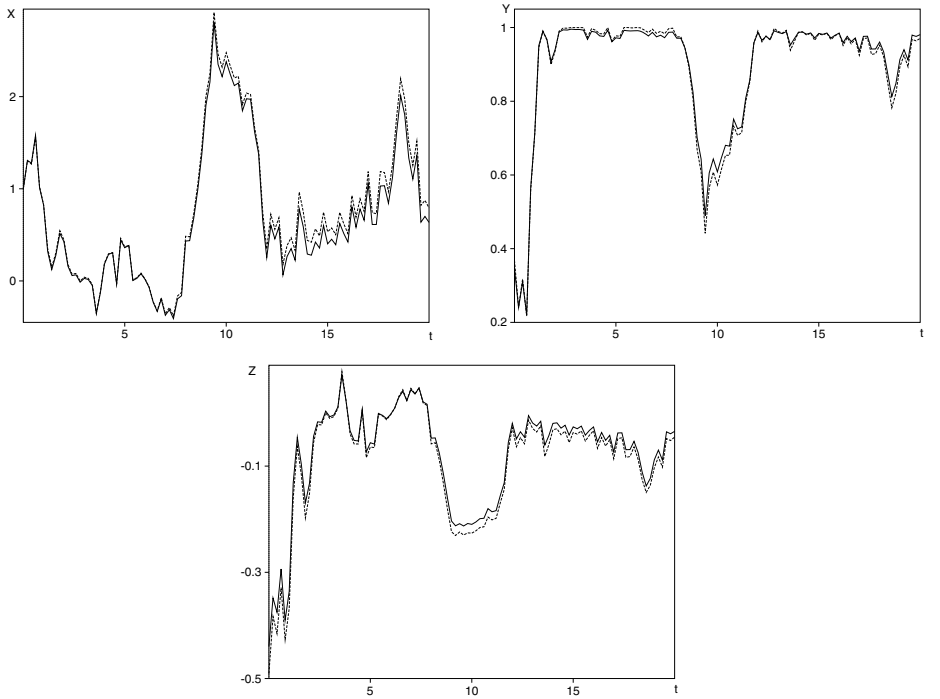


FIG. 1. *Simulation of the FBSDE* (7.1)–(7.2) *using the layer method from Algorithm* 2.2 *and the Euler scheme* (4.2) *and* (4.4) *(solid lines) with* $h = 0.2$, $\varkappa = 1$, *and* $x = 1$. *The corresponding exact trajectory (dashed lines) is found due to* (7.10). *The upper left figure gives the sample trajectories for* $X(t)$, *the upper right figure for* $Y(t)$, *and the lower figure for* $Z(t)$.

this example we restrict simulation to the space interval $[-20, 20]$. To check that this truncation does not affect accuracy, we performed control simulation for the interval $[-30, 30]$.

Figure 1 presents a comparison of the exact sample trajectories $X(t)$, $Y(t)$, $Z(t)$ found due to (7.10) and the approximate trajectories obtained by the Euler scheme (4.2) and (4.4). Table 1 gives errors in simulation of the test problem (7.1)–(7.2) by the Euler scheme (4.2) and (4.4). The "±" reflects the Monte Carlo error only; it does not reflect the error of the method. More precisely, the averages presented in the table are computed in the following way:

$$E\left(X(T) - X_N\right)^2 \doteq \frac{1}{M}\sum_{m=1}^{M}\left(X^{(m)}(T) - X_N^{(m)}\right)^2 \pm 2\sqrt{\frac{\bar{D}_M}{M}},$$

where

$$\bar{D}_M = \frac{1}{M}\sum_{m=1}^{M}\left(X^{(m)}(T) - X_N^{(m)}\right)^4 - \left[\frac{1}{M}\sum_{m=1}^{M}\left(X^{(m)}(T) - X_N^{(m)}\right)^2\right]^2$$

and $X^{(m)}(T)$ and $X_N^{(m)}$ are independent realizations of $X(T)$ and X_N, respectively. The numerical results are in good agreement with the theoretical ones proved for the Euler method: Convergence of X_N, Y_N, Z_N is of mean-square order $1/2$. We also see that $\bar{u}$ has the first-order convergence.

TABLE 1

Errors in simulation of the FBSDE (7.1)–(7.2) *by the Euler scheme* (4.2) *and* (4.4) *with* $\varkappa = 1$ *and various time steps* h. *The corresponding exact solution is found due to* (7.10). *Here* $T = 20$ *and* $x = 1$. *The expectations are computed by the Monte Carlo technique simulating* $M = 1000$ *independent realizations of* $X(T)$ *and* X_N. *The "±"reflects the Monte Carlo error only; it does not reflect the error of the method.*

h	$\max\limits_{k,j} \lvert u(t_k, x_j) - \bar{u}(t_k, x_j) \rvert$	$\left[E\left(X(T) - X_N\right)^2\right]^{1/2}$	$\left[E\left(Y(T) - Y_N\right)^2\right]^{1/2}$	$\left[E\left(Z(T) - Z_N\right)^2\right]^{1/2}$
0.5	0.15×10^{0}	0.249 ± 0.014	0.0330 ± 0.0021	0.0127 ± 0.0008
0.2	0.58×10^{-1}	0.162 ± 0.010	0.0220 ± 0.0014	0.0080 ± 0.0005
0.05	0.14×10^{-1}	0.080 ± 0.005	0.0109 ± 0.0007	0.0041 ± 0.0003
0.02	0.53×10^{-2}	0.051 ± 0.003	0.0069 ± 0.0004	0.0024 ± 0.0002
0.005	0.12×10^{-2}	0.025 ± 0.002	0.0034 ± 0.0002	0.0012 ± 0.0001

The algorithms from section 6 were tested on the FBSDE with random terminal time (7.11)–(7.12). The tests supported the obtained theoretical results.

REFERENCES

[1] F. ANTONELLI, *Backward-forward stochastic differential equations*, Ann. Appl. Probab., 3 (1993), pp. 777–793.

[2] V. BALLY, *Approximation scheme for solutions of BSDE*, in Backward Stochastic Differential Equations, N. El Karoui and L. Mazliak, eds., Pitman, London, 1997, pp. 177–191.

[3] V. BALLY, G. PAGÈS, AND J. PRINTEMS, *A stochastic quantization method for nonlinear problems*, Monte Carlo Methods Appl., 7 (2001), pp. 21–34.

[4] D. CHEVANCE, *Numerical methods for backward stochastic differential equations*, in Numerical Methods in Finance, L. C. G. Rogers and D. Talay, eds., Cambridge University Press, Cambridge, UK, 1997, pp. 232–244.

[5] J. CVITANIĆ AND J. MA, *Hedging options for a large investor and forward-backward SDE's*, Ann. Appl. Probab., 6 (1996), pp. 370–398.

[6] J. DOUGLAS, JR., J. MA, AND P. PROTTER, *Numerical methods for forward-backward stochastic differential equations*, Ann. Appl. Probab., 6 (1996), pp. 940–968.

[7] D. DUFFIE, J. MA, AND J. YONG, *Black's consol rate conjecture*, Ann. Appl. Probab., 5 (1995), pp. 356–382.

[8] P. GRINDROD, *The Theory and Applications of Reaction-Diffusion Equations: Patterns and Waves*, Clarendon Press, Oxford, UK, 1996.

[9] N. EL KAROUI, S. PENG, AND M. C. QUENEZ, *Backward stochastic differential equations in finance*, Math. Finance, 7 (1997), pp. 1–71.

[10] O. A. LADYZHENSKAYA, V. A. SOLONNIKOV, AND N. N. URAL'CEVA, *Linear and Quasilinear Equations of Parabolic Type*, AMS, Providence, RI, 1988.

[11] J. MA, P. PROTTER, AND J. YONG, *Solving forward-backward stochastic differential equations explicitly—a four step scheme*, Probab. Theory Related Fields, 98 (1994), pp. 339–359.

[12] J. MA AND J. YONG, *Forward-Backward Stochastic Differential Equations and Their Applications*, Lecture Notes in Math. 1702, Springer, Berlin, 1999.

[13] G. N. MIL'SHTEIN, *A theorem on the order of convergence of mean-square approximations of solutions of systems of stochastic differential equations*, Theory Probab. Appl., 32 (1987), pp. 738–741.

[14] G. N. MILSTEIN, *The probability approach to numerical solution of nonlinear parabolic equations*, Num. Methods Partial Differential Equations, 18 (2002), pp. 490–522.

[15] G. N. MILSTEIN AND M. V. TRETYAKOV, *Simulation of a space-time bounded diffusion*, Ann. Appl. Probab., 9 (1999), pp. 732–779.

[16] G. N. MILSTEIN AND M. V. TRETYAKOV, *Numerical solution of the Dirichlet problem for nonlinear parabolic equations by a probabilistic approach*, IMA J. Numer. Anal., 21 (2001), pp. 887–917.

[17] G. N. MILSTEIN AND M. V. TRETYAKOV, *A probabilistic approach to the solution of the Neumann problem for nonlinear parabolic equations*, IMA J. Numer. Anal., 22 (2002), pp. 599–622.

[18] G. N. Milstein and M. V. Tretyakov, *Stochastic Numerics for Mathematical Physics*, Springer, Berlin, 2004.
[19] E. Pardoux, *Backward stochastic differential equations and viscosity solutions of systems of semilinear parabolic and elliptic PDEs of second order*, in Stochastic Analysis and Related Topics VI, Birkhäuser Boston, Boston, 1998, pp. 79–127.
[20] E. Pardoux and S. Peng, *Backward stochastic differential equations and quasilinear parabolic partial differential equations*, in Stochastic Partial Differential Equations and Their Applications, Lecture Notes in Control and Inform. Sci. 176, Springer, New York, 1992, pp. 200–217.
[21] E. Pardoux and S. Tang, *Forward backward stochastic differential equations and quasilinear parabolic PDEs*, Probab. Theory Related Fields, 114 (1999), pp. 123–150.
[22] S. Peng, *Probabilistic interpretation for systems of quasilinear parabolic partial differential equations*, Stochastics Stochastics Rep., 37 (1991), pp. 61–74.
[23] A. A. Samarskii, V. A. Galaktionov, S. P. Kurdyumov, and A. P. Mikhailov, *Blow-up in Quasilinear Parabolic Equations*, Walter de Gruyter, Berlin, New York, 1995.
[24] M. E. Taylor, *Partial Differential Equations* III. *Nonlinear Equations*, Springer, Berlin, 1996.
[25] J. Yong and X. Y. Zhou, *Stochastic Controls. Hamiltonian Systems and HJB Equations*, Springer, New York, 1999.

SIAM J. SCI. COMPUT.
Vol. 28, No. 2, pp. 583–603
© 2006 Society for Industrial and Applied Mathematics

OPTIMAL PARAMETER IN HERMITIAN AND SKEW-HERMITIAN SPLITTING METHOD FOR CERTAIN TWO-BY-TWO BLOCK MATRICES*

ZHONG-ZHI BAI†, GENE H. GOLUB‡, AND CHI-KWONG LI§

Abstract. The optimal parameter of the Hermitian/skew-Hermitian splitting (HSS) iteration method for a real two-by-two linear system is obtained. The result is used to determine the optimal parameters for linear systems associated with certain two-by-two block matrices and to estimate the optimal parameters of the HSS iteration method for linear systems with n-by-n real coefficient matrices. Numerical examples are given to illustrate the results.

Key words. non-Hermitian matrix, Hermitian matrix, skew-Hermitian matrix, splitting iteration method, optimal iteration parameter

AMS subject classifications. 65F10, 65F50

CR subject classification. G.1.3

DOI. 10.1137/050623644

1. Introduction.

To solve the large sparse non-Hermitian and positive definite system of linear equations

$$Ax = f,\ A \in \mathbb{C}^{n\times n} \text{ positive definite},\ A \neq A^*,\ \text{and } x, f \in \mathbb{C}^n, \tag{1.1}$$

Bai, Golub, and Ng [2] recently proposed the [illegible] (HSS) iteration method based on the fact that the coefficient matrix A naturally possesses the [illegible] (HS) splitting [10, 19]

$$A = H + S,$$

where

$$H = \frac{1}{2}(A + A^*) \quad \text{and} \quad S = \frac{1}{2}(A - A^*),$$

with A^* being the conjugate transpose of the matrix A. They showed that this HSS iteration converges unconditionally to the exact solution of the system of linear

*Received by the editors February 1, 2005; accepted for publication (in revised form) November 23, 2005; published electronically May 5, 2006.
http://www.siam.org/journals/sisc/28-2/62364.html

†Department of Mathematics, Fudan University, Shanghai 200433, People's Republic of China, and State Key Laboratory of Scientific/Engineering Computing, Institute of Computational Mathematics and Scientific/Engineering Computing, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, P. O. Box 2719, Beijing 100080, People's Republic of China (bzz@lsec.cc.ac.cn). The work of this author was supported by The National Basic Research Program (2005CB321702); the China NNSF National Outstanding Young Scientist Foundation (10525102); the National Natural Science Foundation (10471146), People's Republic of China; and the 2004 Ky and Yu-Fen Fan Fund Travel Grant of the American Mathematical Society.

‡Scientific Computing and Computational Mathematics Program, Department of Computer Science, Stanford University, Stanford, CA 94305-9025 (golub@sccm.stanford.edu). The work of this author was partially supported by the Department of Energy (DE-FC02-01ER41177).

§Department of Mathematics, The College of William and Mary, P. O. Box 8795, Williamsburg, VA 23187-8795 (ckli@math.wm.edu). The work of this author was partially supported by an NSF grant. This author is an honorary professor of Heilongjiang University and of the University of Hong Kong.

equations (1.1), with the upper bound on convergence speed about the same as that of the conjugate gradient method when applied to Hermitian matrices. Moreover, the upper bound of the contraction factor is dependent on the spectrum of the Hermitian part H but is independent of the spectrum of the skew-Hermitian part S as well as the eigenvalues of the matrices H, S, and A. Numerical experiments have shown that the HSS iteration method is very efficient and robust both as a solver and as a preconditioner (to Krylov subspace methods such as GMRES and BiCGSTAB; see [15, 18]) for solving non-Hermitian and positive definite linear systems.

To further improve the efficiency of the method, it is desirable to determine or find a good estimate for the optimal parameter α^*. Unfortunately, there is no good method for doing that. In this paper, we analyze two-by-two real matrices in detail and obtain the optimal parameter α^* that minimizes the spectral radius of the iteration matrix of the corresponding HSS method. We then use the results to determine the optimal parameters for linear systems associated with certain two-by-two block matrices and to estimate the optimal parameter α^* of the HSS method for general n-by-n nonsymmetric positive definite system of linear equations (1.1). Numerical examples are given to show that our estimations improve previous results and are close to the values of the optimal parameters.

Unless specified otherwise, we assume throughout the paper that the non-Hermitian matrix $A \in \mathbb{C}^{n\times n}$ is positive definite, i.e., $A \neq A^*$, and its Hermitian part $H = \frac{1}{2}(A + A^*)$ is Hermitian positive definite.

2. The HSS iteration. Let us first review the HSS iteration method presented in Bai, Golub, and Ng [2].

The HSS iteration method. [illegible] $x^{(0)} \in \mathbb{C}^n$ [illegible] $x^{(k)}$ [illegible] $k = 0, 1, 2, \ldots$ [illegible] $\{x^{(k)}\}$ [illegible]

$$\begin{cases} (\alpha I + H)x^{(k+\frac{1}{2})} = (\alpha I - S)x^{(k)} + f, \\ (\alpha I + S)x^{(k+1)} = (\alpha I - H)x^{(k+\frac{1}{2})} + f, \end{cases}$$

[illegible] α [illegible]

In matrix-vector form, the above HSS iteration method can be equivalently rewritten as

$$x^{(k+1)} = \mathcal{M}(\alpha)x^{(k)} + \mathcal{G}(\alpha)f, \qquad k = 0, 1, 2, \ldots, \tag{2.1}$$

where

$$\mathcal{M}(\alpha) = (\alpha I + S)^{-1}(\alpha I - H)(\alpha I + H)^{-1}(\alpha I - S) \tag{2.2}$$

and

$$\mathcal{G}(\alpha) = 2\alpha(\alpha I + S)^{-1}(\alpha I + H)^{-1}.$$

Here, $\mathcal{M}(\alpha)$ is the iteration matrix of the HSS method. In fact, (2.1) may also result from the splitting

$$A = B(\alpha) - C(\alpha)$$

of the coefficient matrix A, with

$$\begin{cases} B(\alpha) = \dfrac{1}{2\alpha}(\alpha I + H)(\alpha I + S), \\ C(\alpha) = \dfrac{1}{2\alpha}(\alpha I - H)(\alpha I - S). \end{cases}$$

The following theorem established in [2] describes the convergence property of the HSS iteration.

THEOREM 2.1. $A \in \mathbb{C}^{n\times n}$ $H = \frac{1}{2}(A + A^*)$ $S = \frac{1}{2}(A - A^*)$ α $\rho(\mathcal{M}(\alpha))$ $\mathcal{M}(\alpha)$ (2.2)

$$\sigma(\alpha) = \max_{\lambda_j \in \lambda(H)} \frac{|\alpha - \lambda_j|}{|\alpha + \lambda_j|},$$

$\lambda(\cdot)$

$$\rho(\mathcal{M}(\alpha)) \le \sigma(\alpha) < 1 \qquad \alpha > 0;$$

$x^* \in \mathbb{C}^n$ (1.1)

$\gamma_{\min}$ $\gamma_{\max}$ H

$$\tilde{\alpha} \equiv \arg\min_{\alpha} \left\{ \max_{\gamma_{\min} \le \lambda \le \gamma_{\max}} \left| \frac{\alpha - \lambda}{\alpha + \lambda} \right| \right\} = \sqrt{\gamma_{\min}\gamma_{\max}}$$

$$\sigma(\tilde{\alpha}) = \frac{\sqrt{\gamma_{\max}} - \sqrt{\gamma_{\min}}}{\sqrt{\gamma_{\max}} + \sqrt{\gamma_{\min}}} = \frac{\sqrt{\kappa(H)} - 1}{\sqrt{\kappa(H)} + 1},$$

$\kappa(H)$ H

Of course, $\tilde{\alpha}$ is usually different from the optimal parameter

$$\alpha^* = \arg\min_{\alpha} \rho(\mathcal{M}(\alpha)),$$

and it always holds that

$$\rho(\mathcal{M}(\alpha^*)) \le \sigma(\tilde{\alpha}).$$

Numerical experiments in [2] have confirmed that in most situations, $\rho(\mathcal{M}(\alpha^*)) \ll \sigma(\tilde{\alpha})$. See [3, 12, 6, 16, 1, 13, 8, 14, 11, 7] for further applications and generalizations of the HSS iteration method.

3. The real two-by-two case. In this section, we study linear systems associated with a real two-by-two matrix A with positive definite symmetric part. We first determine the eigenvalues of $\mathcal{M}(\alpha)$ defined in (2.2). The following theorem is stated in general terms so that it can be used more conveniently for future discussion.

THEOREM 3.1. $A = H + S \in \mathbb{R}^{2\times 2}$ H S H $\lambda_1 \ge \lambda_2 > 0$ $\det(S) = q^2$ $q \in \mathbb{R}$ $\mathcal{M}(\alpha)$ (2.2)

$$\lambda_\pm = \frac{(\alpha^2 - \lambda_1\lambda_2)(\alpha^2 - q^2) \pm \sqrt{(\alpha^2 - \lambda_1\lambda_2)^2(\alpha^2 - q^2)^2 - (\alpha^2 - \lambda_1^2)(\alpha^2 - \lambda_2^2)(\alpha^2 + q^2)^2}}{(\alpha + \lambda_1)(\alpha + \lambda_2)(\alpha^2 + q^2)}.$$

[illegible]

$$(\alpha^2-\lambda_1\lambda_2)^2(\alpha^2-q^2)^2 \geq (\alpha^2-\lambda_1^2)(\alpha^2-\lambda_2^2)(\alpha^2+q^2)^2,$$

[illegible] $\rho(\mathcal{M}(\alpha))$ [illegible]

$$\frac{|\alpha^2-\lambda_1\lambda_2||\alpha^2-q^2|+\sqrt{(\alpha^2-\lambda_1\lambda_2)^2(\alpha^2-q^2)^2-(\alpha^2-\lambda_1^2)(\alpha^2-\lambda_2^2)(\alpha^2+q^2)^2}}{(\alpha+\lambda_1)(\alpha+\lambda_2)(\alpha^2+q^2)};$$

[illegible]

$$(\alpha^2-\lambda_1\lambda_2)^2(\alpha^2-q^2)^2 < (\alpha^2-\lambda_1^2)(\alpha^2-\lambda_2^2)(\alpha^2+q^2)^2,$$

[illegible] $\rho(\mathcal{M}(\alpha))$ [illegible]

$$\sqrt{\frac{(\alpha-\lambda_1)(\alpha-\lambda_2)}{(\alpha+\lambda_1)(\alpha+\lambda_2)}}.$$

[illegible]. Let $A = H + S \in \mathbb{R}^{2\times 2}$, where H is symmetric positive definite, and S is skew-symmetric. Then there is an orthogonal matrix $Q \in \mathbb{R}^{2\times 2}$ such that Q^tHQ is a diagonal matrix with diagonal entries λ_1 and λ_2, where Q^t denotes the transpose matrix of Q. We may replace A by Q^tAQ without changing the assumptions and conclusions of our theorem. So, assume that

$$H = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \quad \text{and} \quad S = \begin{bmatrix} 0 & q \\ -q & 0 \end{bmatrix} \qquad \text{with } q \in \mathbb{R}.$$

Then $(\alpha I + H)^{-1}(\alpha I - H)(\alpha I + S)^{-1}(\alpha I - S)$ equals

$$\frac{1}{\alpha^2+q^2} \cdot \begin{bmatrix} \dfrac{(\alpha^2-q^2)(\alpha-\lambda_1)}{\alpha+\lambda_1} & -\dfrac{2q\alpha(\alpha-\lambda_1)}{\alpha+\lambda_1} \\ \dfrac{2q\alpha(\alpha-\lambda_2)}{\alpha+\lambda_2} & \dfrac{(\alpha^2-q^2)(\alpha-\lambda_2)}{\alpha+\lambda_2} \end{bmatrix}.$$

The formula for $\lambda_\pm$ and the assertion on $\rho(\mathcal{M}(\alpha))$ follow. □

One may want to use the formula of $\rho(\mathcal{M}(\alpha))$ in Theorem 3.1 to determine the optimal choice of α. It turns out that the analysis is very complicated and not productive. The main difficulty is the expression

$$\sqrt{(\alpha^2-\lambda_1\lambda_2)^2(\alpha^2-q^2)^2-(\alpha^2-\lambda_1^2)(\alpha^2-\lambda_2^2)(\alpha^2+q^2)^2} \tag{3.1}$$

in the formula of $\rho(\mathcal{M}(\alpha))$. For example, one may see [5] for the analysis of a similar and simpler problem. Here, we use a different approach that allows us to avoid the complicated expression (3.1).

For notational simplicity, we write

$$\rho(\alpha) = \rho(\mathcal{M}(\alpha)).$$

Define

$$\phi(\alpha) = \left\{\frac{\operatorname{trace}(\mathcal{M}(\alpha))}{2}\right\}^2 = \left\{\frac{(\alpha^2-q^2)(\alpha^2-\lambda_1\lambda_2)}{(\alpha^2+q^2)(\alpha+\lambda_1)(\alpha+\lambda_2)}\right\}^2,$$

$$\psi(\alpha) = \det(\mathcal{M}(\alpha)) = \frac{(\alpha-\lambda_1)(\alpha-\lambda_2)}{(\alpha+\lambda_1)(\alpha+\lambda_2)},$$

and

$$\omega(\alpha) = \max\{\phi(\alpha),\ |\psi(\alpha)|\}.$$

Evidently,

$$\rho(\alpha)^2 \geq \omega(\alpha).$$

Moreover,

$$1 = \phi(0) = \lim_{\alpha\to+\infty} \phi(\alpha) \quad \text{and} \quad 1 = \psi(0) = \lim_{\alpha\to+\infty} \psi(\alpha).$$

Thus,

$$\lim_{\alpha\to+\infty} \omega(\alpha) = \omega(0) = 1 > \omega(\xi) \quad \text{for all } \xi > 0.$$

Since $\omega(\alpha)$ is continuous and nonnegative, there exists $\alpha^* > 0$ such that

$$\omega(\alpha^*) = \min\{\omega(\alpha) \mid \alpha > 0\}.$$

We will show that

$$\phi(\alpha^*) = |\psi(\alpha^*)|. \tag{3.2}$$

As a result, the eigenvalues of $\mathcal{M}(\alpha^*)$ have the same modulus, and thus

$$\rho(\alpha)^2 \geq \omega(\alpha) \geq \omega(\alpha^*) = \rho(\alpha^*)^2 \quad \text{for all } \alpha > 0.$$

By the above discussion, establishing (3.2) will lead to the following theorem.

THEOREM 3.2. [illegible] 3.1 [illegible] ϕ [illegible] ψ [illegible] $\alpha^* > 0$ [illegible]

$$\rho(\mathcal{M}(\alpha^*)) = \min\{\rho(\mathcal{M}(\alpha)) \mid \alpha > 0\}$$

[illegible]

$$\mathbf{S} = \{\alpha > 0 \mid \phi(\alpha) = |\psi(\alpha)|\},$$

[illegible] $\alpha > 0$ [illegible]

$$(\alpha^2 + q^2)^2(\alpha^2 - \lambda_1^2)(\alpha^2 - \lambda_2^2) = (\alpha^2 - q^2)^2(\alpha^2 - \lambda_1\lambda_2)^2 \tag{3.3}$$

[illegible]

$$(\alpha^2 + q^2)^2(\lambda_1^2 - \alpha^2)(\alpha^2 - \lambda_2^2) = (\alpha^2 - q^2)^2(\alpha^2 - \lambda_1\lambda_2)^2. \tag{3.4}$$

[illegible]. We need only demonstrate the validity of (3.2), i.e.,

$$\phi(\alpha^*) = |\psi(\alpha^*)|.$$

Note that $\phi(\alpha)$ is continuously differentiable for $\alpha > 0$, and $|\psi(\alpha)|$ is continuous for $\alpha > 0$ and differentiable except for $\alpha = \lambda_1$ and λ_2. Since $\omega(\alpha) = \max\{\phi(\alpha),\ |\psi(\alpha)|\}$, if α^* satisfies

$$\omega(\alpha^*) = \min\{\omega(\alpha) \mid \alpha > 0\},$$

then one of the following holds:

(i) $\phi(\alpha^*) = |\psi(\alpha^*)|$;

(ii) $\phi(\alpha^*) > |\psi(\alpha^*)|$ and $\phi(\alpha^*)$ is a local minimum of $\phi(\alpha)$;

(iii) $|\psi(\alpha^*)| > \phi(\alpha^*)$ and $|\psi(\alpha^*)|$ is a local minimum of $|\psi(\alpha)|$.

First, we claim that (iii) cannot happen. To see this, note that

$$\psi'(\alpha) = \frac{2(\lambda_1+\lambda_2)(\alpha^2-\lambda_1\lambda_2)}{[(\alpha+\lambda_1)(\alpha+\lambda_2)]^2},$$

which is positive or negative according to $\alpha > \sqrt{\lambda_1\lambda_2}$ or $\alpha < \sqrt{\lambda_1\lambda_2}$, respectively, and $\psi'\left(\sqrt{\lambda_1\lambda_2}\right) = 0$. Because $\lambda_1 \geq \sqrt{\lambda_1\lambda_2} \geq \lambda_2$, we see that $|\psi(\alpha)| = \psi(\alpha)$ is differentiable and decreasing on $(0, \lambda_2)$, and $|\psi(\alpha)| = \psi(\alpha)$ is differentiable and increasing on $(\lambda_1, +\infty)$. Thus, there cannot be α^* in $(0, \lambda_2) \cup (\lambda_1, +\infty)$ satisfying (iii). Furthermore, $|\psi(\lambda_j)| = 0$ for $j = 1, 2$; so, it is impossible to have $\alpha^* = \lambda_j$ satisfying (iii). Finally, $|\psi(\alpha)| = -\psi(\alpha)$ is differentiable on (λ_2, λ_1) with a local maximum at $\sqrt{\lambda_1\lambda_2}$. Thus, there cannot be α^* in (λ_2, λ_1) satisfying (iii).

Next, we show that (ii) cannot happen. The analysis is more involved. Instead of $\phi(\alpha)$, we consider its square root

$$F(\alpha) = \frac{(\alpha^2-q^2)(\alpha^2-\lambda_1\lambda_2)}{(\alpha^2+q^2)(\alpha+\lambda_1)(\alpha+\lambda_2)} = \left(1 - \frac{2q^2}{\alpha^2+q^2}\right)\left(1 - \frac{\lambda_1}{\alpha+\lambda_1} - \frac{\lambda_2}{\alpha+\lambda_2}\right).$$

Then

$$\begin{aligned} F'(\alpha) &= \frac{4q^2\alpha}{(\alpha^2+q^2)^2}\frac{(\alpha^2-\lambda_1\lambda_2)}{(\alpha+\lambda_1)(\alpha+\lambda_2)} + \frac{(\alpha^2-q^2)}{(\alpha^2+q^2)}\left(\frac{\lambda_1}{(\alpha+\lambda_1)^2} + \frac{\lambda_2}{(\alpha+\lambda_2)^2}\right) \\ &= \frac{\lambda_1+\lambda_2}{(\alpha^2+q^2)^2(\alpha+\lambda_1)^2(\alpha+\lambda_2)^2} \cdot P(\alpha), \end{aligned}$$

where

$$\begin{aligned} P(\alpha) = \alpha^6 &+ \frac{4(\lambda_1\lambda_2+q^2)}{\lambda_1+\lambda_2}\alpha^5 + (\lambda_1\lambda_2+4q^2)\alpha^4 \\ &- q^2(q^2+4\lambda_1\lambda_2)\alpha^2 - \frac{4q^2\lambda_1\lambda_2(q^2+\lambda_1\lambda_2)}{\lambda_1+\lambda_2}\alpha - \lambda_1\lambda_2q^4. \end{aligned}$$

Suppose $q \leq \sqrt{\lambda_1\lambda_2}$. Then $F'(\alpha) < 0$ for $\alpha \in (0, q)$, and $F'(\alpha) > 0$ for $\alpha \in (\sqrt{\lambda_1\lambda_2}, +\infty)$. So, $F(\alpha)$ is decreasing on $(0, q)$ and increasing on $[\sqrt{\lambda_1\lambda_2}, +\infty)$. Hence, $\phi(\alpha) = F(\alpha)^2$ cannot have a local minimum in these two intervals. Thus, there cannot be an α^* in these intervals satisfying (ii). Because $\phi(q) = \phi\left(\sqrt{\lambda_1\lambda_2}\right) = 0$, we cannot have $\alpha^* = q$ or $\sqrt{\lambda_1\lambda_2}$ satisfying (ii).

Next, we [illegible] that $F(\alpha)$ has only one critical point in $\left(q, \sqrt{\lambda_1\lambda_2}\right)$, which is a local minimum. This point will be the unique critical point for $\phi(\alpha) = F(\alpha)^2$ on $\left(q, \sqrt{\lambda_1\lambda_2}\right)$, which is a local maximum. Thus, there cannot be an α^* in this interval satisfying (ii).

To prove our claim, note that $\alpha > 0$ is a critical point of $F(\alpha)$ if and only if α is a zero of $P(\alpha)$. So, it suffices to show that $P(\alpha)$ has only one positive zero. Now

$$P''(\alpha) = 30\alpha^4 + \frac{80(\lambda_1\lambda_2+q^2)}{\lambda_1+\lambda_2}\alpha^3 + 12(\lambda_1\lambda_2+4q^2)\alpha^2 - 2q^2(q^2+4\lambda_1\lambda_2).$$

Since $P''(0) < 0$ and $\lim_{\alpha\to+\infty} P''(\alpha) = +\infty$, we see that $P''(\alpha)$ has a positive zero. However, $P''(\alpha)$ cannot have two or more positive zeros (counting multiplicity). Otherwise,

$$P'''(\alpha) = 120\alpha^3 + \frac{240(\lambda_1\lambda_2 + q^2)}{\lambda_1 + \lambda_2}\alpha^2 + 24(\lambda_1\lambda_2 + 4q^2)\alpha$$

has a positive zero, which is impossible. Next, we argue that

$$P'(\alpha) = 6\alpha^5 + \frac{20(\lambda_1\lambda_2 + q^2)}{\lambda_1 + \lambda_2}\alpha^4 + 4(\lambda_1\lambda_2 + 4q^2)\alpha^3$$

$$- 2q^2(q^2 + 4\lambda_1\lambda_2)\alpha - \frac{4q^2\lambda_1\lambda_2(q^2 + \lambda_1\lambda_2)}{\lambda_1 + \lambda_2}$$

has exactly one positive zero. Since $P'(0) < 0$ and $\lim_{\alpha\to+\infty} P'(\alpha) = +\infty$, we see that $P'(\alpha)$ has a positive zero. Since $P''(0) < 0$ and $P''(\alpha)$ has only one positive zero α_1, we see that $P'(\alpha)$ is decreasing on $(0, \alpha_1)$ and increasing on $(\alpha_1, +\infty)$. So, $P'(\alpha)$ can have only one positive zero $\alpha_2 > \alpha_1$.

We can apply the same argument to $P(\alpha)$. Since $P(0) < 0$ and $\lim_{\alpha\to+\infty} P(\alpha) = +\infty$, we see that $P(\alpha)$ has a positive zero. Since $P'(0) < 0$ and $P'(\alpha)$ has only one positive zero α_2, we see that $P(\alpha)$ is decreasing on $(0, \alpha_2)$ and increasing on $(\alpha_2, +\infty)$. So, $P(\alpha)$ can have only one positive zero $\alpha_3 > \alpha_2$, and our claim is proved.

One can use a similar argument to get the desired conclusion if $q > \sqrt{\lambda_1\lambda_2}$. We omit the discussion. □

Remark 3.3. Using the absolute value function, we can combine (3.3) and (3.4) to a single equation

$$|(\alpha^2 + q^2)^2(\lambda_1^2 - \alpha^2)(\alpha^2 - \lambda_2^2)| = [(\alpha^2 - q^2)(\alpha^2 - \lambda_1\lambda_2)]^2.$$

Nonetheless, the polynomial equations are easier to solve and use. In fact, if $\lambda_1 = \lambda_2 = \lambda^*$, then (3.3) and (3.4) have only one positive solution, namely, $\alpha = \lambda^*$. Suppose $\lambda_1 > \lambda_2$. If we use the substitution $\beta = \alpha^2$, then (3.3) reduces to the quadratic equation

$$[(\lambda_1 - \lambda_2)^2 - 4q^2]\beta^2 + 2q^2(\lambda_1 + \lambda_2)^2\beta + q^2[q^2(\lambda_1 - \lambda_2)^2 - 4\lambda_1^2\lambda_2^2] = 0,$$

which has solutions

$$\frac{q(2\lambda_1\lambda_2 - q\lambda_1 + q\lambda_2)}{\lambda_1 - \lambda_2 + 2q} \quad \text{and} \quad \frac{-q(2\lambda_1\lambda_2 + q\lambda_1 - q\lambda_2)}{\lambda_1 - \lambda_2 - 2q}$$

if $|\lambda_1 - \lambda_2| \neq 2q$. Otherwise, the equation is linear and has a solution

$$\beta = \frac{2(q^4 - \lambda_1^2\lambda_2^2)}{(\lambda_1 + \lambda_2)^2} = \frac{2(q^2 + \lambda_1\lambda_2)(q^2 - \lambda_1\lambda_2)}{(\lambda_1 + \lambda_2)^2}$$

$$= \frac{(\lambda_1 + \lambda_2)^2(q^2 - \lambda_1\lambda_2)}{2(\lambda_1 + \lambda_2)^2} = \frac{1}{2}(q^2 - \lambda_1\lambda_2).$$

Of course, these solutions will be useful only if they are positive and lie outside the interval $[\lambda_2^2, \lambda_1^2]$. Similarly, by the substitution $\beta = \alpha^2$, (3.4) reduces to the

quartic equation

$$\begin{aligned} &2\beta^4 - (\lambda_1+\lambda_2)^2\beta^3 + 2[\lambda_1^2\lambda_2^2 - q^2(\lambda_1-\lambda_2)^2 + q^4]\beta^2 \\ &\quad - q^4(\lambda_1+\lambda_2)^2\beta + 2q^4\lambda_1^2\lambda_2^2 = 0, \end{aligned} \tag{3.5}$$

which has exactly two solutions μ_1 and μ_2 with $\mu_1 \in [\lambda_2^2, \lambda_1\lambda_2)$ and $\mu_2 \in ((\lambda_1^2 + \lambda_2^2)/2, \lambda_1^2]$. Furthermore, if $q^2 \in [\lambda_2^2, \lambda_1^2]$, then $\mu_1 \le q^2$ and $\mu_2 \ge q^2$. In particular, if $q = \lambda_i$ for $i \in \{1, 2\}$, then $\beta = \lambda_i^2$ is a solution. The verification of the above statements will be given in the last section.

As an illustration of Theorems 3.1 and 3.2 as well as Remark 3.3, we consider a simple example for which $\lambda_1 = 2$, $\lambda_2 = 1$, and $q = 1$. By straightforward computations, we know that the only positive roots of (3.3) are $\alpha_1^* = 1$ and $\alpha_2^* = \sqrt{5}$, and those of (3.4) are $\alpha_3^* = 1$ and

$$\alpha_4^* = \frac{\sqrt{6}}{6}\sqrt{\left(7 + \sqrt[3]{712 + 9\sqrt{5277}} + \frac{43}{\sqrt[3]{712 + 9\sqrt{5277}}}\right)} \approx 1.914.$$

We remark that now (3.4) is equivalent to $(\beta - 1)(2\beta^3 - 7\beta^2 + \beta - 8) = 0$. Based on Theorem 3.1, from direct calculations we have

$$\rho(\mathcal{M}(\alpha_1^*)) = \rho(\mathcal{M}(\alpha_3^*)) = 0, \quad \rho(\mathcal{M}(\alpha_2^*)) = \frac{\sqrt{3}(7 - 3\sqrt{5})}{6} \approx 0.0842$$

and

$$\rho(\mathcal{M}(\alpha_4^*)) \approx 0.201.$$

Therefore, for the HSS iteration method, the optimal parameter is $\alpha^* = 1$ and the corresponding optimal convergence factor is $\rho(\mathcal{M}(\alpha^*)) = 0$. On the other hand, from Theorem 2.1 we can easily obtain

$$\tilde{\alpha} = \sqrt{2} \approx 1.414 \quad \text{and} \quad \sigma(\tilde{\alpha}) = 3 - 2\sqrt{2} \approx 0.172.$$

Obviously, it holds that $\rho(\mathcal{M}(\alpha^*)) < \sigma(\tilde{\alpha})$.

Remark 3.4. Note that in our proof of Theorem 3.2, we get much information for the function $\omega(\alpha) = \max\{\phi(\alpha), |\psi(\alpha)|\}$ with $\alpha > 0$. In particular, we show that all the local minima of $\omega(\alpha)$ satisfy

$$\phi(\alpha) = |\psi(\alpha)|,$$

and they are the roots of (3.3) and (3.4). Moreover, the local minima of $\omega(\alpha)$ are also local minima of $\rho(\alpha)$, and the global minimum of $\rho(\alpha)$ and $\omega(\alpha)$ occurs at the same α^*. If one can prove independently that the local minima $\rho(\alpha)$ always occur when the eigenvalues of $\mathcal{M}(\alpha)$ have the same magnitude, then one can conclude that the functions $\omega(\alpha)$ and $\rho(\alpha)$ have the same local and global minima. Once again, this is difficult to do because of the expression (3.1) in $\rho(\alpha) = \rho(\mathcal{M}(\alpha))$.

Using the notation defined in Theorem 3.1, we immediately get the following conclusion from Theorem 3.2.

COROLLARY 3.5. 3.1 $\alpha^* > 0$

$$\rho(\mathcal{M}(\alpha^*)) = \min\{\rho(\mathcal{M}(\alpha)) \mid \alpha > 0\}$$

$$(\alpha^2 + q^2)^2(\alpha^2 - \lambda_1^2)(\alpha^2 - \lambda_2^2) = (\alpha^2 - q^2)^2(\alpha^2 - \lambda_1\lambda_2)^2$$

$$(\alpha^2 + q^2)^2(\lambda_1^2 - \alpha^2)(\alpha^2 - \lambda_2^2) = (\alpha^2 - q^2)^2(\alpha^2 - \lambda_1\lambda_2)^2.$$

3.6. The first equation in Corollary 3.5 can be reduced to a quadratic equation and the second one can be reduced to a quartic equation with respect to $\beta = \alpha^2$, respectively, analogously to those in Remark 3.3.

3.7. Note that the results in this section are also valid if $\lambda_1 > 0 = \lambda_2$ and $q \neq 0$.

3.8. We should point out again that Theorem 3.2 has been established only for real matrices. For complex matrices, how to determine the optimal iteration parameter of the HSS iteration method is still an open problem.

4. Two-by-two block matrices. In this section, we determine the optimal parameter α^* for a two-by-two block matrix of the form

$$A = \begin{bmatrix} \lambda_1 I_r & E \\ -E^* & \lambda_2 I_s \end{bmatrix}, \tag{4.1}$$

where $\lambda_1 > \lambda_2 > 0$. Note that the case when $\lambda_1 = \lambda_2$ is trivial.[1] Systems of linear equations with the two-by-two block matrices (4.1) arise in many applications; for details we refer the readers to [17, Chapter 6], [9, Chapters 4, 5, and 10], [20, 10, 4, 5], and the references therein. According to Young [20] and Varga [17], the matrix A in (4.1) is called a 2-cyclic matrix and is connected with property A.

THEOREM 4.1. $A \in \mathbb{C}^{n\times n}$ (4.1) $\lambda_1 > \lambda_2 > 0$ $E \in \mathbb{C}^{r\times s}$ $q_1 \geq q_2 \geq \cdots \geq q_k$

$$H = \frac{1}{2}(A + A^*) = \begin{bmatrix} \lambda_1 I_r & 0 \\ 0 & \lambda_2 I_s \end{bmatrix} \qquad S = \frac{1}{2}(A - A^*) = \begin{bmatrix} 0 & E \\ -E^* & 0 \end{bmatrix}$$

A $\rho(\mathcal{M}(\alpha))$ $\mathcal{M}(\alpha)$ (2.2) α^*

[1] If $\lambda_1 = \lambda_2 \equiv \lambda^*$, then the iteration matrix of the HSS iteration method is given by $\mathcal{M}(\alpha) = \frac{\alpha-\lambda^*}{\alpha+\lambda^*}Q(\alpha)$, where $Q(\alpha) := (\alpha I + S)^{-1}(\alpha I - S)$ is a Cayley transform of S and is, thus, unitary, and $S = \left[\begin{smallmatrix} 0 & E \\ -E^* & 0 \end{smallmatrix}\right]$; see (2.2). Obviously, it holds that $\rho(\mathcal{M}(\alpha)) = \frac{|\alpha-\lambda^*|}{\alpha+\lambda^*}$. Therefore, for the HSS iteration method applied to this special linear system, the optimal parameter is $\alpha^* = \lambda^*$ and the corresponding optimal convergence factor is $\rho(\mathcal{M}(\alpha^*)) = 0$.

$$(\alpha - \sqrt{\lambda_1\lambda_2})(\alpha - \sqrt{q_1 q_k}) = 0,$$

$$(\alpha^2 + q_j^2)^2(\alpha^2 - \lambda_1^2)(\alpha^2 - \lambda_2^2) = (\alpha^2 - q_j^2)^2(\alpha^2 - \lambda_1\lambda_2)^2$$

$$(\alpha^2 + q_j^2)^2(\lambda_1^2 - \alpha^2)(\alpha^2 - \lambda_2^2) = (\alpha^2 - q_j^2)^2(\alpha^2 - \lambda_1\lambda_2)^2,$$

$j = 1, k$

. Suppose A satisfies the hypotheses of the theorem. Then A is unitarily similar to $A_1 \oplus \cdots \oplus A_k \oplus \lambda_1 I_u \oplus \lambda_2 I_v$, where $\oplus$ denotes the matrix direct sum, $A_j = \left[\begin{smallmatrix} \lambda_1 & q_j \\ -q_j & \lambda_2 \end{smallmatrix}\right]$, $u = r - k$, and $v = s - k$. For $\alpha > 0$, define

$$\rho(\alpha) = \rho(\mathcal{M}(\alpha)) \quad \text{and} \quad \rho_j(\alpha) = \rho(\mathcal{M}_j(\alpha)),$$

where $\mathcal{M}(\alpha)$ is unitarily similar to $\mathcal{M}_1(\alpha) \oplus \cdots \oplus \mathcal{M}_k(\alpha) \oplus \frac{\alpha-\lambda_1}{\alpha+\lambda_1} I_u \oplus \frac{\alpha-\lambda_2}{\alpha+\lambda_2} I_v$, with $\mathcal{M}_j(\alpha)$ being the HSS iteration matrix associated with A_j for $j = 1, 2, \ldots, k$. Furthermore, define

$$f_j(\alpha) = |\alpha - \lambda_j|/|\alpha + \lambda_j| \quad \text{for } j = 1, 2.$$

Consider four cases.

1. If $r > k$ and $s > k$, then $f_1(\alpha)$ and $f_2(\alpha)$ are eigenvalues of $\mathcal{M}(\alpha)$ and

$$\rho(\alpha) = \max\{|f_1(\alpha)|,\ |f_2(\alpha)|\}$$

is the largest singular value of $\mathcal{M}(\alpha)$. Thus,

$$\rho(\alpha^*) = \rho\left(\sqrt{\lambda_1\lambda_2}\right) = \frac{\sqrt{\lambda_1} - \sqrt{\lambda_2}}{\sqrt{\lambda_1} + \sqrt{\lambda_2}}.$$

2. Suppose $r > k$ and $s = k$. Let $\rho_0(\alpha) = f_1(\alpha)$. Then

$$\rho(\alpha) = \max\{\rho_j(\alpha) \mid 0 \le j \le k\}.$$

For $\alpha \in \left(0, \sqrt{\lambda_1\lambda_2}\right]$,

$$\rho(\alpha) = \rho_0(\alpha) = f_1(\alpha)$$

is the largest singular value of $\mathcal{M}(\alpha)$. If $\alpha \ge (\lambda_1^3/\lambda_2)^{1/2}$, then

$$f_1(\alpha) = \left|\frac{\alpha - \lambda_1}{\alpha + \lambda_1}\right| = \left|1 - \frac{2\lambda_1}{\alpha + \lambda_1}\right| \ge \left|1 - \frac{2\lambda_1}{\sqrt{\lambda_1\lambda_2} + \lambda_1}\right| = \rho\left(\sqrt{\lambda_1\lambda_2}\right).$$

Thus, $\rho(\alpha) \ge \rho\left(\sqrt{\lambda_1\lambda_2}\right)$ if $\alpha \in \left[0, (\lambda_1\lambda_2)^{1/2}\right] \cup \left[(\lambda_1^3/\lambda_2)^{1/2}, +\infty\right)$. So, we focus on the interval

$$J = \left((\lambda_1\lambda_2)^{1/2},\ (\lambda_1^3/\lambda_2)^{1/2}\right).$$

For $\alpha \in J$, let

$$\phi_j(\alpha) = \left\{\frac{\operatorname{trace}(\mathcal{M}_j(\alpha))}{2}\right\}^2 = \left\{\frac{(\alpha^2 - q_j^2)(\alpha^2 - \lambda_1\lambda_2)}{(\alpha^2 + q_j^2)(\alpha + \lambda_1)(\alpha + \lambda_2)}\right\}^2$$

and

$$\psi(\alpha) = \det\left(\mathcal{M}_j(\alpha)\right) = \frac{(\alpha-\lambda_1)(\alpha-\lambda_2)}{(\alpha+\lambda_1)(\alpha+\lambda_2)}, \qquad j = 1, 2, \ldots, k.$$

Clearly, $\psi(\alpha)$ is independent of the index j. By Theorem 3.1, if $\phi_j(\alpha) \geq \psi(\alpha)$, then

$$\rho_j(\alpha) = \sqrt{\phi_j(\alpha)} + \sqrt{\phi_j(\alpha) - \psi(\alpha)};$$

otherwise,

$$\rho_j(\alpha) = \sqrt{\frac{(\alpha-\lambda_1)(\alpha-\lambda_2)}{(\alpha+\lambda_1)(\alpha+\lambda_2)}}.$$

We claim that

$$\max\{\rho_j(\alpha) : 1 \leq j \leq k\} = \max\{\rho_1(\alpha),\ \rho_k(\alpha)\}, \qquad \alpha \in J.$$

Suppose $1 < j < k$. If $\phi_j(\alpha) < \psi(\alpha)$, then

$$\rho_j(\alpha) = \sqrt{\frac{(\alpha-\lambda_1)(\alpha-\lambda_2)}{(\alpha+\lambda_1)(\alpha+\lambda_2)}},$$

and $\min\{\phi_1(\alpha), \phi_k(\alpha)\} \leq \phi_j(\alpha) < \psi(\alpha)$. It follows that $\rho_1(\alpha) = \rho_j(\alpha)$ or $\rho_k(\alpha) = \rho_j(\alpha)$. If $\phi_j(\alpha) \geq \psi(\alpha)$ and $\alpha \geq q_j^2$, then

$$0 \leq \phi_j(\alpha) \leq \phi_k(\alpha),$$

and hence

$$\rho_j(\alpha) = \sqrt{\phi_j(\alpha)} + \sqrt{\phi_j(\alpha) - \psi(\alpha)} \leq \sqrt{\phi_k(\alpha)} + \sqrt{\phi_k(\alpha) - \psi(\alpha)} = \rho_k(\alpha);$$

if $\phi_j(\alpha) \geq \psi(\alpha)$ and $\alpha < q_j^2$, then

$$0 < \phi_j(\alpha) \leq \phi_1(\alpha),$$

and hence

$$\rho_j(\alpha) = \sqrt{\phi_j(\alpha)} + \sqrt{\phi_j(\alpha) - \psi(\alpha)} \leq \sqrt{\phi_1(\alpha)} + \sqrt{\phi_1(\alpha) - \psi(\alpha)} = \rho_1(\alpha).$$

Thus, our claim is proved.

For each $\alpha \in J$, we have

$$\sqrt{|\alpha-\lambda_1|/|\alpha+\lambda_1|} \leq \sqrt{|\alpha-\lambda_2|/|\alpha+\lambda_2|},$$

and hence

$$\rho_0(\alpha) = \left|\frac{\alpha-\lambda_1}{\alpha+\lambda_1}\right| \leq \sqrt{\left|\frac{(\alpha-\lambda_1)(\alpha-\lambda_2)}{(\alpha+\lambda_1)(\alpha+\lambda_2)}\right|} \leq \max\{\rho_1(\alpha),\ \rho_k(\alpha)\}.$$

Combining the above, we see that

$$\rho(\alpha) = \max\{\rho_1(\alpha),\ \rho_k(\alpha)\}, \qquad \alpha \in J.$$

Let

$$\Omega(\alpha) = \max\{\phi_1(\alpha),\ \phi_k(\alpha),\ |\psi(\alpha)|\}.$$

Then

$$\Omega(\alpha) \le \max\{\rho_1(\alpha)^2,\ \rho_k(\alpha)^2\}.$$

If $\alpha^* \in J$ is such that $\Omega(\alpha^*) \le \Omega(\alpha)$ for all $\alpha \in J$, then one of the following holds:

(1) $\phi_1(\alpha^*) = \phi_k(\alpha^*) = |\psi(\alpha^*)|$;
(2.a) $\phi_1(\alpha^*) = \phi_k(\alpha^*) > |\psi(\alpha^*)|$, α^* is a local minimum of the function $\max\{\phi_1(\alpha),\ \phi_k(\alpha)\}$,
(2.b) $\phi_1(\alpha^*) = |\psi(\alpha^*)| > \phi_k(\alpha^*)$, α^* is a local minimum of the function $\max\{\phi_1(\alpha),\ |\psi(\alpha)|\}$,
(2.c) $\phi_k(\alpha^*) = |\psi(\alpha^*)| > \phi_1(\alpha^*)$, α^* is a local minimum of the function $\max\{\phi_k(\alpha),\ |\psi(\alpha)|\}$;
(3.a) $\max\{\phi_1(\alpha^*),\ \phi_k(\alpha^*)\} < |\psi(\alpha^*)|$ and α^* is a local minimum of the function $|\psi(\alpha^*)|$,
(3.b) $\max\{\phi_1(\alpha^*),\ |\psi(\alpha^*)|\} < \phi_k(\alpha^*)$ and α^* is a local minimum of the function $\phi_k(\alpha^*)$,
(3.c) $\max\{\phi_k(\alpha^*),\ |\psi(\alpha^*)|\} < \phi_1(\alpha^*)$ and α^* is a local minimum of the function $\phi_1(\alpha^*)$.

By the proof of Theorem 3.2, we see that the function $|\psi(\alpha)|$ has a differentiable local maximum at $\alpha > 0$ and two nondifferentiable local minima at λ_2 and λ_1, where $\psi(\lambda_1) = \psi(\lambda_2) = 0$. Thus, condition (3.a) cannot hold. Similarly, for $j = 1, k$, the proof of Theorem 3.2 shows that the function $\phi_j(\alpha)$ has a local maximum at $\alpha > 0$ and two local minima at $|q_j|$ and $\sqrt{\lambda_1\lambda_2}$, where $\phi_j(|q_j|) = \phi_j(\sqrt{\lambda_1\lambda_2}) = 0$. Thus, neither condition (3.b) nor (3.c) holds.

Now suppose that (1) or (2.a) holds. Then $\phi_1(\alpha^*) = \phi_k(\alpha^*)$ implies that $\alpha^* = \sqrt{q_1 q_k}$. In both cases, we have

$$\rho_1(\alpha^*) = \sqrt{\phi_1(\alpha^*)} + \sqrt{\phi_1(\alpha^*) - \psi(\alpha^*)} = \sqrt{\phi_k(\alpha^*)} + \sqrt{\phi_k(\alpha^*) - \psi(\alpha^*)} = \rho_k(\alpha^*),$$

and $\Omega(\alpha^*) = \max\{\rho_1(\alpha^*)^2,\ \rho_k(\alpha^*)^2\}$.

Suppose that (2.b) holds. If $\phi_k(\alpha^*) \ge \psi(\alpha^*)$, then

$$\rho_k(\alpha^*) = \sqrt{\phi_k(\alpha^*)} + \sqrt{\phi_k(\alpha^*) - \psi(\alpha^*)} < \sqrt{\phi_1(\alpha^*)} + \sqrt{\phi_1(\alpha^*) - \psi(\alpha^*)} = \rho_1(\alpha^*);$$

otherwise,

$$\rho_k(\alpha^*) = \sqrt{\frac{(\alpha^* - \lambda_1)(\alpha^* - \lambda_2)}{(\alpha^* + \lambda_1)(\alpha^* + \lambda_2)}} \le \rho_1(\alpha^*).$$

Thus, $\Omega(\alpha^*) = \rho_1(\alpha^*)^2 = \max\{\rho_1(\alpha^*)^2,\ \rho_k(\alpha^*)^2\}$. Moreover, since α^* is a local minimum of the function $\max\{\phi_1(\alpha),\ |\psi(\phi)|\}$, by Remark 3.4 we see that α^* is a root of the equation

$$(\alpha^2 + q_1^2)^2(\alpha^2 - \lambda_1^2)(\alpha^2 - \lambda_2^2) = (\alpha^2 - q_1^2)^2(\alpha^2 - \lambda_1\lambda_2)^2$$

or

$$(\alpha^2 + q_1^2)^2(\lambda_1^2 - \alpha^2)(\alpha^2 - \lambda_2^2) = (\alpha^2 - q_1^2)^2(\alpha^2 - \lambda_1\lambda_2)^2.$$

Suppose that (2.c) holds. We can use an argument similar to the case of (2.b) to conclude that $\Omega(\alpha^*) = \rho_k(\alpha^*)^2 = \max\{\rho_1(\alpha^*)^2,\ \rho_k(\alpha^*)^2\}$ and that α^* is a root of the equation

$$(\alpha^2 + q_k^2)^2(\alpha^2 - \lambda_1^2)(\alpha^2 - \lambda_2^2) = (\alpha^2 - q_k^2)^2(\alpha^2 - \lambda_1\lambda_2)^2$$

or

$$(\alpha^2 + q_k^2)^2(\lambda_1^2 - \alpha^2)(\alpha^2 - \lambda_2^2) = (\alpha^2 - q_k^2)^2(\alpha^2 - \lambda_1\lambda_2)^2.$$

Note that in all the cases (1), (2.a), (2.b), and (2.c), we have

$$\Omega(\alpha^*) = \max\{\rho_1(\alpha^*)^2,\ \rho_k(\alpha^*)^2\}.$$

Consequently, if $\alpha^* \in J$ yields the smallest $\Omega(\alpha)$, then

$$\rho(\alpha)^2 = \max\{\rho_1(\alpha)^2,\ \rho_k(\alpha)^2\} \geq \Omega(\alpha) \geq \Omega(\alpha^*) = \max\{\rho_1(\alpha^*)^2,\ \rho_k(\alpha^*)^2\} = \rho(\alpha^*)^2.$$

So, we need only consider those α satisfying the specified equations in the theorem to determine the optimal parameter α^*.

3. Suppose $r = k$ and $s > k$. Let $\rho_0(\alpha) = f_2(\alpha)$. Then

$$\rho(\alpha) = \max\{\rho_j(\alpha) \mid 0 \leq j \leq k\}.$$

Similarly to the proof of Case 2, we can show that

$$\rho(\alpha) \geq \rho(\sqrt{\lambda_1\lambda_2})$$

if α is positive and lies outside the interval

$$J = \left((\lambda_2^3/\lambda_1)^{1/2},\ (\lambda_1\lambda_2)^{1/2}\right).$$

So, we can focus on the interval J.

For $\alpha \in J$, we have

$$\rho(\alpha) = \max\{\rho_1(\alpha),\ \rho_k(\alpha)\}.$$

Note that in this case, for each $\alpha \in J$, we have

$$\sqrt{|\alpha - \lambda_2|/|\alpha + \lambda_2|} \leq \sqrt{|\alpha - \lambda_1|/|\alpha + \lambda_1|},$$

and hence

$$\rho_0(\alpha) = \left|\frac{\alpha - \lambda_2}{\alpha + \lambda_2}\right| \leq \sqrt{\left|\frac{(\alpha - \lambda_1)(\alpha - \lambda_2)}{(\alpha + \lambda_1)(\alpha + \lambda_2)}\right|} \leq \max\{\rho_1(\alpha),\ \rho_k(\alpha)\}.$$

Finally, if $\alpha^* \in J$ is such that $\rho(\alpha^*) < \rho(\sqrt{\lambda_1\lambda_2})$, we can show that α^* must satisfy one of the three specified equations using an argument similar to the one in Case 2.

4. Suppose $r = s = k$. Then

$$\rho(\alpha) = \max\{\rho_j(\alpha) \mid 1 \leq j \leq k\}.$$

Note that if α is positive and lies outside the interval

$$J = \left((\lambda_2^3/\lambda_1)^{1/2},\ (\lambda_1^3/\lambda_2)^{1/2}\right),$$

then each of the singular values of $\mathcal{M}(\alpha)$, which is equal to $|\alpha-\lambda_1|/|\alpha+\lambda_1|$ or $|\alpha-\lambda_2|/|\alpha+\lambda_2|$, has magnitude larger than $\rho(\sqrt{\lambda_1\lambda_2})$. Thus

$$\rho(\alpha) \geq |\det(\mathcal{M}(\alpha))|^{1/n} \geq \rho(\sqrt{\lambda_1\lambda_2}).$$

So, we can focus on those $\alpha \in J$. For each $\alpha \in J$, we can show that

$$\rho(\alpha) = \max\{\rho_1(\alpha),\ \rho_k(\alpha)\}.$$

Moreover, if $\alpha^* \in J$ is such that $\rho(\alpha^*) < \rho(\sqrt{\lambda_1\lambda_2})$, we can show that α^* must satisfy one of the three specified equations. □

REMARK 4.2. The second group of equations in Theorem 4.1 can be reduced to a group of quadratic equations and the last one can be reduced to a group of quartic equations with respect to $\beta = \alpha^2$, respectively, analogously to those in Remark 3.3.

As an illustration of Theorem 4.1 and Remark 4.2, we consider a simple example for which $\lambda_1 = 2$, $\lambda_2 = 1$, $q_1 = 1$, and $q_k = 2$. Obviously, $\alpha_0^* = \sqrt{2} \approx 1.414$ is the root of the first equation in Theorem 4.1. In addition, by straightforward computations, we know that the only positive roots of the group of the equations with respect to $q_1 = 1$ in Theorem 4.1 are

$$\alpha_1^* = 1, \quad \alpha_2^* = \sqrt{5} \approx 2.236$$

and

$$\alpha_3^* = \frac{\sqrt{6}}{6}\sqrt{\left(7 + \sqrt[3]{712+9\sqrt{5277}} + \frac{43}{\sqrt[3]{712+9\sqrt{5277}}}\right)} \approx 1.914,$$

and those with respect to $q_k = 2$ in Theorem 4.1 are

$$\alpha_4^* = 2, \quad \alpha_5^* = \frac{\sqrt{20}}{5} \approx 0.894$$

and

$$\alpha_6^* = \frac{\sqrt{6}}{6}\sqrt{\left(1 + \sqrt[3]{1477+36\sqrt{5277}} - \frac{167}{\sqrt[3]{1477+36\sqrt{5277}}}\right)} \approx 1.045.$$

We remark that now the second one in the group of equations with respect to $q_1 = 1$ is equivalent to $(\beta-1)(2\beta^3-7\beta^2+\beta-8) = 0$, and the second one in the group of equations with respect to $q_k = 2$ is equivalent to $(\beta-4)(2\beta^3-\beta^2+28\beta-32) = 0$. Based on Theorem 3.1, from direct calculations we have

$$\rho(\mathcal{M}(\alpha_0^*)) = 3-2\sqrt{2} \approx 0.172, \quad \rho(\mathcal{M}(\alpha_1^*)) = 0, \quad \rho(\mathcal{M}(\alpha_2^*)) = \frac{\sqrt{3}(7-3\sqrt{5})}{6} \approx 0.0842$$

and

$$\rho(\mathcal{M}(\alpha_3^*)) \approx 0.201, \quad \rho(\mathcal{M}(\alpha_4^*)) = 0, \quad \rho(\mathcal{M}(\alpha_5^*)) \approx 0.146, \quad \rho(\mathcal{M}(\alpha_6^*)) \approx 0.0899.$$

Therefore, for the HSS iteration method, the optimal parameter is $\alpha^* = 1$ or $\alpha^* = 2$ and the corresponding optimal convergence factor is $\rho(\mathcal{M}(\alpha^*)) = 0$. On the other hand, from Theorem 2.1 we can easily obtain

$$\tilde{\alpha} = \sqrt{2} \quad \text{and} \quad \sigma(\tilde{\alpha}) = 3-2\sqrt{2} \approx 0.172.$$

Obviously, it holds that $\rho(\mathcal{M}(\alpha^*)) < \sigma(\tilde{\alpha})$.

4.3. Our proof techniques can be used to handle the case when $\lambda_2 = 0$, which also occurs in applications; see [4] and the references therein. In such a case, we may normalize $\lambda_1 = 1$, and we always assume that $s = k$ to ensure that A is invertible. In such a case, we can use the analysis of Cases 2 and 4 in our proof of Theorem 4.1. Note that in this case, we have

$$\psi(\alpha) = \frac{\alpha - 1}{\alpha + 1}, \qquad \phi_j(\alpha) = \left\{ \frac{(\alpha^2 - q_j^2)\alpha}{(\alpha^2 + q_j^2)(\alpha + \lambda_1)} \right\}^2 \quad \text{for} \quad j = 1, k.$$

If $q_1 = q_k$, then A is unitarily similar to

$$A_1 \oplus \cdots \oplus A_k \oplus I_u$$

with

$$A_j = \begin{bmatrix} 1 & q_1 \\ -q_1 & 0 \end{bmatrix}, \qquad j = 1, 2, \ldots, k.$$

So, the analysis reduces to the two-by-two case, and Theorem 3.2 applies.

Suppose $q_1 > q_k > 0$. Then the optimal value α^* can be easily determined by checking whether $\rho_1(\alpha)$ and $\rho_k(\alpha)$ intersect at a point α^* such that

$$\phi_1(\alpha^*) - \psi(\alpha^*) = \phi_k(\alpha^*) - \psi(\alpha^*) \geq 0.$$

This happens if and only if $q_1 q_k \leq \frac{1}{2}(q_1 + q_k)$. In this case, $\alpha^* = \sqrt{q_1 q_k}$ is the optimal parameter, and

$$\rho(\mathcal{M}(\alpha^*)) = \frac{q_1 - q_k}{q_1 + q_k} \left(\frac{\sqrt{q_1 q_k}}{\sqrt{q_1 q_k} + 1} + \frac{\sqrt{(q_1 + q_k)^2 - 4q_1^2 q_k^2}}{(\sqrt{q_1 q_k} + 1)(q_1 - q_k)} \right).$$

Otherwise, $\alpha^* = \frac{q_1}{\sqrt{2q_1 - 1}}$ is the optimal parameter such that $\phi_1(\alpha^*) = \psi(\alpha^*) > \phi_k(\alpha^*)$ and

$$\rho(\mathcal{M}(\alpha^*)) = \frac{|q_1 - 1|}{q_1 + \sqrt{2q_1 - 1}}.$$

Note that the second case rarely appears in applications and its discussion was not included in [4]. Also, note that there were some typos in the formula of $\rho(\mathcal{M}(\alpha^*))$ for the first case in [4].

5. Estimation of optimal parameters for n-by-n matrices and numerical examples. In general, for a nonsymmetric and positive definite system of linear equations (1.1), the eigenvalues of its coefficient matrix A are evidently contained in the complex domain $\mathcal{D}_A := [\lambda_{\min}, \lambda_{\max}] \times \imath[-q, q]$, where $\imath$ is the imaginary unit, $\lambda_{\min}$ and $\lambda_{\max}$ are, respectively, the smallest and the largest eigenvalues of the Hermitian part H, and q is the largest module of the eigenvalues of the skew-Hermitian part S, of the coefficient matrix A. If a reduced (simpler and lower-dimensional) matrix A_R whose eigenvalues possess the same contour as the domain $\mathcal{D}_A$ is used to approximate the matrix A, then we may expect that the main mathematical and numerical properties of the HSS iteration method for the original linear system with the coefficient matrix A can be roughly preserved by the HSS iteration method applied to the linear

system with the reduced coefficient matrix A_R. A simple choice of the reduced matrix is given by

$$A_R = \begin{bmatrix} \lambda_{\max} & q \\ -q & \lambda_{\min} \end{bmatrix}$$

with $q = \|S\|$ or $q = \rho(H^{-1}S)\sqrt{\lambda_{\min}\lambda_{\max}}$. We can then use Theorem 3.2 and Corollary 3.5 to estimate the optimal parameter α^* of the HSS iteration method as follows.

ESTIMATION 5.1. [illegible] $A \in \mathbb{R}^{n\times n}$ [illegible] $H, S \in \mathbb{R}^{n\times n}$ [illegible] $\lambda_{\min} = \min\{\lambda \mid \lambda \in \lambda(H)\}$ $\lambda_{\max} = \max\{\lambda \mid \lambda \in \lambda(H)\}$ [illegible]

$$q = \|S\| \quad [\text{illegible}] \quad q = \rho(H^{-1}S)\sqrt{\lambda_{\min}\lambda_{\max}}.$$

[illegible]

$$(\alpha^2 + q^2)^2(\alpha^2 - \lambda_{\max}^2)(\alpha^2 - \lambda_{\min}^2) = (\alpha^2 - q^2)^2(\alpha^2 - \lambda_{\min}\lambda_{\max})^2$$

[illegible]

$$(\alpha^2 + q^2)^2(\lambda_{\max}^2 - \alpha^2)(\alpha^2 - \lambda_{\min}^2) = (\alpha^2 - q^2)^2(\alpha^2 - \lambda_{\min}\lambda_{\max})^2$$

[illegible] $\alpha^* > 0$ [illegible]

$$\rho(\mathcal{M}(\alpha^*)) = \min\{\rho(\mathcal{M}(\alpha)) \mid \alpha > 0\}.$$

[illegible] $\mathcal{M}(\alpha)$ [illegible] (2.2)

We first illustrate our estimates with the following example of a general nonsymmetric positive definite system of linear equations; see [2].

Consider the two-dimensional convection-diffusion equation

$$-(u_{xx} + u_{yy}) + \delta(u_x + u_y) = g(x, y)$$

on the unit square $[0, 1]\times[0, 1]$, with constant coefficient δ and subject to Dirichlet-type boundary conditions. When the five-point centered finite difference discretization is applied to it, we get the system of linear equations (1.1) with the coefficient matrix

$$A = T \otimes I + I \otimes T, \tag{5.1}$$

where the equidistant stepsize $h = \frac{1}{m+1}$ is used in the discretization on both directions and the natural lexicographic ordering is employed to the unknowns. In addition, $\otimes$ denotes the Kronecker product, T is a tridiagonal matrix given by

$$T = \text{tridiag}(-1 - R_e, 2, -1 + R_e),$$

and

$$R_e = \frac{\delta h}{2}$$

is the mesh Reynolds number. We remark that here the first-order derivatives are also approximated by the centered difference scheme.

From Lemma A.1 in [2, Appendix], we know that

$$\begin{cases} \gamma_{\min} \equiv \lambda_{\min} = \min\limits_{1\le j,k\le m} \lambda_{j,k}(H) = 4(1-\cos(\pi h)), \\ \gamma_{\max} \equiv \lambda_{\max} = \max\limits_{1\le j,k\le m} \lambda_{j,k}(H) = 4(1+\cos(\pi h)) \end{cases}$$

and

$$\min_{1\le j,k\le m} |\lambda_{j,k}(S)| = 0, \quad \max_{1\le j,k\le m} |\lambda_{j,k}(S)| = 4R_e\cos(\pi h).$$

Therefore, the optimal parameter $\tilde{\alpha}$ that minimizes the upper bound $\sigma(\alpha)$ of the convergence factor $\rho(\mathcal{M}(\alpha))$ of the HSS iteration method is given by

$$\tilde{\alpha} \equiv \sqrt{\gamma_{\min}\gamma_{\max}} = \sqrt{\lambda_{\min}\lambda_{\max}} = 4\sin(\pi h);$$

see Theorem 2.1.

In Table 1 we list the experimental optimal parameter α (denoted by $\alpha_{\exp}$), the estimated optimal parameter α (denoted by $\alpha_{\rm est}$) determined by Estimation 5.1, the upper-bound minimizer $\tilde{\alpha}$, and the corresponding spectral radii $\rho(\mathcal{M}(\alpha))$ of the HSS iteration matrix $\mathcal{M}(\alpha)$ for $\alpha = \alpha_{\exp}, \alpha_{\rm est}$, and $\tilde{\alpha}$. From this table we see that $\tilde{\alpha}$ always overestimates $\rho(\mathcal{M}(\alpha))$ when compared with $\alpha_{\rm est}$ and that $\alpha_{\rm est}$ yields quite a good approximation to $\alpha_{\exp}$.

TABLE 1
α versus $\rho(\mathcal{M}(\alpha))$ when $m = 32$.

δ	10	50	100	500	1000
$\alpha_{\exp}$	0.5195	2.2129	3.5606	12.0063	17.6346
$\rho(\mathcal{M}(\alpha_{\exp}))$	0.7794	0.4414	0.4635	0.6357	0.7161
α_{est}	0.5967	2.7084	5.1536	10.2948	15.0075
$\rho(\mathcal{M}(\alpha_{est}))$	0.8055	0.4582	0.4771	0.6374	0.7179
$\tilde{\alpha}$	0.3802	0.3802	0.3802	0.3802	0.3802
$\rho(\mathcal{M}(\tilde{\alpha}))$	0.8312	0.8702	0.8839	0.8999	0.9030

TABLE 2
IT and CPU when $m = 32$ and $\varepsilon = 10^{-6}$.

δ		10	50	100	500	1000
$\alpha_{\exp}$	IT	70	38	36	58	79
	CPU	0.7414	0.3905	0.6986	1.1147	1.5510
α_{est}	IT	66	44	45	55	72
	CPU	0.6842	0.4568	0.9238	1.0780	1.3991

In Table 2 we list the number of iterations (denoted by "IT") and the elapsed CPU time in seconds (denoted by "CPU") of the HSS iteration method when it is applied to the nonsymmetric positive definite system of linear equations (1.1) with coefficient matrix (5.1). From this table we see that the numerical results produced with $\alpha_{\rm est}$ coincide with those using $\alpha_{\exp}$, and the match is quite pertinent.

Here and in the next example, we choose the right-hand side vector f such that the exact solution of the system of linear equations is $(1, 1, \ldots, 1)^T \in \mathbb{R}^n$. In addition,

all runs are initiated from the initial vector $x^{(0)} = 0$ and terminated if the current iteration satisfies

$$\frac{\|f - Ax^{(k)}\|_2}{\|f - Ax^{(0)}\|_2} \leq \varepsilon.$$

The experiments are run in MATLAB (version 6.1) with a machine precision 10^{-16}. The machine used is a Pentium-III 500 personal computer with 256M memory.

Then we use the following example of two-by-two block system of linear equations to further confirm the above observations.

Consider the system of linear equations (1.1) with the coefficient matrix

$$A = \begin{bmatrix} B & E \\ -E^T & 0.5I \end{bmatrix}, \tag{5.2}$$

where

$$B = \begin{bmatrix} I \otimes T_H + T_H \otimes I & 0 \\ 0 & I \otimes T_H + T_H \otimes I \end{bmatrix} \in \mathbb{R}^{2m^2 \times 2m^2}, \tag{5.3}$$

$$E = \begin{bmatrix} I \otimes F \\ F \otimes I \end{bmatrix} \in \mathbb{R}^{2m^2 \times m^2}, \tag{5.4}$$

and

$$T_H = \text{tridiag}(-1, 2, -1) \in \mathbb{R}^{m \times m}, \qquad F = \delta h \cdot \text{tridiag}(-1, 1, 0) \in \mathbb{R}^{m \times m}, \tag{5.5}$$

with $h = \frac{1}{m+1}$ the discretization meshsize; see [4].

For this example, we have $r = 2m^2$ and $s = m^2$. Hence, the total number of variables is $r + s = 3m^2$.

In Table 3 we list the experimental optimal parameter α_{exp}, the estimated optimal parameter α_{est} determined by Estimation 5.1, and the corresponding spectral radii $\rho(\mathcal{M}(\alpha))$ of the HSS iteration matrix $\mathcal{M}(\alpha)$ for $\alpha = \alpha_{\text{exp}}$ and α_{est}. Here, considering Theorem 4.1 and the two-by-two block structure of the matrix A in (5.2), we may apply the equations in Estimation 5.1 to the q which is either the largest or the smallest singular value (denoted, respectively, by q_1 or q_k) of the matrix E, and consider the two points $\alpha = \sqrt{\lambda_{\min}\lambda_{\max}}$ and $\alpha = \sqrt{q_1 q_k}$, to obtain a more accurate estimate to the optimal iteration parameter for the HSS iteration method. From this table we can see that α_{est} yields quite a good approximation to α_{exp}.

TABLE 3
α versus $\rho(\mathcal{M}(\alpha))$.

δ	10			100		
m	16	24	32	16	24	32
α_{exp}	0.7457	0.5087	0.3849	1.0340	0.6553	0.4639
$\rho(\mathcal{M}(\alpha_{\text{exp}}))$	0.8291	0.8812	0.9090	0.7700	0.8490	0.8912
α_{est}	0.7350	0.5013	0.3802	0.7350	0.5013	0.3802
$\rho(\mathcal{M}(\alpha_{est}))$	0.8304	0.8816	0.9091	0.8304	0.8816	0.9091

In Table 4 we list the number of iterations and the elapsed CPU time of the HSS iteration method when it is applied to the two-by-two block system of linear equations (1.1) with the coefficient matrix (5.2)–(5.5). From this table we can see that the numerical results produced with α_{est} coincide with those using α_{exp}, and the match is quite pertinent.

TABLE 4
IT and CPU when $\varepsilon = 10^{-6}$.

δ		10			100		
m		16	24	32	16	24	32
$\alpha_{\exp}$	IT	59	90	117	43	70	97
	CPU	0.4416	3.0069	15.2751	0.3531	2.3821	13.3629
α_{est}	IT	60	90	119	60	92	118
	CPU	0.4250	3.0139	15.2881	0.4617	3.1193	15.6046

6. Proof of Remark 3.3. The assertion concerning $\lambda_1 = \lambda_2$ and the assertion concerning (3.3) can be readily verified. We consider (3.4) and the reduced equation (3.5). Note that for $\alpha > 0$, the right-hand side of (3.4) is nonnegative, and the left-hand side is nonnegative on the interval $[\lambda_2, \lambda_1]$ only. So, all the positive solutions $\beta = \alpha^2$ of (3.5) lie in $[\lambda_2^2, \lambda_1^2]$. Define

$$f(\beta) = (\lambda_1^2 - \beta)(\beta - \lambda_2^2), \quad g_1(\beta) = (\beta - \lambda_1\lambda_2)^2, \quad g_2(\beta) = \frac{(\beta - q^2)^2}{(\beta + q^2)^2},$$

$$g(\beta) = g_1(\beta)g_2(\beta) \quad \text{and} \quad h(\beta) = f(\beta) - g(\beta)$$

for $\beta > 0$. Then β is a positive solution of (3.5) if and only if β is a positive solution of

$$h(\beta) = 0.$$

Obviously, the graph of $f(\beta)$ is a concave parabola which intersects the β-axis at $\beta = \lambda_2^2$ and λ_1^2, while the graph of $g_1(\beta)$ is a convex parabola touching the β-axis at $\beta = \lambda_1\lambda_2$. Note that $f(\beta) = g_1(\beta)$ has two solutions in the intervals $[\lambda_2^2, \lambda_1\lambda_2)$ and $((\lambda_1^2 + \lambda_2^2)/2, \lambda_1^2]$. The graph of $g(\beta)$ can be obtained from that of $g_1(\beta)$ by reducing the vertical height of each point by the factor $g_2(\beta) \in (0, 1)$. Thus, the solution of $f(\beta) = g_1(\beta)$ closer to λ_2^2 will move further left to a solution of $f(\beta) = g(\beta)$; the solution of $f(\beta) = g_1(\beta)$ closer to λ_1^2 will move further right to a solution of $f(\beta) = g(\beta)$.

We claim that $h(\beta)$ cannot have more than 2 positive solutions in $[\lambda_2^2, \lambda_1^2]$; equivalently, (3.5) cannot have more than 2 positive solutions in $[\lambda_2^2, \lambda_1^2]$.

First, we prove the claim if $q^2 < \lambda_2^2$ or $q^2 > \lambda_1^2$. Let $p(\beta)$ be the polynomial on the left-hand side of (3.5) divided by 2. Then the product of the roots of (3.5) is the constant term of $p(\beta)$ and equals the positive number $q^4\lambda_1^2\lambda_2^2$. So, if (3.5) has more than 2 positive roots, then it must have 4 positive roots, say, $\beta_1 \le \beta_2 \le \beta_3 \le \beta_4$. Suppose $p'(\beta)$ has zeros $\beta_1' \le \beta_2' \le \beta_3'$, $p''(\beta)$ has zeros $\beta_1'' \le \beta_2''$, and $p'''(\beta)$ has zero ${\beta_1}'''$. We have $\beta_1' \le \beta_1'' \le \beta_2' \le \beta_2'' \le \beta_3'$ and $\beta_1'' \le {\beta_1}''' \le \beta_2''$. Thus,

$$\beta_1' \le {\beta_1}''' \le \beta_3'.$$

Now, examining the constant terms of the polynomials $p'(\beta)/4$ and $p'''(\beta)/24$, we see that

$$\beta_1'\beta_2'\beta_3' = q^4(\lambda_1 + \lambda_2)^2/8 \tag{6.1}$$

and ${\beta_1}''' = (\lambda_1 + \lambda_2)^2/8$. If $q^2 < \lambda_2^2$, then $q^2 < \beta_1 \le \beta_1' \le \beta_2'$, and hence

$$\beta_1'\beta_2'\beta_3' > (q^2)(q^2)({\beta_1}''') = q^4(\lambda_1 + \lambda_2)^2/8,$$

which contradicts (6.1). Similarly, if $q^2 > \lambda_1^2$, then $q^2 > \beta_4 \geq \beta_3' \geq \beta_2'$, and hence

$$\beta_1'\beta_2'\beta_3' < \beta_1'''(q^2)(q^2) = q^4(\lambda_1 + \lambda_2)^2/8,$$

which again contradicts (6.1). Thus, our claim is proved in these cases.

Next, we assume that $q^2 \in [\lambda_2^2, \lambda_1^2]$. Then

$$g(\beta) = \frac{(\beta - q^2)^2(\beta - \lambda_1\lambda_2)^2}{(\lambda_1^2 - \beta)(\beta - \lambda_2^2)}$$

has two zeros in $[\lambda_2^2, \lambda_1^2]$, namely, $\beta = q^2$ and $\beta = \lambda_1\lambda_2$.

If $\lambda_2^2 \leq q^2 \leq \lambda_1\lambda_2$, then f is increasing on $[\lambda_2^2, (\lambda_1^2 + \lambda_2^2)/2]$ and $q^2 \leq \lambda_1\lambda_2 \leq (\lambda_1^2 + \lambda_2^2)/2$. So, f is increasing on $[\lambda_2^2, q^2]$. Since $g(\beta)$ is decreasing on $[\lambda_2^2, q^2]$ and the function $h(\beta)$ has different signs at λ_2^2 and q^2, it follows that $h(\beta) = 0$ has a root in $[\lambda_2\lambda_1, q^2]$. On $(q^2, (\lambda_1^2 + \lambda_2^2)/2]$, we have $f(\beta) > g_1(\beta) \geq g_1(\beta)g_2(\beta)$. Thus, $f(\beta) = g_1(\beta)g_2(\beta)$ has no root in this interval. Finally, on the interval $[(\lambda_1^2+\lambda_2^2)/2, \lambda_1^2]$, f is decreasing and $g(\beta)$ is increasing, and the function $h(\beta)$ assumes nonzero values with different signs at the end points. Thus, $f(\beta) = g_1(\beta)g_2(\beta)$ has a root in $((\lambda_1^2 + \lambda_2^2)/2, \lambda_1^2)$.

Suppose $\lambda_1\lambda_2 < q^2 \leq \lambda_1^2$. On $[\lambda_2^2, \lambda_1\lambda_2]$, f is increasing and $g(\beta)$ is decreasing. Moreover, since the function $h(\beta)$ assumes nonzero values with different signs at λ_2^2 and $\lambda_1\lambda_2$, it follows that $h(\beta) = 0$ has a root in $(\lambda_2^2, \lambda_1\lambda_2)$.

Suppose $q^2 < (\lambda_1^2 + \lambda_2^2)/2$. On $(\lambda_1\lambda_2, (\lambda_1^2 + \lambda_2^2)/2]$, we have $f(\beta) > g_1(\beta) \geq g_1(\beta)g_2(\beta)$. Thus, $f(\beta) = g_1(\beta)g_2(\beta)$ has no root in this interval. On the interval $[(\lambda_1^2+\lambda_2^2)/2, \lambda_1^2]$, f is decreasing and $g(\beta)$ is increasing, and the function $h(\beta)$ assumes nonzero values with different signs at the end points. Thus, $f(\beta) = g(\beta)$ has a root in $((\lambda_1^2 + \lambda_2^2)/2, \lambda_1^2)$.

Finally, if $q^2 \in [(\lambda_1^2 + \lambda_2^2)/2, \lambda_1^2]$, then for $\beta \in (\lambda_1\lambda_2, q^2)$ we have

$$(\beta - \lambda_1\lambda_2)^2/(\beta + q)^2 \in (0, 1),$$

and hence

$$f(\beta) > (\beta - q)^2 > g(\beta).$$

Thus, $f(\beta) = g(\beta)$ has no solution in $[(\lambda_1^2 + \lambda_2^2)/2, q^2)$. Now, on the interval $[q^2, \lambda_1^2]$, f is decreasing and $g(\beta)$ is increasing, and the function $h(\beta)$ assumes values with different signs at the end points. Thus, $f(\beta) = g(\beta)$ has a root in $[q^2, \lambda_1^2]$.

Acknowledgment. Part of this work was done during the first author's visits to the Department of Mathematics of The College of William and Mary during May–June 2004 and to Shanghai Key Laboratory of Contemporary Applied Mathematics of Fudan University. The authors are very much indebted to Jun-Feng Yin for his kind help with running the numerical results.

REFERENCES

[1] Z.-Z. Bai, G. H. Golub, L.-Z. Lu, and J.-F. Yin, *Block triangular and skew-Hermitian splitting methods for positive-definite linear systems*, SIAM J. Sci. Comput., 26 (2005), pp. 844–863.

[2] Z.-Z. Bai, G. H. Golub, and M. K. Ng, *Hermitian and skew-Hermitian splitting methods for non-Hermitian positive definite linear systems*, SIAM J. Matrix Anal. Appl., 24 (2003), pp. 603–626.

[3] Z.-Z. Bai, G. H. Golub, and M. K. Ng, *On Successive-Overrelaxation Acceleration of the Hermitian and Skew-Hermitian Splitting Iterations*, Tech. report SCCM-02-06, Stanford University, 2002; also available online at http://www-sccm.stanford.edu/wrap/pub-tech.html.
[4] Z.-Z. Bai, G. H. Golub, and J.-Y. Pan, *Preconditioned Hermitian and skew-Hermitian splitting methods for non-Hermitian positive semidefinite linear systems*, Numer. Math., 98 (2004), pp. 1–32.
[5] M. Benzi, M. J. Gander, and G. H. Golub, *Optimization of the Hermitian and skew-Hermitian splitting iteration for saddle-point problems*, BIT, 43 (2003), pp. 881–900.
[6] M. Benzi and G. H. Golub, *A preconditioner for generalized saddle point problems*, SIAM J. Matrix Anal. Appl., 26 (2004), pp. 20–41.
[7] M. Benzi and M. K. Ng, *Preconditioned iterative methods for weighted Toeplitz least squares problems*, SIAM J. Matrix Anal. Appl., 27 (2006), pp. 1106–1124.
[8] D. Bertaccini, G. H. Golub, S. Serra-Capizzano, and C. T. Possio, *Preconditioned HSS methods for the solution of non-Hermitian positive definite linear systems and applications to the discrete convection-diffusion equation*, Numer. Math., 99 (2005), pp. 441–484.
[9] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed., The Johns Hopkins University Press, Baltimore, MD, 1996.
[10] G. H. Golub and A. J. Wathen, *An iteration for indefinite systems and its application to the Navier–Stokes equations*, SIAM J. Sci. Comput., 19 (1998), pp. 530–539.
[11] M.-K. Ho and M. K. Ng, *Splitting iterations for circulant-plus-diagonal systems*, Numer. Linear Algebra Appl., 12 (2005), pp. 779–792.
[12] M. K. Ng, *Circulant and skew-circulant splitting methods for Toeplitz systems*, J. Comput. Appl. Math., 159 (2003), pp. 101–108.
[13] J.-Y. Pan, Z.-Z. Bai, and M. K. Ng, *Two-step waveform relaxation methods for implicit linear initial value problems*, Numer. Linear Algebra Appl., 12 (2005), pp. 293–304.
[14] J.-Y. Pan, M. K. Ng, and Z.-Z. Bai, *New preconditioners for saddle point problems*, Appl. Math. Comput., 172 (2006), pp. 762–771.
[15] Y. Saad, *Iterative Methods for Sparse Linear Systems*, PWS, Boston, 1996.
[16] V. Simoncini and M. Benzi, *Spectral properties of the Hermitian and skew-Hermitian splitting preconditioner for saddle point problems*, SIAM J. Matrix Anal. Appl., 26 (2004), pp. 377–389.
[17] R. S. Varga, *Matrix Iterative Analysis*, 2nd revised and expanded ed., Springer-Verlag, Berlin, 2000.
[18] H. A. van der Vorst, *Iterative Krylov Methods for Large Linear Systems*, Cambridge University Press, Cambridge, UK, 2003.
[19] C.-L. Wang and Z.-Z. Bai, *Sufficient conditions for the convergent splittings of non-Hermitian positive definite matrices*, Linear Algebra Appl., 330 (2001), pp. 215–218.
[20] D. M. Young, *Iterative Solution of Large Linear Systems*, Academic Press, New York, London, 1971.

SIAM J. SCI. COMPUT.
Vol. 28, No. 2, pp. 604–625
© 2006 Society for Industrial and Applied Mathematics

STEPSIZE CONTROL FOR MEAN-SQUARE NUMERICAL METHODS FOR STOCHASTIC DIFFERENTIAL EQUATIONS WITH SMALL NOISE*

WERNER RÖMISCH† AND RENATE WINKLER†

Abstract. A strategy for controlling the stepsize in the numerical integration of stochastic differential equations (SDEs) is presented. It is based on estimating the pth mean of local errors. The strategy leads to stepsize sequences that are identical for all computed paths. For the family of Euler schemes for SDEs with small noise, we derive computable estimates for the dominating term of the pth mean of local errors and show that the strategy becomes efficient for reasonable stepsizes. Numerical experience is reported for test examples including scalar SDEs and a stochastic circuit model.

Key words. stepsize control, stochastic differential equations, small noise, mean-square numerical methods

AMS subject classifications. 65C30, 60-80

DOI. 10.1137/030601429

1. Introduction.

We consider Itô stochastic differential equations (SDEs) of the type

$$x(t) = x_0 + \int_{t_0}^{t} f(x(s), s)ds + \int_{t_0}^{t} G(x(s), s)dw(s), \quad t \in \mathcal{J}, \tag{1.1}$$

where $\mathcal{J} = [t_0, T]$, $f : \mathbb{R}^n \times \mathcal{J} \to \mathbb{R}^n$, $G : \mathbb{R}^n \times \mathcal{J} \to \mathbb{R}^{n\times m}$ are continuous functions, and, moreover, f has continuous derivatives with respect to x. w is an m-dimensional Wiener process on a given probability space $(\Omega, \mathcal{F}, P)$ with filtration $(\mathcal{F}_t)_{t\in\mathcal{J}}$, and x_0 is a given $\mathcal{F}_{t_0}$-measurable initial value, independent of the Wiener process. It is assumed that there exists a pathwise unique, strong solution $x(\cdot)$.

We study mean-square and, more generally, pth mean convergent numerical methods for solving (1.1) based on time discretization. Our work is motivated by practical SDE models in circuit simulation [24, 27, 28] that do not satisfy the commutativity condition for G and are large scale with respect to n and m. As function calls are costly, we look at variable stepsize methods of low order and propose a new strategy for selecting stepsizes.

Several variable stepsize strategies for SDEs were developed during the last few years. Most of them are based on pathwise arguments and lead to pathwise different stepsize sequences. Such approaches often require a separate convergence analysis, as the available convergence theory for SDEs (e.g., in the mean-square or weak sense) is based on properties of certain expectations rather than paths which are typically nonsmooth objects. The strategies for pathwise controlling stepsizes differ for each approach. The classical paper [7] proposes a pathwise strategy by comparing results

*Received by the editors October 24, 2003; accepted for publication (in revised form) January 13, 2006; published electronically May 5, 2006. This work was supported by the German Federal Ministry of Education and Research under grant 03ROM3B3 and by the DFG Research Center MATHEON Mathematics for Key Technologies in Berlin.
http://www.siam.org/journals/sisc/28-2/60142.html
†Institut für Mathematik, Humboldt-Universität zu Berlin, 10099 Berlin, Germany (romisch@mathematik.hu-berlin.de, winkler@mathematik.hu-berlin.de).

of a given integration scheme with those of a higher order method. Hence, at least the higher order method requires the (approximate) computation of multiple Itô-integrals. The approaches in [18, 19, 4] are also based on a comparison of two Runge–Kutta schemes of different order. In [15] conditions are provided that imply mean-square convergence of the Euler–Maruyama scheme with pathwise different stepsize sequences. A different approach was developed in [12, 13, 23], where the authors obtain stepsize sequences that are (mean-square and pth mean, respectively) optimal for asymptotically small stepsizes.

In contrast to the above approaches we present a stepsize control that is based on estimates of the mean square or the pth mean of the local discretization error. This is justified by the fact that pth mean global errors can be estimated by the corresponding local ones provided that the method is stable and the exact initial value is given. In particular, we analyze the errors for the family of Euler–Maruyama schemes in the case of small noise. The local errors are represented in terms of stochastic integrals. The terms containing multiple stochastic integrals become so small that they are negligible for realistic stepsizes, and the low asymptotic order of convergence $1/2$ of the Euler–Maruyama schemes is observed only for stepsizes that are far too small to be used. We provide estimates for the mean square or pth mean of the dominating local error term that does not cost additional evaluations of the coefficients of the SDE or their derivatives.

Implementing a numerical scheme for the approximate integration of SDEs requires also a discretization of the sample space. One can compute only a finite number of paths. To implement the stepsize control we used a heuristic approach, where the mean square of the local terms was approximated by the information available from an ensemble of approximate solution paths that is computed simultaneously. This way our approach leads to stepsize sequences that are identical for all computed paths.

The stepsize strategy was implemented for the drift-implicit Euler method and extensively tested on a set of test examples. The choice of this drift-implicit method allows us to study the effects of the stepsize selection on the accuracy, i.e., the global discretization error, and on the convergence behavior of Newton's method for solving the implicit nonlinear equations simultaneously. In the case of step rejections, the method described in [19] is used to ensure that the correct Wiener paths are followed. Our numerical experience of the stepsize strategy confirms its usefulness and potential for SDEs with small noise, and also provides some information on its limitations. As expected by the analysis, it turns out that the stepsize control works successfully for ranges of stepsizes that lead to reasonably accurate results but that are still not so small that the asymptotic order of convergence $\mathcal{O}(h^{\frac{1}{2}})$ dominates the error. The latter phenomenon for SDEs with small noise was experimentally observed in [1, 6]. In the test examples we have used deterministic initial values. In general we think of applications in which the initial value has at least small variance. For deterministic differential equations with random initial values a pathwise stepsize control should be more efficient.

Our paper is organized as follows. In section 2 we introduce the class of discretization schemes that will be considered in this paper. We recall basic pth mean stability results and conditions for pth mean convergence stated in terms of the pth mean of the local discretization error and of its rate of convergence as the stepsize tends to zero. Starting from this observation we present, in section 3, a strategy for selecting pathwise identical stepsize sequences by estimating the pth mean of the local error. For the special case of integrating SDEs with small noise by the family of Euler

schemes, we provide computable estimates for the local errors in section 4. Finally, in section 5 we report on numerical experience of test runs of an implementation of the stepsize control for the implicit Euler scheme.

2. Numerical stability, consistency, and convergence of discretization methods for SDEs. We consider the drift-implicit discretization scheme of the form

$$x_\ell = x_{\ell-1} + \varphi(x_{\ell-1}, x_\ell; t_{\ell-1}, h_\ell) + \psi(x_{\ell-1}; t_{\ell-1}, h_\ell, I_{t_{\ell-1},h_\ell}), \quad \ell = 1, \dots, N, \tag{2.1}$$

for solving (1.1) on the deterministic grid $t_0 < t_1 < \cdots < t_N = T$ with stepsizes $h_\ell := t_\ell - t_{\ell-1}$, $\ell = 1, \dots, N$. Here, φ and ψ are functions defined on $\mathbb{R}^n \times \mathbb{R}^n \times \mathcal{T}$ and $\mathbb{R}^n \times \mathcal{T} \times \mathbb{R}^{m_I}$ with $\mathcal{T} := \{(t,h) : t, t+h \in \mathcal{J},\ h \in \mathbb{R}_+\}$, respectively, and mapping to $\mathbb{R}^n$. By $I_{t,h}$ we denote a vector of m_I multiple stochastic integrals of the form

$$I_{i_1 \dots i_k; t,h} = \int_t^{t+h} \int_t^{s_1} \cdots \int_t^{s_{k-1}} dw_{i_1}(s_k) dw_{i_2}(s_{k-1}) \cdots dw_{i_k}(s_1),$$

where the indices $i_1, \dots, i_k$ are in $\{0, 1, \dots, m\}$, k is bounded by a certain finite order k_{max}, and $dw_0(s)$ corresponds to ds.

For example, the family of drift-implicit Euler schemes, sometimes also called stochastic θ-methods, is of the form

$$x_\ell := x_{\ell-1} + h_\ell(\theta f(x_\ell, t_\ell) + (1-\theta) f(x_{\ell-1}, t_{\ell-1})) + G(x_{\ell-1}, t_{\ell-1}) \Delta w_\ell, \quad \ell = 1, \dots, N, \tag{2.2}$$

where $\theta \in [0,1]$, and $\Delta w_\ell := w(t_\ell) - w(t_{\ell-1}) = (I_{i;t_{\ell-1},h_\ell})_{i=1}^m$. Hence, one has $k_{max} := 1$, $m_I := m$, and

$$\varphi(z, x; t, h) := h(\theta f(x, t+h) + (1-\theta) f(z, t)),$$
$$\psi(z; t, h, I_{t,h}) := G(z,t)(w(t+h) - w(t)) = \sum_{j=1}^m g_j(z,t) \int_t^{t+h} dw_j(s),$$

where $g_j(z,t)$, $j = 1, \dots, m$, are the columns of the matrix $G(z,t)$.

The family of drift-implicit Milstein schemes differs from the Euler schemes by an additional correction term for the stochastic part. The Milstein schemes are described by the same function φ, and $k_{max} := 2$, $m_I := m + m^2$, and

$$\psi(z; t, h, I_{t,h}) := G(z,t) \Delta w_{t,h} + \sum_{j=1}^m (g_{j_x} G)(z,t) I_{(j);t,h}, \tag{2.3}$$

where $\Delta w_{t,h} := w(t+h) - w(t) = (I_{i;t,h})_{i=1}^m$, and $I_{(j);t,h} := (I_{j,i;t,h})_{i=1}^m$.

For measuring errors in the discretization scheme we use the norm for pth order integrable random variables $z \in L_p(\Omega, \mathbb{R}^n)$, i.e., $\|z\|_{L_p} := (\mathbb{E}|z|^p)^{1/p}$, where the exponent $p \geq 1$ is properly chosen in the sense that the initial value x_0 has a pth order moment and in that it fits into the unknown statistical parameters of $x(\cdot)$, which have to be computed approximately. We start our analysis by stating a result concerning the pth mean stability of (2.1), which extends the fundamental result of Milstein [20, 21]. Its proof is given in [24, 27].

THEOREM 2.1. [illegible] $p \geq 1$ [illegible] x_0 [illegible] p [illegible]
(2.1) [illegible]

- [illegible] $z, \tilde{z}, x, \tilde{x} \in \mathbb{R}^n$ $(t,h) \in \mathcal{T}$ $h \le h^1$ [illegible]

  (A1) $$|\varphi(z,x;t,h) - \varphi(\tilde{z},\tilde{x};t,h)| \le h(L_1|z-\tilde{z}| + L_2|x-\tilde{x}|)$$

- [illegible] h^1, L_1, L_2
- [illegible] $(t,h) \in \mathcal{T}$ $h \le h^1$ [illegible] $\mathcal{F}_t$ [illegible] $y, \tilde{y}$ [illegible]

  (A2) $$\mathbb{E}(\psi(y;t,h,I_{t,h}) - \psi(\tilde{y};t,h,I_{t,h})|\mathcal{F}_t) = 0,$$

  (A3) $$\mathbb{E}(|\psi(y;t,h,I_{t,h}) - \psi(\tilde{y};t,h,I_{t,h})|^p|\mathcal{F}_t) \le h^{\frac{p}{2}} L_3^p |y-\tilde{y}|^p,$$

  (A4) $$\mathbb{E}|\psi(0;t,h,I_{t,h})|^p < \infty$$

[illegible] $L_3 > 0$
[illegible] $a \ge 1$ [illegible] $h^0 > 0$ [illegible]
$S > 0$ [illegible] $\{t_0, t_1, \dots, t_N\}$ [illegible]
$h := \max_{\ell=1,\dots,N} h_\ell \le h^0$ [illegible] $h \cdot N \le a \cdot (T - t_0)$:
[illegible] $\mathcal{F}_{t_0}$ [illegible] $x_0^*, \tilde{x}_0$ [illegible] p [illegible]
[illegible] $\ell \in \{1, \dots, N\}$ [illegible] $\mathcal{F}_{t_\ell}$ [illegible] $d_\ell^*, \tilde{d}_\ell$ [illegible] p [illegible]

(2.4)
$$\tilde{x}_\ell = \tilde{x}_{\ell-1} + \varphi(\tilde{x}_{\ell-1}, \tilde{x}_\ell; t_{\ell-1}, h_\ell) + \psi(\tilde{x}_{\ell-1}; t_{\ell-1}, h_\ell, I_{t_{\ell-1},h_\ell}) + \tilde{d}_\ell, \quad \ell = 1, \dots, N,$$

[illegible] $\{\tilde{x}_\ell\}_{\ell=0}^N$ [illegible]
[illegible] $\{x_\ell^*\}_{\ell=1}^N$ $\{\tilde{x}_\ell\}_{\ell=0}^N$ [illegible]
$\{d_\ell^*\}_{\ell=1}^N$ $\{\tilde{d}_\ell\}_{\ell=1}^N$ [illegible] $d_\ell := d_\ell^* - \tilde{d}_\ell$ [illegible] $d_\ell = s_\ell + r_\ell$ [illegible]
$\mathbb{E}(s_\ell|\mathcal{F}_{t_{\ell-1}}) = 0$:

$$\mathbb{E} \max_{\ell=1,\dots,N} |x_\ell^* - \tilde{x}_\ell|^p \le S^p \left(\mathbb{E}|x_0^* - \tilde{x}_0|^p + \frac{\max\limits_{\ell=1,\dots,N} \mathbb{E}|s_\ell|^p}{h^{\frac{p}{2}}} + \frac{\mathbb{E} \max\limits_{\ell=1,\dots,N} |r_\ell|^p}{h^p} \right), \tag{2.5}$$

$$\max_{\ell=1,\dots,N} \mathbb{E}|x_\ell^* - \tilde{x}_\ell|^p \le S^p \left(\mathbb{E}|x_0^* - \tilde{x}_0|^p + \frac{\max\limits_{\ell=1,\dots,N} \mathbb{E}|s_\ell|^p}{h^{\frac{p}{2}}} + \frac{\max\limits_{\ell=1,\dots,N} \mathbb{E}|r_\ell|^p}{h^p} \right). \tag{2.6}$$

Extracting the pth root from (2.6) yields the stability inequality

(2.7)
$$\max_{\ell=1,\dots,N} \|x_\ell^* - \tilde{x}_\ell\|_{L_p} \le S \left(\|x_0^* - \tilde{x}_0\|_{L_p} + \max_{\ell=1,\dots,N} \|s_\ell\|_{L_p}/h^{\frac{1}{2}} + \max_{\ell=1,\dots,N} \|r_\ell\|_{L_p}/h \right).$$

The scheme (2.1) is called p[illegible] if it satisfies properties (2.5) and (2.6), respectively, in the above result. Furthermore, it is called p[illegible] of order $\gamma > 0$ if the [illegible] l_ℓ at step ℓ, i.e.,

(2.8)
$$l_\ell := x(t_\ell) - x(t_{\ell-1}) - \varphi(x(t_{\ell-1}), x(t_\ell); t_{\ell-1}, h_\ell) - \psi(x(t_{\ell-1}); t_{\ell-1}, h_\ell, I_{t_{\ell-1},h_\ell}),$$

satisfies the properties

$$\|l_\ell\|_{L_p} \le c \cdot h_\ell^{\gamma+\frac{1}{2}} \quad \text{and} \quad \|\mathbb{E}(l_\ell|\mathcal{F}_{t_{\ell-1}})\|_{L_p} \le \bar{c} \cdot h_\ell^{\gamma+1}, \quad \ell = 1, \dots, N,$$

with constants $c, \bar{c} > 0$ depending only on the SDE and its solution. Clearly, the local discretization error arises by inserting the exact solution $x(\cdot)$ into the numerical scheme.

By the global errors e_ℓ we denote the difference

$$e_\ell := x(t_\ell) - x_\ell$$

of the exact and approximate solution at time t_ℓ. The discretization scheme (2.1) for (1.1) is called p... of order $\gamma > 0$ if the global error $e_\ell := x(t_\ell) - x_\ell$ satisfies the property

$$\max_{\ell=1,\dots,N} \|e_\ell\|_{L_p} \le C \cdot h^\gamma, \quad \text{where } h := \max_{\ell=1,\dots,N} h_\ell,$$

with a grid-independent constant $C > 0$.

THEOREM 2.2. ... (2.1) ... (1.1) ... p... ... $\gamma > 0$ ... 2.1 ... (2.1) ... p... ... γ ... $x(t_\ell)$ ... $\tilde{x}_\ell$ ... (2.4) ...

$$\max_{\ell=1,\dots,N} \|x(t_\ell) - \tilde{x}_\ell\|_{L_p} \le S\left((c+\bar{c})h^\gamma + \max_{\ell=1,\dots,N} \delta_\ell/h^{1/2} + \max_{\ell=1,\dots,N} \bar{\delta}_\ell/h\right), \tag{2.9}$$

... $\delta_\ell := \|\tilde{d}_\ell\|_{L_p}$, $\bar{\delta}_\ell := \|\mathbb{E}(\tilde{d}_\ell|\mathcal{F}_{t_{\ell-1}})\|_{L_p}$, ... $\tilde{d}_\ell$ ... (2.4).

... The assertion follows by applying the triangle inequality

$$\max_{\ell=1,\dots,N} \|x(t_\ell) - \tilde{x}_\ell\|_{L_p} \le \max_{\ell=1,\dots,N} \|x(t_\ell) - x_\ell\|_{L_p} + \max_{\ell=1,\dots,N} \|x_\ell - \tilde{x}_\ell\|_{L_p}$$

and the stability estimate (2.5) once to $x(t_\ell)$ related to the perturbations l_ℓ and once again to $\tilde{x}_\ell$ related to the perturbations $\tilde{d}_\ell$. The pth mean convergence follows as a special case of (2.9) for $d_\ell^* = l_\ell$, $\tilde{d}_\ell = 0$. By means of $r_\ell = \mathbb{E}(l_\ell|\mathcal{F}_{t_{\ell-1}})$, a suitable splitting $d_\ell = l_\ell = s_\ell + r_\ell$ is realized. □

The general results immediately apply to well-known schemes for SDEs. We illustrate this for the families of drift-implicit Euler and Milstein schemes. Both schemes fit into the framework of (2.1). By checking the assumptions of Theorem 2.1 we observe that both are pth mean stable: (A1) follows from the Lipschitz continuity of the drift coefficient f, (A2) is satisfied due to the explicit, nonanticipative discretization of the diffusion term, (A3) follows from the Lipschitz continuity of the diffusion coefficient G (and in the case of the Milstein scheme of the functions $g_{j_x}G$), and (A4) is a more technical assumption, which is satisfied since the function $G(0,\cdot)$ (and the functions $(g_{j_x}G)(0,\cdot)$) are bounded on the compact interval $\mathcal{J}$. Summarizing we have the following.

PROPOSITION 2.3. ... f ... G ... x ... (2.2) ... p... ... g_{j_x} $j = 1, \dots, m$ ... $g_{j_x} \cdot G$ ... x ... (2.3) ... p...

From the literature (see, e.g., [21]) it is known that the Euler schemes (2.2) are mean-square consistent of order 1/2 if, in addition, the coefficients are Hölder continuous with exponent 1/2 with respect to t. The Milstein schemes are mean-square consistent of order 1 if the functions f and G are sufficiently smooth. Appealing to Theorem 2.2 then provides the known mean-square convergence of the Euler schemes of order 1/2 and of the Milstein schemes of order 1.

3. Stepsize control based on the *p*th mean of local errors. For the efficient numerical integration of applied nonlinear SDEs, a reasonable stepsize control is indispensable. The problem was addressed in a number of recent papers, e.g., [3, 4, 7, 16, 18, 19]. Most of them suggest individual stepsize sequences for every path. Our approach is rigorously based on the stability and convergence results presented in the previous section. It leads to adaptive stepsize sequences that are uniform for all computed paths.

By means of the stability inequality (2.7) we know that the pth mean of the global errors $e_\ell := x(t_\ell) - x_\ell$ can be estimated by a term that is proportional to the pth mean of the local errors l_ℓ. Here, we assume that the initial value x_0 is given exactly. Therefore, a natural approach consists of controlling the local error term

$$\eta_\ell := \max\{\|s_\ell\|_{L_p}/h_\ell^{1/2}\,, \|r_\ell\|_{L_p}/h_\ell\}, \text{ where } l_\ell = s_\ell + r_\ell,\ \mathbb{E}(s_\ell|\mathcal{F}_{t_{\ell-1}}) = 0, \tag{3.1}$$

according to a given tolerance. Controlling this term requires some insight into its behavior. Clearly, we have $\eta_\ell = \mathcal{O}(h_\ell^\gamma)$ for a method that is pth mean convergent of order γ. However, for problems with small noise and stepsizes that cannot be considered asymptotically small and are of interest, η_ℓ may even be dominated by a term $\kappa_\ell \cdot h_\ell^{\bar\gamma}$, where $\bar\gamma \geq \gamma$ and κ_ℓ is a slowly varying factor (cf. section 4). The insight into the behavior of η_ℓ should lead to expressions for the error constant in the form $\kappa_\ell = \|k_\ell\|_{L_p}$ with $k_\ell = \chi(x_{\ell-1}, x_\ell, t_{\ell-1}, t_\ell)$.

Given that, for an interesting range of stepsizes h_ℓ, the local error term η_ℓ is dominated by

$$\eta_\ell \leq \kappa_\ell \cdot h_\ell^{\bar\gamma}, \quad \kappa_\ell = \|k_\ell\|_{L_p} = \|\chi(x_{\ell-1}, x_\ell, t_{\ell-1}, t_\ell)\|_{L_p}; \tag{3.2}$$

the stepsizes should be chosen according to the following conceptual algorithm.

ALGORITHM 3.1. Given the initial value $x_0 \in L_p(\Omega)$ at t_0, an initial stepsize h_1 and a tolerance tol, set $\ell := 1$.

(1) Solve the system

$$x_\ell = x_{\ell-1} + \varphi(x_{\ell-1}, x_\ell; t_{\ell-1}, h_\ell) + \psi(x_{\ell-1}; t_{\ell-1}, h_\ell, I_{t_{\ell-1},h_\ell})$$

for $x_\ell \in L_p(\Omega)$.

(2) Set

$$\bar\eta_\ell := h_\ell^{\bar\gamma}\|\chi(x_{\ell-1}, x_\ell, t_{\ell-1}, t_\ell)\|_{L_p}. \tag{3.3}$$

(3) Apply a control strategy that aims at matching the tolerance multiplied by a safety factor, say 0.8, with the proposed new stepsize. For example, the elementary control leads to

$$\frac{h_{new}}{h_\ell} := \left(\frac{0.8 \cdot \text{tol}}{\bar\eta_\ell}\right)^{1/\bar\gamma} \tag{3.4}$$

for $\ell \geq 2$; the proportional integral control PI34 (cf. [10, 25]) leads to

$$\frac{h_{new}}{h_\ell} := \left(\frac{0.8 \cdot \text{tol}}{\bar\eta_\ell}\right)^{0.3/\bar\gamma} \left(\frac{\bar\eta_{\ell-1}}{\bar\eta_\ell}\right)^{0.4/\bar\gamma}. \tag{3.5}$$

(4) If $\bar\eta_\ell \leq$ tol, accept the step.
If $t_\ell \geq T$, stop, else set $\ell := \ell + 1$, $h_\ell := h_{new}$ and go to (1).
If $\bar\eta_\ell >$ tol, reject the step and repeat it with the smaller stepsize $h_\ell := h_{new}$ that results from (3.4).

We emphasize that a stepsize sequence generated by Algorithm 3.1 is deterministic, since $\|\chi(x_{\ell-1}, x_\ell, t_{\ell-1}, t_\ell)\|_{L_p}$ is deterministic, though $x_{\ell-1}, x_\ell$ are random variables. Hence, Theorems 2.1 and 2.2 apply. If Algorithm 3.1 is realized and the computed quantities $\bar\eta_\ell$ are really dominating the local error term η_ℓ, say, e.g., $\eta_\ell \le 1.2\,\bar\eta_\ell$, then we have

$$\max_{\ell=1,\dots,N} \|x(t_\ell) - x_\ell\|_{L_p} \le 2.4\, S \cdot \mathrm{tol}$$

by the pth mean stability inequality (2.7) following Theorem 2.1. Even if the actual errors η_ℓ show a smaller order than the quantities $\bar\eta_\ell$ used, and only a relation like $\eta_\ell \le \mathrm{const} \cdot \bar\eta_\ell^{\,\beta}$, $\beta \in (0,1)$, is true, we still have

$$\max_{\ell=1,\dots,N} \|x(t_\ell) - x_\ell\|_{L_p} \le 2\,\mathrm{const} \cdot S \cdot \mathrm{tol}^{\beta}\,.$$

Intending to make use of this theoretical result for practical implementations, one faces several questions.

First, how large is the stability constant S? Of course, it is problem dependent. A more detailed look into the proof of Theorem 2.1, together with Proposition 2.3, shows that S behaves essentially as $e^{L(T-t_0)}$, where L is a Lipschitz constant for the drift and diffusion coefficients f and G (and of $(g_j)'_x \cdot G$ for the Milstein schemes) of the problem. By analogy with deterministic ODEs one may consider the problem as nonstiff and numerically well-posed, as long as $e^{L(T-t_0)}$ is moderate. Modifications for stiff problems are desirable, but go beyond the scope of this paper.

Second, how can the observed order $\bar\gamma$ and the error function χ be determined, and what conditions guarantee an estimate like $\eta_\ell \le 1.2\,\bar\eta_\ell$? These questions are discussed for the family of Euler–Maruyama schemes in the next section.

Finally, we discuss implementation issues of the conceptual algorithm, Algorithm 3.1. Of course, any implementation requires a finite number of realizations of the random variables. Here, we consider an ensemble of M paths starting from M samples $x_0^1, \dots, x_0^M$ of the initial value. Starting from $\ell = 1$, the next elements $x_\ell^1, \dots, x_\ell^M$ of the M paths are computed by solving the nonlinear equations

$$x_\ell^i = x_{\ell-1}^i + \varphi(x_{\ell-1}^i, x_\ell^i; t_{\ell-1}, h_\ell) + \psi(x_{\ell-1}^i; t_{\ell-1}, h_\ell, I^i_{t_{\ell-1},h_\ell}), \quad i = 1, \dots, M,$$

where $I^i_{t_{\ell-1},h_\ell}$ are samples of the corresponding stochastic integrals.

Next, the L_p-norm $\|\chi(x_{\ell-1}, t_{\ell-1}, x_\ell, t_\ell)\|_{L_p}$ in step (2) of Algorithm 3.1 is estimated by using the M samples of $x_{\ell-1}$ and x_ℓ, namely, by

$$\bar\kappa_\ell := \left(\frac{1}{M} \sum_{i=1}^{M} |\chi(x_{\ell-1}^i, x_\ell^i, t_{\ell-1}, t_\ell)|^p \right)^{\frac{1}{p}}. \tag{3.6}$$

Set

$$\hat\eta_\ell := h_\ell^{\bar\gamma} \bar\kappa_\ell \tag{3.7}$$

and perform steps (3) and (4) of Algorithm 3.1 with $\bar\eta_\ell$ replaced by $\hat\eta_\ell$. In the case of step rejections, the information computed so far is stored and used to compute intermediate values according to the strategy in [18, 19]. For a scheme (2.1), which uses only the Wiener increments $\Delta w_\ell = w(t_\ell) - w(t_{\ell-1})$, the computation of the intermediate values of the Wiener process is done as follows: Given $\Delta w_h := w(t+h) -$

$w(t)$ for some $t \in \mathbb{R}_+$ and $h > 0$, and $h = h_1 + h_2$, $h_1 > 0$, $h_2 > 0$, the Wiener increments

$$\Delta w_{h_1} = w(t + h_1) - w(t) \quad \text{and} \quad \Delta w_{h_2} = w(t + h_1 + h_2) - w(t + h_1)$$

are simulated according to the formulas

$$\Delta w_{h_1} = \frac{h_1}{h}\Delta w_h + \sqrt{\frac{h_1 h_2}{h}}\nu, \quad \Delta w_{h_2} = \frac{h_2}{h}\Delta w_h - \sqrt{\frac{h_1 h_2}{h}}\nu, \quad \nu \sim N(0, I_m).$$

In this way, the estimate (3.7) of (3.3) leads to estimates of the next stepsize and, hence, of the whole grid $t_0, t_1, \ldots, t_N$. Though all computed paths $(x_0^i, x_1^i, \ldots, x_N^i)$, $i = 1, \ldots, M$, are determined by using the same stepsizes $h_1, \ldots, h_N$, the stepsize sequence is no longer deterministic and, due to possible step rejections, the computed grid-points do not need to be stopping times. The estimate of the grid depends on the M paths, and its quality clearly depends on the sample size M as well as on the smallness of the noise. The resulting implementable algorithm represents a theory-based heuristic stepsize control.

The numerical results in section 5 show that a relatively small number $M = 100$ already provides good results for SDEs with small noise.

4. Local error estimates for the family of Euler schemes for SDEs with small noise. There are important applications of SDEs with small noise, where the diffusion coefficients are orders of magnitude smaller than the drift coefficients. For such problems the asymptotic order of convergence is too pessimistic for a reasonable range of stepsizes. Special numerical methods are constructed in [22], taking into account the smallness of the stochastic parts in such systems. Here, we will deal with the family of Euler schemes and present an efficient stepsize control based on the pth mean of local errors.

Following along the lines of [22] we let the SDE with small noise be of the form

$$x(t) = x_0 + \int_{t_0}^{t} f(x(s), s)ds + \int_{t_0}^{t} \epsilon\tilde{G}(x(s), s)dw(s), \quad t \in \mathcal{J}, \tag{4.1}$$

where $f : \mathbb{R}^n \times \mathcal{J} \to \mathbb{R}^n$, $\tilde{G} : \mathbb{R}^n \times \mathcal{J} \to \mathbb{R}^{n\times m}$ are functions satisfying the assumptions introduced in section 1 for f and G, and ϵ is a small parameter.

The family of drift-implicit Euler schemes with parameter $\theta \in [0, 1]$ for solving (4.1) on the deterministic grid $t_0 < t_1 < \cdots < t_N = T$ with stepsizes $h_\ell := t_\ell - t_{\ell-1}$, $\ell = 1, \ldots, N$, has the form

$$x_\ell = x_{\ell-1} + h_\ell\big(\theta f(x_\ell, t_\ell) + (1 - \theta)f(x_{\ell-1}, t_{\ell-1})\big) + \epsilon\tilde{G}(x_{\ell-1}, t_{\ell-1})\Delta w_\ell, \quad \ell = 1, \ldots, N, \tag{4.2}$$

where $\Delta w_\ell = w(t_\ell) - w(t_{\ell-1}) \sim N(0, h_\ell I_m)$.

In order to derive estimates for the local discretization error l_ℓ of (4.2), we first establish, similarly to [22], a representation of l_ℓ in terms of certain multiple stochastic integrals obtained by the Itô–Taylor expansion. The L_p-norm of these stochastic integrals is then characterized in terms of $\mathcal{O}(h_\ell^{k/2}\epsilon^q)$ for some $k, q \in \mathbb{N} \cup \{0\}$. Finally, we discuss which terms are dominating for interesting ranges of stepsizes and present computable estimates for the dominating terms.

4.1. Estimating local errors by Itô–Taylor expansion. In order to characterize the conditions on f and $\tilde{G}$ that are needed in the following, we introduce the classes C_L and $C^{s,s-1}$, $s \in \mathbb{N}$, of functions from $\mathbb{R}^n \times \mathcal{J}$ to $\mathbb{R}^n$. The class C_L contains all continuous functions that are Lipschitz continuous with a uniform constant with respect to the first variable. $C^{s,s-1}$ is the class of all functions having continuous partial derivatives up to order $s-1$ and, in addition, continuous partial derivatives of order s with respect to the first variable.

Let $x(\cdot)$ be a solution of the SDE (4.1) and y be a function in $C^{2,1}$. Then Itô's formula provides the expansion

$$y(x(t),t) - y(x_0,t_0) = \int_{t_0}^{t} \left(y_t + y_x f + \epsilon^2 \frac{1}{2} \sum_{r=1}^{m} \sum_{i,j=1}^{n} y_{x_i x_j} \tilde{g}_{ri} \tilde{g}_{rj} \right)(x(s),s)ds + \epsilon \sum_{r=1}^{m} \int_{t_0}^{t} (y_x \tilde{g}_r)(x(s),s) dw_r(s), \quad t \in \mathcal{J}. \tag{4.3}$$

Following [22] we introduce $m+1$ operators Λ_0 and Λ_r, $r = 1, \dots, m$, defined on $C^{2,1}$ and $C^{1,0}$, respectively, by

$$\Lambda_0 y = y_t + y_x f + \epsilon^2 \frac{1}{2} \sum_{r=1}^{m} \sum_{i,j=1}^{n} y_{x_i x_j} \tilde{g}_{ri} \tilde{g}_{rj}, \quad \Lambda_r y = y_x \tilde{g}_r, \quad r = 1, \dots, m.$$

Then the Itô formula (4.3) reads

$$y(x(t),t) - y(x_0,t_0) = \int_{t_0}^{t} \Lambda_0 y(x(s),s) ds + \epsilon \sum_{r=1}^{m} \int_{t_0}^{t} \Lambda_r y(x(s),s) dw_r(s), \quad t \in \mathcal{J}. \tag{4.4}$$

For $y \in C_L$, and similarly to section 2, we denote multiple stochastic integrals over intervals $[t, t+h] \subseteq \mathcal{J}$ by

$$I_{i_1 \dots i_j;t,h}(y) = \int_t^{t+h} \int_t^{s_1} \cdots \int_t^{s_{j-1}} y(x(s_j), s_j) dw_{i_1}(s_j) \cdots dw_{i_{j-1}}(s_2) dw_{i_j}(s_1),$$

where $i_1, \dots, i_j$ take values in $\{0, \dots, m\}$, and $dw_0(s)$ is understood to mean ds. As the function y has linear growth with respect to the first variable, the stochastic integrals are well defined.

LEMMA 4.1. [illegible] $p \geq 1$ [illegible] x_0 [illegible] p [illegible] $(t,h) \in \mathcal{T}$ [illegible] $i_j \in \{1, \dots, m\}$ $j = 1, \dots, k$ [illegible] $y \in C_L$ [illegible]

$$\mathbb{E}(I_{i_1 \dots i_k;t,h}(y) | \mathcal{F}_t) = 0 \quad \text{if } i_j \neq 0 \text{ for } j \in \{1, \dots, k\},$$

$$\|\mathbb{E}(I_{i_1 \dots i_k;t,h}(y) | \mathcal{F}_t)\|_{L_p} \leq \|I_{i_1 \dots i_k;t,h}(y)\|_{L_p} = \mathcal{O}(h^{\sum_{j=1}^{k} \nu_j}), \quad \nu_j = \begin{cases} 1, & i_j = 0, \\ \frac{1}{2}, & i_j \neq 0. \end{cases}$$

[illegible]. The first property is well known. The first estimate in the second assertion is due to properties of the conditional expectation. For $p = 2$ the second part is proved in [21, Lemma 2.1]. For $1 \leq p < 2$ it is a consequence of the estimate $\|\cdot\|_{L_p} \leq \|\cdot\|_{L_2}$.

Now, let $p > 2$. For $i_1 = 0$ we obtain by Hölder's inequality that

$$\|I_{0,i_2 \dots i_k;t,h}(y)\|_{L_p}^p = \mathbb{E}(|I_{0,i_2 \dots i_k,0;t,h}(y)|^p) \leq \left(\mathbb{E} \int_t^{t+h} |I_{i_2 \dots i_k;t,s_1-t}(y)| ds_1 \right)^p$$

$$\leq h^{\frac{p}{q}} \int_t^{t+h} \mathbb{E}(|I_{i_2 \dots i_k;t,s_1-t}(y)|^p) ds_1,$$

where $\frac{1}{q}+\frac{1}{p}=1$. Hence, for $\|I_{0,i_2\dots i_k;t,h}(y)\|^p_{L_p}$ we obtain the order $\mathcal{O}(h^{\frac{p}{q}+1})=\mathcal{O}(h^p)$.

For $i_1\neq 0$ we make use of estimates for the pth mean of stochastic integrals (see [8, section 1.4, Theorem 6], [17, section 1.7, Theorem 7.1]) and have

$$\|I_{i_1\dots i_k;t,h}(y)\|^p_{L_p}\leq\left(\frac{1}{2}p(p-1)\right)^{\frac{p}{2}}h^{\frac{p-2}{2}}\int_t^{t+h}\mathbb{E}(|I_{i_2\dots i_k;t,t-s_1}(y)|^p)ds_1.$$

Hence, in this case we obtain the order $\mathcal{O}(h^{\frac{p-2}{2}+1})=\mathcal{O}(h^{\frac{p}{2}})$.

Repeating these arguments successively, and using that the function y has linear growth and, thus, that $y(x(\cdot),\cdot)$ has a finite pth mean, completes the proof. □

PROPOSITION 4.2. [illegible] $f\in C^{4,3}$ [illegible] $\tilde g_r\in C^{2,1}$ $r=1,\dots,m$ [illegible] $\Lambda_0\Lambda_0 f$ $\Lambda_0\tilde g_r$ $\Lambda_r\Lambda_0 f$ $\Lambda_r f$ [illegible] $\Lambda_k\tilde g_r$ $k,r=1,\dots,m$ [illegible] C_L [illegible] l_ℓ ([illegible] (2.8)) [illegible] (4.2) [illegible] ℓ [illegible]

$$l_\ell=s_\ell+r_\ell \quad [illegible] \quad \mathbb{E}(s_\ell|\mathcal{F}_{t_{\ell-1}})=0$$

[illegible]

$$\|r_\ell\|_{L_p}/h_\ell=h_\ell\left|\theta-\frac{1}{2}\right|\|(\Lambda_0 f)_{t_{\ell-1}}\|_{L_p}+\mathcal{O}(h_\ell^2),$$
$$\|s_\ell\|_{L_p}/h_\ell^{1/2}=\epsilon\mathcal{O}(h_\ell)+\epsilon^2\mathcal{O}(h_\ell^{1/2}).$$

[illegible]. For $y\in C_L$ we make use of the following abbreviations:

$$y_s:=y(x(s),s),\quad I_{i_1\dots i_j}(y)=I_{i_1\dots i_j;t_{\ell-1},h_\ell}(y).$$

By reformulating the local error and by expanding all of its components at the pair $(x(t_{\ell-1}),t_{\ell-1})$ using (4.4) and the smoothness properties $f,\tilde g_r\in C^{2,1}$, $r=1,\dots,m$, we obtain

$$\begin{aligned}
l_\ell &= x(t_\ell)-x(t_{\ell-1})-h_\ell\big(\theta f(x(t_\ell),t_\ell)+(1-\theta)f(x(t_{\ell-1}),t_{\ell-1})\big)-\epsilon\tilde G(x(t_{\ell-1}),t_{\ell-1})\Delta w_\ell\\
&=\int_{t_{\ell-1}}^{t_\ell}f_s ds-h_\ell\big(\theta f_{t_\ell}+(1-\theta)f_{t_{\ell-1}}\big)+\int_{t_{\ell-1}}^{t_\ell}\epsilon\tilde G_s dw(s)-\epsilon\tilde G_{t_{\ell-1}}\Delta w_\ell\\
&=\int_{t_{\ell-1}}^{t_\ell}\left\{f_{t_{\ell-1}}+\int_{t_{\ell-1}}^{s}(\Lambda_0 f)_\tau d\tau+\epsilon\sum_{r=1}^m\int_{t_{\ell-1}}^{s}(\Lambda_r f)_\tau dw_r(\tau)\right\}ds\\
&\quad-\theta h_\ell\left\{f_{t_{\ell-1}}+\int_{t_{\ell-1}}^{t_\ell}(\Lambda_0 f)_\tau d\tau+\epsilon\sum_{r=1}^m\int_{t_{\ell-1}}^{t_\ell}(\Lambda_r f)_\tau dw_r(\tau)\right\}-(1-\theta)h_\ell f_{t_{\ell-1}}\\
&\quad+\epsilon\sum_{r=1}^m\int_{t_{\ell-1}}^{t_\ell}\left\{\int_{t_{\ell-1}}^{s}(\Lambda_0\tilde g_r)_\tau d\tau+\epsilon\sum_{k=1}^m\int_{t_{\ell-1}}^{s}(\Lambda_k\tilde g_r)_\tau dw_k(\tau)\right\}dw_r(s)\\
&=I_{00}(\Lambda_0 f)-\theta h_\ell I_0(\Lambda_0 f)+\epsilon\sum_{r=1}^m\big(I_{0r}(\Lambda_r f)-\theta h_\ell I_r(\Lambda_r f)I_{r0}(\Lambda_0\tilde g_r)\big)+\epsilon^2\sum_{r,k=1}^m I_{rk}(\Lambda_k\tilde g_r),
\end{aligned}$$

and, hence, a representation of the local error in terms of (multiple) stochastic integrals.

Next, we study the leading term $I_{00}(\Lambda_0 f) - \theta h_\ell I_0(\Lambda_0 f)$ of this representation. Since $\Lambda_0 f$ belongs to $C^{2,1}$, we may use the Itô formula (4.4) again and obtain

(4.5)
$$\Lambda_0 f_\tau = \Lambda_0 f_{t_{\ell-1}} + \int_{t_{\ell-1}}^{\tau} (\Lambda_0\Lambda_0 f)_s ds + \epsilon \sum_{r=1}^{m} \int_{t_{\ell-1}}^{\tau} (\Lambda_r \Lambda_0 f)_s dw_r(s)), \quad \tau \in [t_{\ell-1}, t_\ell],$$

which is taken to compute the desired (multiple) stochastic integrals and the whole leading term, respectively, i.e.,

$$I_{00}(\Lambda_0 f) = \frac{1}{2} h_\ell^2 (\Lambda_0 f)_{t_{\ell-1}} + I_{000}(\Lambda_0\Lambda_0 f) + \epsilon \sum_{r=1}^{m} I_{00r}(\Lambda_r\Lambda_0 f),$$

$$I_0(\Lambda_0 f) = h_\ell (\Lambda_0 f)_{t_{\ell-1}} + I_{00}(\Lambda_0\Lambda_0 f) + \epsilon \sum_{r=1}^{m} I_{0r}(\Lambda_r\Lambda_0 f),$$

$$\begin{aligned} I_{00}(\Lambda_0 f) - \theta h_\ell I_0(\Lambda_0 f) = h_\ell^2 \left(\frac{1}{2} - \theta\right) (\Lambda_0 f)_{t_{\ell-1}} + I_{000}(\Lambda_0\Lambda_0 f) - \theta h_\ell I_{00}(\Lambda_0\Lambda_0 f) \\ + \epsilon \sum_{r=1}^{m} \big(I_{00r}(\Lambda_r\Lambda_0 f) - \theta h_\ell I_{0r}(\Lambda_r\Lambda_0 f)\big). \end{aligned}$$

Now, we split $l_\ell = s_\ell + r_\ell$, where s_ℓ is composed of all integral terms with at least one nonzero index. Then we have $\mathbb{E}(s_\ell | \mathcal{F}_{t_{\ell-1}}) = 0$, and

$$\begin{aligned} r_\ell &= h_\ell^2 \left(\frac{1}{2} - \theta\right) (\Lambda_0 f)_{t_{\ell-1}} + I_{000}(\Lambda_0\Lambda_0 f) - \theta h_\ell I_{00}(\Lambda_0\Lambda_0 f), \\ s_\ell &= \epsilon \sum_{r=1}^{m} \big(I_{00r}(\Lambda_r\Lambda_0 f) - \theta h_\ell I_{0r}(\Lambda_r\Lambda_0 f)\big) \\ &\quad + \epsilon \sum_{r=1}^{m} \big(I_{0r}(\Lambda_r f) - \theta h_\ell I_r(\Lambda_r f) + I_{r0}(\Lambda_0 \tilde{g}_r)\big) + \epsilon^2 \sum_{r,k=1}^{m} I_{rk}(\Lambda_k \tilde{g}_r). \end{aligned}$$

Since all the functions $\Lambda_0\Lambda_0 f$, $\Lambda_0 \tilde{g}_r$, $\Lambda_r\Lambda_0 f$, $\Lambda_r f$, and $\Lambda_k \tilde{g}_r$, $k, r = 1, \ldots, m$, appearing as integrands of multiple stochastic integrals, satisfy the assumptions of Lemma 4.1, we may use the lemma repeatedly to obtain the assertion. □

4.2. Suggestions for local error estimates. The previous result enables us to study the local error term η_ℓ for special relations between ϵ and the stepsizes h_ℓ. Unless $\theta = 1/2$ for the trapezoidal rule, the dominating term of $\|r_\ell\|_{L_p}/h_\ell$ is

$$h_\ell \left|\theta - \frac{1}{2}\right| \|(\Lambda_0 f)_{t_{\ell-1}}\|_{L_p} = h_\ell \left|\theta - \frac{1}{2}\right| \|(f_t + f_x f)_{t_{\ell-1}}\|_{L_p} + \epsilon^2 \mathcal{O}(h_\ell),$$

and hence,

(4.6)
$$\eta_\ell = \bar{\eta}_\ell^* + \mathcal{O}(\epsilon^2 h_\ell^{1/2} + \epsilon h_\ell + h_\ell^2), \quad \bar{\eta}_\ell^* := h_\ell \left|\theta - \frac{1}{2}\right| \|(f_t + f_x f)(x(t_{\ell-1}), t_{\ell-1})\|_{L_p}.$$

Substituting the exact solution value $x(t_{\ell-1})$ with its numerical approximation $x_{\ell-1}$ does not lead to perturbations that are larger than $\mathcal{O}(\epsilon^2 h^{1/2} + \epsilon h + h^2)$. This motivates

the choice of $\bar{\gamma} = 1$ and $\chi(z, x, s, t) := |\theta - \frac{1}{2}|(f_t + f_x f)(z, s)$ for the definition of $\bar{\eta}_\ell$ in (3.3). We assume now that $\theta \neq \frac{1}{2}$ and that the relation of the small parameter ϵ and the applied stepsize h_ℓ is such that the global error term of order $\mathcal{O}(h_\ell)$ dominates the error term of order $\mathcal{O}(\epsilon^2 h_\ell^{1/2})$. This is realized if h_ℓ is much larger than $\epsilon^2 h_\ell^{1/2}$, which we express by

$$\epsilon^2 h_\ell^{1/2} \ll h_\ell, \quad \text{i.e.,} \quad \epsilon^4 \ll h_\ell. \tag{4.7}$$

More precisely, we need that

$$\epsilon^2 \left\| \sum_{r,k=1}^{m} I_{rk}(\Lambda_k \tilde{g}_r) \right\|_{L_p} \Big/ h_\ell^{1/2} < h_\ell \left| \theta - \frac{1}{2} \right| \|(f_t + f_x f)_{t_{\ell-1}}\|_{L_p}, \tag{4.8}$$

which follows from (4.7) provided that the values of the functions $\tilde{g}_{rx}\tilde{g}_k$, $r, k = 1, \ldots, m$, and $f_t + f_x f$ are moderate. Then, the local error term η_ℓ is indeed dominated by $\bar{\eta}_\ell^*$ from (4.6). Note that the corresponding choice of $\bar{\eta}_\ell$ leads to an a priori known estimate of the local error, since $\bar{\eta}_\ell$ involves only the knowledge of $x_{\ell-1}$, not of x_ℓ. That way, step rejections do not occur.

One drawback of the choice of (4.6) is its explicit use of the derivatives of the drift function f, which may not be available in practical problems. Therefore, we look for a derivative-free estimate of η_ℓ. We use

$$h_\ell \|(\Lambda_0 f)_{t_{\ell-1}}\|_{L_p} = \|f_{t_\ell} - f_{t_{\ell-1}}\|_{L_p} + \mathcal{O}(h_\ell^2 + \epsilon h_\ell^{3/2} + \epsilon h_\ell^{1/2}),$$

which follows from the expansion

$$f_{t_\ell} - f_{t_{\ell-1}} = h_\ell (\Lambda_0 f)_{t_{\ell-1}} + I_{00}(\Lambda_0 \Lambda_0 f) + \epsilon \sum_{r=1}^{m} I_{r0}(\Lambda_r \Lambda_0 f) + \epsilon \sum_{r=1}^{m} I_r(\Lambda_r f)$$

that is valid under the assumptions of Proposition 4.2 and is obtained by inserting (4.5) into Itô's formula (4.4) for $y = f$. Hence,

$$\eta_\ell = \bar{\bar{\eta}}_\ell^* + \mathcal{O}(h_\ell^2 + \epsilon h_\ell + \epsilon h_\ell^{1/2}), \quad \bar{\bar{\eta}}_\ell^* := \left| \theta - \frac{1}{2} \right| \|f(x(t_\ell), t_\ell) - f(x(t_{\ell-1}), t_{\ell-1})\|_{L_p}. \tag{4.9}$$

The estimate $\bar{\bar{\eta}}_\ell^*$ is dominating the local error term η_ℓ under the more restrictive assumption

$$\epsilon h_\ell^{1/2} \ll h_\ell, \quad \text{i.e.,} \quad \epsilon^2 \ll h_\ell, \tag{4.10}$$

or, more precisely,

$$\epsilon \left\| \sum_{r=1}^{m} I_r(\Lambda_r f) \right\|_{L_p} < h_\ell \|(\Lambda_0 f)_{t_{\ell-1}}\|_{L_p}. \tag{4.11}$$

Then, the term $\bar{\bar{\eta}}_\ell^*$ behaves like an order 1 term. Hence, we represent this choice by $\bar{\gamma} = 1$ and $\chi(z, x, s, t) := |\theta - \frac{1}{2}|\,(f(x, t) - f(z, s))/(t - s)$. In Algorithm 3.1 this is realized by the choice

$$\bar{\gamma} := 1, \quad \bar{\eta}_\ell := \left| \theta - \frac{1}{2} \right| \|f(x_\ell, t_\ell) - f(x_{\ell-1}, t_{\ell-1})\|_{L_p}. \tag{4.12}$$

REMARK 4.3. Note that $\bar{\eta}_\ell^*$ defined by (4.6), as well as $\bar{\bar{\eta}}_\ell^*$ defined by (4.9), vanishes for the trapezoidal rule. Here a more detailed discussion of the remaining terms is necessary. Then the part $\|r_\ell\|_{L_p}/h_\ell$ of the local error is of order 2 and is dominated by $\|(\Lambda_0\Lambda_0 f)_{t_{\ell-1}}\|_{L_p} \cdot h_\ell^2/12$. This term dominates the local error term η_ℓ as long as $\epsilon \ll h_\ell$, which is more restrictive than (4.10). Details will be given in a separate paper.

REMARK 4.4. If the length $T - t_0$ of the considered time interval differs considerably from 1, the more detailed stability estimate

$$\begin{aligned}&\max_{\ell=1,\dots,N} \|x_\ell^* - \tilde{x}_\ell\|_{L_p} \\ &\le \tilde{S}\left(\|x_0^* - \tilde{x}_0\|_{L_p} + \max_{\ell=1,\dots,N}\|s_\ell\|_{L_p}\left(\frac{T-t_0}{h}\right)^{\frac{1}{2}} + \max_{\ell=1,\dots,N}\|r_\ell\|_{L_p}\frac{T-t_0}{h}\right)\end{aligned} \tag{4.13}$$

(cf. [24]) should be used as a starting point. Since the length of the interval affects the local error terms differently, condition (4.8) modifies to

$$\epsilon^2\left\|\sum_{r,k=1}^m I_{rk}(\Lambda_k\tilde{g}_r)\right\|_{L_p}\left(\frac{T-t_0}{h_\ell}\right)^{\frac{1}{2}} < (T-t_0)h_\ell\left|\theta-\frac{1}{2}\right|\|(f_t+f_xf)_{t_{\ell-1}}\|_{L_p}, \tag{4.14}$$

which is true for

$$\epsilon^2((T-t_0)h_\ell)^{\frac{1}{2}} \ll (T-t_0)h_\ell, \quad \text{i.e.,} \quad \epsilon^4 \ll (T-t_0)h_\ell, \tag{4.15}$$

and moderate coefficients $\tilde{g}_{rx}\tilde{g}_k$, $r,k = 1,\dots,m$, and $(f_t + f_x f)$.

REMARK 4.5. The conditions (4.14), (4.15), (4.11), (4.10) are independent of the used time scale.

REMARK. A transformation of the time scale $t \in [t_0, T]$ to $\tau \in [0,1]$ via

$$\begin{aligned}&\tau = (t-t_0)/(T-t_0), \qquad t(\tau) = t_0 + (T-t_0)\tau, \\ &y(\tau) := x(t(\tau)), \qquad \hat{w}(\tau) := (T-t_0)^{-\frac{1}{2}}w(t(\tau))\end{aligned}$$

leads to the transformed SDE

$$y(s)\Big|_0^\tau = (T-t_0)\int_0^\tau f(y(s),t(s))ds + \epsilon(T-t_0)^{\frac{1}{2}}\int_{t_0}^\tau \tilde{G}(y(s),t(s))d\hat{w}(s), \quad \tau\in[0,1].$$

The conditions (4.14), (4.15), (4.11), (4.10) in terms of the transformed functions and variables

$$\hat{f}(y,\tau) = (T-t_0)f(y,t(\tau)), \quad \hat{\tilde{G}}(y,\tau) = (T-t_0)^{\frac{1}{2}}\tilde{G}(y,t(\tau)), \quad \hat{h} = h/(T-t_0), \quad \hat{T} = 1,$$

coincide with the original conditions.

REMARK 4.6. The simple conditions (4.7), (4.15), and (4.10), together with the condition of moderate function values, describe rules of thumb for the used stepsizes. We specify them for $p = 2$, a scalar Wiener process, and the diffusion coefficient $G = (g)$. Neglecting higher order terms in (4.8), (4.14), and (4.11), we then obtain the conditions

$$h_\ell > \frac{\|(g_xg)_{t_{\ell-1}}\|_{L_2}^2}{2|\theta-\frac{1}{2}|\,\|(f_t+f_xf)_{t_{\ell-1}}\|_{L_2}^2}, \tag{4.16}$$

$$(T-t_0)h_\ell > \frac{\|(g_xg)_{t_{\ell-1}}\|_{L_2}^2}{2|\theta-\frac{1}{2}|\,\|(f_t+f_xf)_{t_{\ell-1}}\|_{L_2}^2}, \tag{4.17}$$

and

$$h_\ell > \frac{\|(f_x g)_{t_{\ell-1}}\|_{L_2}^2}{\|(f_t + f_x f + \frac{1}{2} f_{xx} gg)_{t_{\ell-1}}\|_{L_2}^2} \tag{4.18}$$

in terms of the ratio of the coefficients.

For $m = 1$ condition (4.8) simplifies to

$$h_\ell^{-\frac{1}{2}} \|I_{11}(g_x g)\|_{L_p} < h_\ell \left|\theta - \frac{1}{2}\right| \|(f_t + f_x f)_{t_{\ell-1}}\|_{L_p}.$$

Choosing $p = 2$ and neglecting higher order terms, one obtains

$$h_\ell^{-\frac{1}{2}} h_\ell 2^{-\frac{1}{2}} \|(g_x g)_{t_{\ell-1}}\|_{L_2} < h_\ell \left|\theta - \frac{1}{2}\right| \|(f_t + f_x f)_{t_{\ell-1}}\|_{L_2},$$

i.e., (4.16). Analogous arguments apply to (4.14) and (4.11).

5. Test results. We implemented the stepsize strategy proposed in section 3 for the drift-implicit Euler scheme (i.e., (4.2) with $\theta = 1$). The error estimate (3.3) has been realized by the derivative-free choice of (4.12), and the control has been realized by (3.4) for $\ell = 1$ and (3.5) for $\ell \geq 2$. We tested the resulting algorithm extensively for $p = 2$ on a set of SDEs with small noise. First, we report results for two scalar SDEs with known analytic solutions, where we can access the actual errors. The accuracy is measured by the empirical error quantity

$$\max_{\ell=1,\ldots,N} \left(\frac{1}{M} \sum_{j=1}^{M} |x^j(t_\ell) - x_\ell^j|^2 \right)^{1/2} \tag{5.1}$$

that is considered as an estimate of the maximum L_2-norm of the global errors in the considered time interval. Here N denotes the number of steps, M the number of computed paths, and x_ℓ^j the computed approximation of the jth path at time t_ℓ, while $x^j(t_\ell)$ denotes the corresponding exact solution value.

Finally, we present results for a low-dimensional electronic circuit model. We draw some conclusions on the potential and on the limitations of the strategy.

Example 5.1 (linear homogeneous SDE with constant coefficients). We consider the linear scalar SDE in complex arithmetic

$$x(t) = x_0 + \int_0^t \alpha x(s) ds + \int_0^t i\beta x(s) dw(s), \quad t \in [0, 1], \tag{5.2}$$

with coefficients $f(x,t) := \alpha x$, $G(x,t) = (g(x,t)) = (i\beta x)$, initial value x_0, parameters $\alpha, \beta \in \mathbb{R}$, and a scalar Wiener process w. It was implemented as a two-dimensional system in real arithmetic. Its solution is given by the geometric Brownian motion $x(t) = x_0 \exp\big((\alpha + \frac{1}{2}\beta^2)t + i\beta w(t)\big)$. Conditions (4.16) and (4.18) are equivalent to

$$\frac{\beta^4}{2\alpha^4} \ll h_\ell \quad \text{and} \quad \frac{\beta^2}{\alpha^2} \ll h_\ell,$$

respectively. Here, the ratio $|\beta/\alpha|$ plays the role of the small parameter ϵ. As long as stepsizes with $\beta^4/(2\alpha^4) \ll h_\ell$ are used, the Euler scheme shows order 1 of convergence.

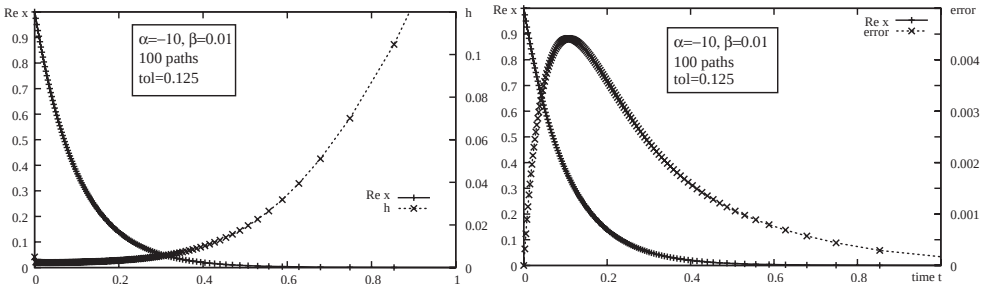


FIG. 5.1. *A computed solution path and stepsizes (left) or global errors (right) for the SDE* (5.2).

As long as even $\beta^2/\alpha^2 \ll h_\ell$ is satisfied, the proposed stepsize control should act properly.

In regions where the first condition is satisfied but the second one is violated, the controlled quantity $\|f_\ell - f_{\ell-1}\|_{L_2}$ is dominated by

$$\left\| \int_{t_{\ell-1}}^{t_\ell} f_x g(x(s), s) dw(s) \right\|_{L_2} \approx |\alpha\beta| h^{\frac{1}{2}} \, \|x_{\ell-1}\|_{L_2}$$

instead of

$$\left\| \int_{t_{\ell-1}}^{t_\ell} f_x f(x(s), s) ds \right\|_{L_2} \approx \alpha^2 h \|x_{\ell-1}\|_{L_2}.$$

The proposed control leads to stepsizes that match

$$\alpha^2 \|x_{\ell-1}\|_{L_2} h \approx \mathrm{tol}^2 / (|\beta| \, \|x_{\ell-1}\|_{L_2}).$$

In the following we present results for different values of the parameters α, β. The initial value was chosen to be $x_0 = 1$. We start with results for the parameters $\alpha = -10$, $\beta = 10^{-2}$ with ratio $|\beta/\alpha| = 10^{-3}$. The solution shows an exponential decrease with the steepest gradients at the beginning of the integration interval. In Figure 5.1 we give the real part of the solution (+) together with the adaptively chosen stepsizes (left picture, ×) and the observed global errors (right picture, ×) for the relative tolerance tol = 0.125 in Algorithm 3.1. The total number of steps was 129 corresponding to an average stepsize of $7.75 \cdot 10^{-3}$, whereas the minimal stepsize was $2.33 \cdot 10^{-3}$.

One hundred paths are computed simultaneously. Figure 5.2 gives the tolerance and accuracy (5.1) versus the number of steps. We plot the relative tolerance (△) and the accuracy with adaptively chosen stepsizes (+) and with constant stepsizes (×) versus the number of steps in logarithmic scale with base 10. Lines with slopes -1 and -0.5 are provided to enable comparisons with convergence of order 1 or 1/2.

To see how much the results are influenced by the statistical error due to only 100 randomly chosen paths, we repeated the computations 100 times for selected values of the relative tolerance. In Table 5.1 the mean value and standard deviation (in percentages) of the observed grid-points of the rejected steps and of the accuracy (5.1) are reported.

Let us add some observations. The global errors do not increase over the whole time interval, but show an exponential decrease after a shorter phase of increase at

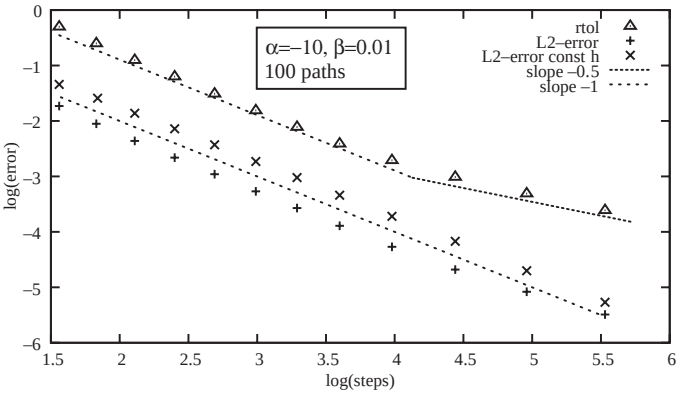


FIG. 5.2. *Tolerance and accuracy versus steps for the SDE* (5.2) *for* $\alpha = -10$ *and* $\beta = 10^{-2}$.

TABLE 5.1
Variation of 100 *repetitions of the simulation results for the SDE* (5.2) *with parameters* $\alpha = -10$, $\beta = 0.01$, *and* 100 *simultaneously computed paths.*

	Successful steps		Rejected steps		Accuracy	
Tolerance	Mean	Deviation	Mean	Deviation	Mean	Deviation
2^{-5}	494	0%	0	0%	$1.08 \cdot 10^{-3}$	0.073%
2^{-10}	27426.25	0.11%	17.96	21.43%	$2.11 \cdot 10^{-5}$	0.077%

the beginning. Therefore, the maximum of the global errors over the integration steps is more meaningful than just the global error at the end of the interval. Furthermore, the error estimates are too pessimistic. Both effects are due to the damping of the solution by $\alpha = -10$, since the errors are damped as well. Further, in the case of the small noise parameter $|\beta/\alpha| = 10^{-3}$, the stepsize control works well up to average stepsizes of 10^{-4} and provides more accurate results than solving the SDE with the same number of constant steps. For higher numbers of steps, the smaller stepsizes at the beginning of the integration interval already lie below the threshold $\beta^2/\alpha^2 \ll h_\ell$. The results in Table 5.1 show that the proposed stepsize control works very well for stepsizes above this threshold and is still quite reasonable for stepsizes slightly below the threshold. The achieved accuracy differs only by less than 0.1% from one repetition of the simulation to another.

Next, we present results for the parameters $\alpha = -0.5$ and $\beta = 0.01$, i.e., $|\beta/\alpha| = 0.02$. Here, the decrease of the solution is much slower, and constant stepsizes are nearly optimal. Compared to the problem with $\alpha = -10$, the noise intensity is larger. In Figures 5.3 and 5.4 and Table 5.2 we present results of experiments that correspond to those described for $\alpha = -10$.

Due to less small noise, the range of accuracy, where the stepsize selection strategy works as intended, decreases to average stepsizes below 10^{-2}. In this range one can see a good correspondence of the tolerance to the accuracy. For very low accuracy (see the far left side of Figure 5.4), constant stepsizes seem to perform even better. Here, one has to see that the adaptive stepsize control first had to find the suitable stepsizes that correspond to the tolerance. Further, Table 5.2 shows that the simulations for $|\beta/\alpha| = 0.02$ and average stepsizes below 10^{-2} are slightly more affected by the randomly chosen 100 paths than the simulations for $|\beta/\alpha| = 0.001$ above. However, the differences from one repetition of the experiment to another still lie below 1%.

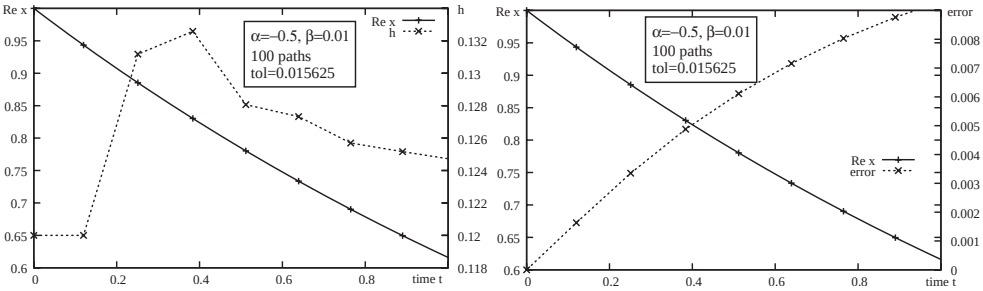


FIG. 5.3. *A computed solution path and stepsizes (left) or global errors (right) for the SDE* (5.2).

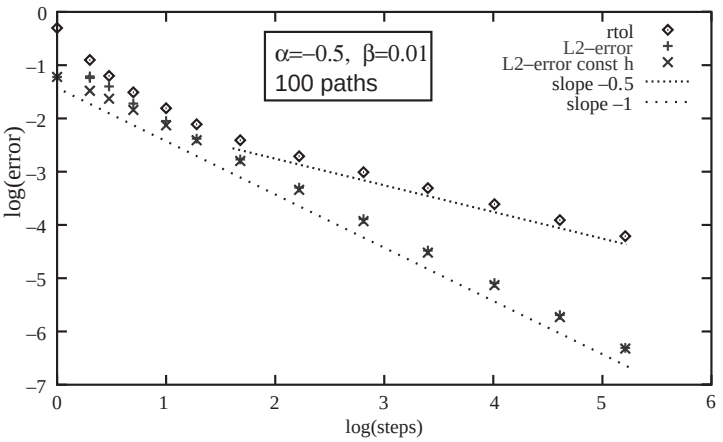


FIG. 5.4. *Tolerance and accuracy versus steps for the SDE* (5.2) *for* $\alpha = -0.5$ *and* $\beta = 10^{-2}$.

TABLE 5.2
Variation of 100 *repetitions of the simulation results for the SDE* (5.2) *with parameters* $\alpha = -0.5$, $\beta = 0.01$, *and* 100 *simultaneously computed paths.*

	Successful steps		Rejected steps		Accuracy	
Tolerance	Mean	Deviation	Mean	Deviation	Mean	Deviation
2^{-6}	8	0%	0	0%	$9.37 \cdot 10^{-3}$	0.018%
2^{-11}	2537.64	0.57%	9.64	34.53%	$3.26 \cdot 10^{-5}$	0.62%

Finally, we report results for $\alpha = -0.5$, $\beta = 0.1$, i.e., $|\beta/\alpha| = 0.2$, where one can hardly speak of small noise. To obtain reasonable approximations of the $\|\cdot\|_{L_2}$-norm of the local error estimates, we increased the number of simultaneously computed paths to M = 1000. Nevertheless, we are in a situation for which the stepsize control proposed in this paper is not tailored. We present corresponding results in Figures 5.5 and 5.6 and in Table 5.3.

Here one can already observe the asymptotic order 1/2 of convergence for medium accuracy. In this region the local error term η_ℓ is dominated by $\|I_{11}(\Lambda_1 g_1)_{\ell-1}\|_{L_2} = \beta^2 h^{1/2} \|x_{\ell-1}\|_{L_2}$, and the stepsize control leads to stepsizes that match

$$\beta^2 h^{1/2} \|x_{\ell-1}\|_{L_2} \approx \mathrm{tol}\, |\beta/\alpha|.$$

Though we have increased the number of simultaneously computed paths to 1000, the results are influenced more by the randomness than in the previous simulations for

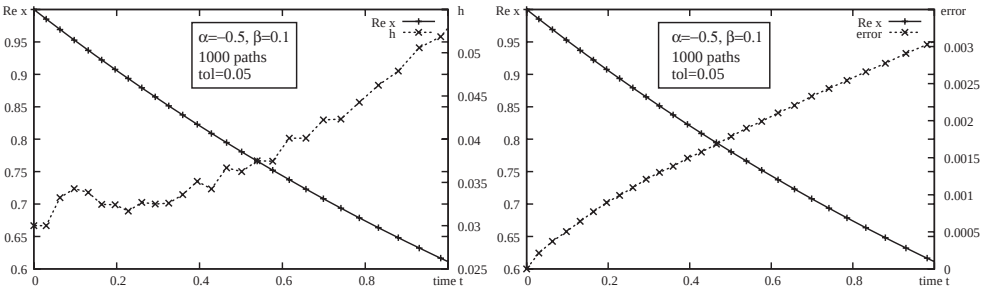


FIG. 5.5. *A computed solution path and stepsizes (left) or global errors (right) for the SDE* (5.2).

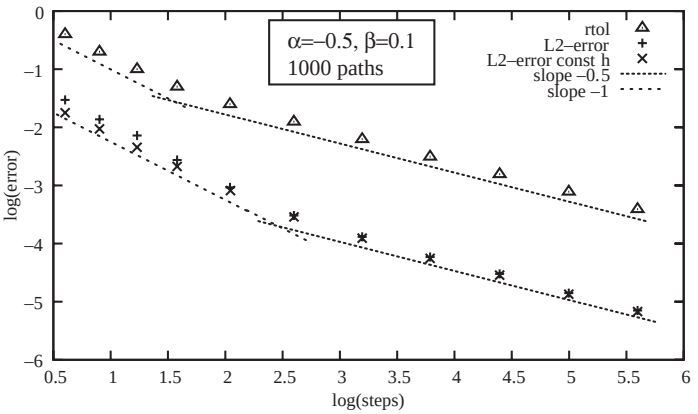


FIG. 5.6. *Tolerance and accuracy versus steps for the SDE* (5.2) *for* $\alpha = -0.5$ *and* $\beta = 0.1$.

TABLE 5.3
Variation of 100 *repetitions of the simulation results for the SDE* (5.2) *with parameters* $\alpha = -0.5$, $\beta = 0.1$, *and* 1000 *simultaneously computed paths.*

	Successful steps		Rejected steps		Accuracy	
Tolerance	Mean	Deviation	Mean	Deviation	Mean	Deviation
$0.1 \cdot 2^{-2}$	102.05	1.17%	0	0%	$9.21 \cdot 10^{-4}$	1.70%
$0.1 \cdot 2^{-6}$	24667.24	1.27%	8.46	60.94%	$2.86 \cdot 10^{-5}$	2.17%

smaller noise. We emphasize that the stepsize selection algorithm relies on a sufficient approximation of the norm of the local error terms by the simultaneously computed paths.

Example 5.2 (SDE with polynomial drift and diffusion). Here we consider a nonlinear scalar SDE with known solution and drift and diffusion coefficients $f(x,t) := -(\alpha + \beta^2 x)(1 - x^2)$, $G(x,t) = (g(x,t)) = (\beta(1 - x^2))$ that are tunable by real parameters α, β:

$$x(t) = \int_0^t -(\alpha + \beta^2 x(s))(1 - x(s)^2)ds + \int_0^t \beta(1 - x(s)^2)dw(s), \quad t \in [0,T], \tag{5.3}$$

where w denotes a scalar Wiener process. The solution is given by (cf. [14, eq. (4.46)])

$$x(t) = \frac{\exp(-2\alpha t + 2\beta w(t)) - 1}{\exp(-2\alpha t + 2\beta w(t)) + 1}.$$

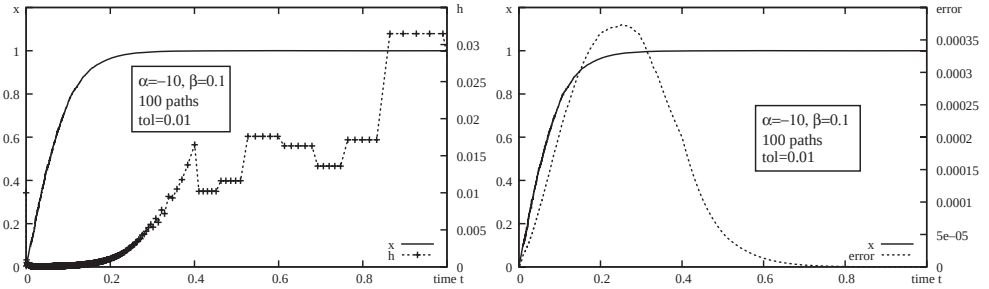


FIG. 5.7. *A computed solution path and stepsizes (left) or global errors (right) for the SDE* (5.3).

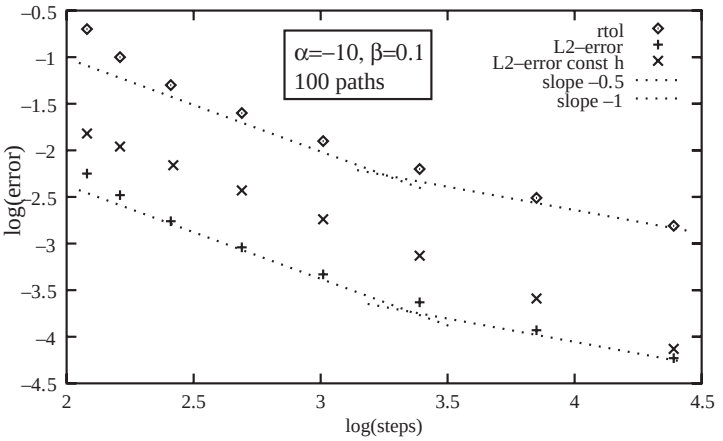


FIG. 5.8. *Tolerance and accuracy versus steps for the SDE* (5.3) *for* $\alpha = -10$, $\beta = 0.1$.

TABLE 5.4
Variation of 100 *repetitions of the simulation results for the SDE* (5.3) *with parameters* $\alpha = -10$, $\beta = 0.1$, *and* 100 *simultaneously computed paths.*

	Successful steps		Rejected steps		Newton failures		Accuracy	
Tolerance	Mean	Deviation	Mean	Deviation	Mean	Deviation	Mean	Deviation
0.01	1316.0	0.20%	5.01	12.46%	17.36	12.94%	$3.75 \cdot 10^{-4}$	0.60 %
0.005	3399.4	0.22%	8.45	23.56%	15.63	12.78%	$1.88 \cdot 10^{-4}$	0.71%

Due to the nonlinearity of the coefficients, the conditions (4.16), (4.17), (4.18) are solution dependent. Another effect of the nonlinearity is that restrictions of the stepsize are necessary to ensure the convergence of Newton's method for solving the nonlinear equations of the drift-implicit Euler scheme in every step. As in the deterministic setting, failures of Newton's method may also cause step rejections, especially for larger stepsizes, where the quality of the starting point for Newton's method may be worse. In such a case we halved the unsuccessful stepsize and forbade stepsize enlargements for the next five steps.

Though the problem is nonlinear, the principal observations we made were the same as for the linear example, Example 5.1. Here, we restrict our presentation to simulation results for one set of parameters, $\alpha = -10$ and $\beta = 0.1$, where 100 paths were computed simultaneously. In Figures 5.7 and 5.8 and Table 5.4 we present results

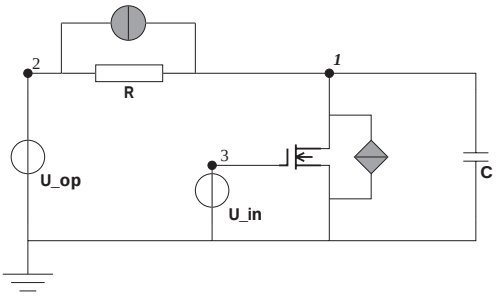


FIG. 5.9. *Thermal noise sources in a MOSFET inverter circuit.*

of experiments that correspond to those described for Example 5.1.

The solution shows a transient behavior at the beginning of the time interval. At the end of the time interval the stepsizes are bounded by the convergence behavior of Newton's method. Failures of Newton's methods also prevented simulation results for larger stepsizes in the case of larger noise. In Table 5.4 additional columns report the number of step rejections due to failures of Newton's method. In Figure 5.8 we observe order 1 behavior up to accuracies of 10^{-3}. The parameter $\alpha = -10$ causes a damping in the solution and a prediction of the global error that is much too pessimistic. The use of adaptive stepsizes provides considerably more accurate results than the computation with the same number of constant steps.

Example 5.3 (electronic circuit). We consider a model of an inverter circuit with a metal oxide semiconductor field effect transistor (MOSFET) under the influence of thermal noise. The equivalent circuit diagram is given in Figure 5.9.

The MOSFET is modeled as a current source from source to drain that is controlled by the nodal potentials at gate, source, and drain: $j_D = f_{mosfet}(e_{gate}, e_{drain}, e_{source})$. In our example the current j_D through the MOSFET is controlled by the input voltage U_{in} and the nodal potential e_1 at node 1: $j_D(U_{in}, e_1) := f_{mosfet}(U_{in}, e_1, 0)$. Often the models of MOSFETs are very sophisticated and involve hundreds of parameters. A first order model for MOSFETs is described in [9], where also further references are given. We simply used

$$j_D = K \cdot \big(\max(U_{GS} - V_{th}, 0)^2 - \max(U_{GD} - V_{th}, 0)^2\big),$$

where $U_{GS} = e_{gate} - e_{source}$, $U_{GD} = e_{gate} - e_{drain}$, and the threshold voltage V_{th} and the current amplification factor K are given parameters. The thermal noise of the resistor and of the MOSFET is modeled by additional white noise current sources that are shunt in parallel to the original, noise-free elements. The noise intensity is given by Nyquist's rule (cf. [2, 5, 26]):

$$I_{th} = \sigma_R \cdot \xi(t) = \sqrt{\tfrac{2k\mathrm{Temp}}{R}} \cdot \xi(t),$$

where $\xi(t)$ is a standard Gaussian white noise process, $k = 1.38066 \cdot 10^{-23}$ $[JK^{-1}]$ is Boltzmann's constant, Temp is the absolute temperature, and R is the resistance. For the thermal noise source of the MOSFET, this formula is modified by considering a solution-dependent conductance $g_D = g_{mosfet}(e_{gate}, e_{drain}, e_{source})$ instead of $g = 1/R$. We used

$$\begin{aligned} g_D &= 0 && \text{if} \quad U_{GS} \le V_{th}, \\ g_D &= \beta \cdot (U_{GS} - V_{th}) \cdot (1 + \lambda U_{DS}) && \text{if} \quad 0 < U_{GS} - V_{th} \le U_{DS}, \end{aligned}$$

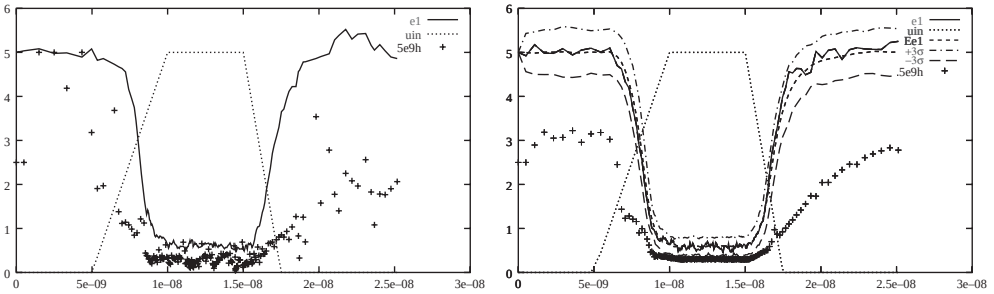


FIG. 5.10. *Simulation results for the noisy inverter circuit. Left:* 1 *path,* 188 (+ 46 *rejected*) *steps. Right:* 100 *paths,* 166 (+ 9 *rejected*) *steps.*

$$g_D = \beta \cdot (U_{DS}) \cdot (1 + \lambda U_{DS}) \qquad \text{if} \quad 0 < U_{GS} - V_{th} > U_{DS},$$

where $U_{DS} = e_{drain} - e_{source}$, and β, λ are parameters. Applying Kirchhoff's current law gives a model for the output voltage e_1 at node 1:

$$Ce_1' - (U_{op} - e_1)/R + j_D(U_{in}, e_1) - \sigma_R \xi_1 + \sigma_D(U_{in}, e_1)\xi_2 = 0, \tag{5.4}$$

where ξ_1, ξ_2 are independent standard Gaussian white noise processes. We treat this system as an Itô SDE with $n = 1$, $m = 2$, $f(x,t) = (U_{op} - x)/(C \cdot R) - j_D(U_{in}, x)/C$, $g_1(x,t) = \sigma_R/C$, $g_2(x,t) = -\sigma_D(U_{in}, x)/C$.

In this simple model, nearly no differences between the solutions of the noisy and the deterministic problem could be seen. Therefore, we dealt with a system where the diffusion coefficients had been scaled by a factor of 1000. In Figure 5.10 we present simulation results for the parameters $C = 2 \cdot 10^{-13}$ $[F]$, $R = 5 \cdot 10^3$ $[\Omega]$, $U_{op} = 5$ $[V]$, Temp $= 300$ $[K]$ on the time interval $[0, 2.5 \cdot 10^{-8}]$ $[s]$. In the MOSFET model we chose $V_{th} = 1$, $K = 10^{-3}$, $\beta = 0.\overline{3} \cdot 10^{-4}$, $\lambda = 0.02$, and the tolerance was 10^{-2}. In solid lines we plotted the values of the nodal potential e_1 and the given input voltage U_{in} versus time. Moreover, the applied stepsizes, suitably scaled, are shown by means of single crosses. We compare simulation results for the computation of a single path (in the left picture of Figure 5.10) with those for the computation of 100 simultaneously computed solution paths (in the right picture of Figure 5.10), where we additionally plotted the mean μ ($E\,e_1$ in the figure) and the lines $\mu \pm 3\sigma$ ($+3\sigma, -3\sigma$ in the figure), where σ was computed as the empirical estimate of the standard deviation for the output voltage. Using the information of an ensemble of simultaneously computed solution paths smoothes the stepsize sequence and considerably reduces the number of rejected steps (from 46 to 9). Even the number of needed successful steps is reduced from 188 to 166.

Acknowledgments. The authors thank Uwe Feldmann and Georg Denk (Infineon Technologies) for invaluable discussions and excellent cooperation and thank two anonymous referees for remarks and suggestions which led to an improved presentation of the material.

REFERENCES

[1] T.A. AVERINA, S.S. ARTIEMIEV, AND H. SCHURZ, *Simulation of Stochastic Auto-oscillating Systems through Variable Stepsize Algorithms with Small Noise*, Preprint 116, Weierstraß-Institut für Angewandte Analysis und Stochastik, Berlin, 1994.

[2] A. Blum, *Elektronisches Rauschen*, Teubner, Stuttgart, 1996.
[3] K. Burrage, P. Burrage, and T. Mitsui, *Numerical solutions of stochastic differential equations—implementation and stability issues*, J. Comput. Appl. Math., 125 (2002), pp. 171–182.
[4] P.M. Burrage and K. Burrage, *A variable stepsize implementation for stochastic differential equations*, SIAM J. Sci. Comput., 24 (2002), pp. 848–864.
[5] A. Demir and A. Sangiovanni-Vincentelli, *Analysis and Simulation of Noise in Nonlinear Electronic Circuits and Systems*, Kluwer, Boston, 1998.
[6] G. Denk and S. Schäffler, *Adams methods for the efficient solution of stochastic differential equations with additive noise*, Computing, 59 (1997), pp. 153–161.
[7] J.G. Gaines and T.J. Lyons, *Variable step size control in the numerical solution of stochastic differential equations*, SIAM J. Appl. Math., 57 (1997), pp. 1455–1484.
[8] I.I. Gichman and A.V. Skorokhod, *Stochastic Differential Equations*, Naukova Dumka, Kiev, 1968 (in Russian).
[9] M. Günther and U. Feldmann, *CAD-based electric-circuit modeling in industry* II. *Impact of circuit configurations and parameters*, Surveys Math. Indust., 8 (1999), pp. 131–157.
[10] K. Gustafsson, M. Lundh, and G. Söderlind, *A PI stepsize control for the numerical integration of ordinary differential equations*, BIT, 28 (1988), pp. 270–287.
[11] E. Hairer, S.P. Nørsett, and G. Wanner, *Solving Ordinary Differential Equations* I, Springer, Berlin, 1987.
[12] N. Hofmann, T. Müller-Gronbach, and K. Ritter, *Optimal approximation of stochastic differential equations by adaptive step-size control*, Math. Comp., 69 (2000), pp. 1017–1034.
[13] N. Hofmann, T. Müller-Gronbach, and K. Ritter, *Step-size control for the uniform approximation of stochastic differential equations with additive noise*, Ann. Appl. Probab., 10 (2000), pp. 616–633.
[14] P.E. Kloeden and E. Platen, *Numerical Solution of Stochastic Differential Equations*, Springer, Berlin, 1992.
[15] H. Lamba, J. Mattingly, and A. Stuart, *An adaptive Euler-Maruyama scheme for SDEs: Convergence and stability*, submitted.
[16] J. Lehn, A. Rössler, and O. Schein, *Adaptive schemes for the numerical solution of SDEs—a comparison*, J. Comput. Appl. Math., 138 (2002), pp. 297–308.
[17] X. Mao, *Stochastic Differential Equations and Their Applications*, Horwood Publishing, Chichester, UK, 1997.
[18] S. Mauthner, *Step size control in the numerical solution of stochastic differential equations*, J. Comput. Appl. Math., 100 (1998), pp. 93–109.
[19] S. Mauthner, *Schrittweitensteuerung bei der numerischen Lösung stochastischer Differentialgleichungen*, Ph.D. thesis, Technische Universität Darmstadt, Darmstadt, Germany, 1999.
[20] G.N. Mil'shtein, *A theorem on the order of convergence of mean-square approximations of solutions of stochastic differential equations*, Theory Probab. Appl., 32 (1988), pp. 738–741.
[21] G.N. Milstein, *Numerical Integration of Stochastic Differential Equations*, Kluwer, Dordrecht, 1995.
[22] G.N. Milstein and M.V. Tret'yakov, *Mean-square numerical methods for stochastic differential equations with small noise*, SIAM J. Sci. Comput., 18 (1997), pp. 1067–1087.
[23] T. Müller-Gronbach, *Strong Approximation of Systems of Stochastic Differential Equations*, Habilitation thesis, Technische Universität Darmstadt, Darmstadt, Germany, 2002.
[24] W. Römisch and R. Winkler, *Stochastic DAEs in circuit simulation*, in Modeling, Simulation and Optimization of Integrated Circuits, K. Antreich, R. Bulirsch, A. Gilg, and P. Rentrop, eds., Birkhäuser, Basel, 2003, pp. 303–318.
[25] G. Söderlind, *The automatic control of numerical integration*, CWI Quart., 11 (1998), pp. 55–74.
[26] L. Weiss and W. Mathis, *A thermodynamical approach to noise in nonlinear networks*, Int. J. Circ. Theor. Appl., 26 (1998), pp. 147–165.
[27] R. Winkler, *Stochastic differential algebraic equations of index* 1 *and applications in circuit simulation*, J. Comput. Appl. Math., 157 (2003), pp. 477–505.
[28] R. Winkler, *Stochastic DAEs in transient noise simulation*, in Scientific Computing in Electrical Engineering, Math. Ind. 4, W. Schilders, E. ter Maten, and S. Houben, eds., Springer, Berlin, 2004, pp. 408–415.

SIAM J. SCI. COMPUT.
Vol. 28, No. 2, pp. 626–650
© 2006 Society for Industrial and Applied Mathematics

ADAPTIVE TIMESTEPPING FOR ELASTOHYDRODYNAMIC LUBRICATION SOLVERS*

C. E. GOODYER† AND M. BERZINS‡

Abstract. The application of variable-step time-integration techniques in a method of lines approach for the numerical solution of transient elastohydrodynamic lubrication calculations is described. Spatial discretization of the coupled integro-differential equations leads to a system of differential-algebraic equations in time. At each timestep a large system of nonlinear equations of 10^4–10^7 variables is solved by using multigrid methods and multilevel multi-integration. The application of temporal error control methods is considered with reference to both the differential-algebraic formulation of the system to be solved and the multigrid-based nonlinear equation solver to be used. The error control method used takes account of the magnitude of the spatial error present by using a local time error control based on the magnitude of this error. This approach is shown to be beneficial in reducing the computational work required. Experimental results on a variety of lubrication problems, including a challenging rough surface problem, are used to display the effectiveness of the methods.

Key words. elastohydrodynamic lubrication, method of lines, differential-algebraic equations, variable timestepping, error balancing

AMS subject classifications. 35Q35, 65M20, 65M15, 65M55

DOI. 10.1137/050622092

1. Introduction.

Elastohydrodynamic lubrication (EHL) problems arise in the mathematical modeling of highly loaded components undergoing deformation when lubricated by fluids such as oil. In particular, EHL may occur in journal bearings and gears where a load is applied on a very small contact area. This load causes pressures of up to 3 GPa and hence elastic deformation of the solid metal contact. Under these extreme conditions lubricating oil may have a glass-like consistency. EHL problems are most often modeled by nonlinear coupled integro-differential equations which require the use of numerical methods. Such EHL problems are of practical industrial interest but as the demand is for fast and robust solvers for increasingly more complicated problems, the ability to reduce the work needed for individual problems is thus paramount. A thorough background of EHL models and computations is provided by [38].

Numerical solutions have been computed since Petrusevich in 1951 [28], and finite difference, and to a lesser extent finite elements (see, e.g., [41]), have been used since then. The biggest advances have come from Lubrecht et al. with the application of multigrid in 1986 [24, 25, 36, 38] and multilevel multi-integration in 1990 [2, 38]. Since then the emphasis has largely been on solving more realistic and hence more physically challenging problems rather than further algorithmic developments. Most of the results are still computed by using a fixed regular grid and a fixed timestep with multilevel methods.

The need to move EHL simulations closer to the design process for both oils and components requires ever more complex problems to be formulated and solved;

*Received by the editors January 6, 2005; accepted for publication (in revised form) December 13, 2005; published electronically May 12, 2006.
http://www.siam.org/journals/sisc/28-2/62209.html
†Computational PDEs Unit, School of Computing, University of Leeds, Leeds, LS2 9JT, UK (ceg@comp.leeds.ac.uk).
‡SCI Institute, University of Utah, Salt Lake City, UT 84112 (mb@sci.utah.edu).

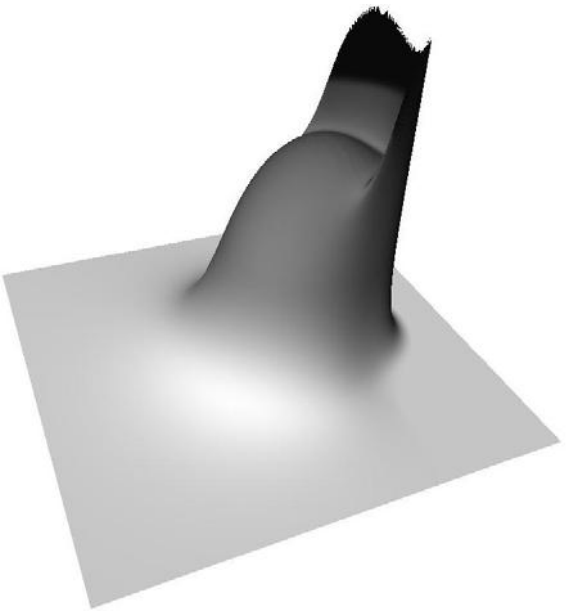


FIG. 1.1. *Typical solution for pressure across an EHL point contact.*

see, e.g., [14]. These problems vary from transient cases, with variable loading and contact speeds, through complicated rheological behavior, to solutions incorporating surface features and, ultimately, true roughness. This requires that the solvers being used must be designed to be extensible to meshes of tens or hundreds of thousands of points in each direction and that the solutions can be obtained sufficiently quickly and reliably for problems with the order of 10^7–10^8 variables. Part of the need for large meshes is that, even in one dimension, mesh spacings as small as 10^{-5} are needed to obtain converged solution values of a pressure spike [17]. The industrial requirement is to be able to make accurate predictions of friction and for this in the case of measured surface roughness large meshes are needed.

EHL problems possess three very different characteristics in different areas of the domain. The cavitation region is the area of the solution beyond the free boundary where the Reynolds equation is not valid. In the center of the domain is the [illegible], where the pressure rises sharply to reach its maximum peak in a near Hertzian shape. EHL pressure profiles differ significantly from those in hydrodynamic cases with no surface deformation in that there is a ridge on the outflow side of the pressure peak. This can be seen in Figure 1.1 where a three-dimensional representation of the pressure profile is shown. Finally, in the noncontact region the pressure is very small compared to the contact region.

The deformation of the surface can be seen in Figure 1.2 which shows the shape of the contact along the centerline. The deformation of the original surface geometry is clearly visible. It can be seen that there is a constriction in the contact toward the outflow, which coincides with a position between the pressure spike and the cavitation boundary. At the outflow boundary a free boundary is formed representing the presence of a cavitation region where bubbles have formed in the lubricant film. The cases considered here always use a thin film approximation and are for the two-dimensional point contact case. In this work we shall restrict our discussion to the time dependent

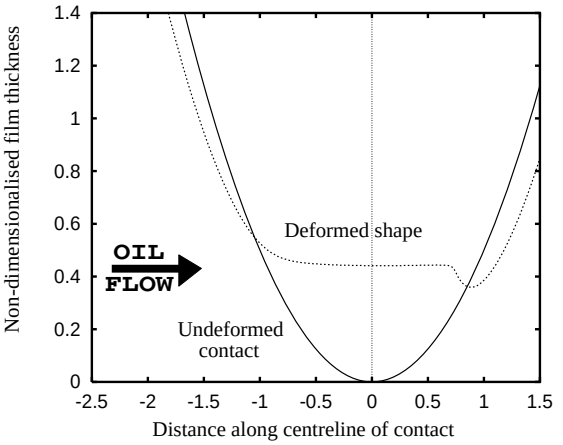


FIG. 1.2. *Typical solution for film thickness along the centerline of an EHL point contact.*

circular point contact cases and consider how to vary the timestep within the context of the multilevel multigrid solver, taking account of the spatial error present. The first results for adaptive timestepping in EHL solutions were presented by Goodyer et al. [13, 16]. More recently Watremetz et al. [39] have presented an alternative method of timestepping based on considering the physical situation rather than any measure of the truncation error.

As larger and larger EHL problems are solved or greater spatial accuracy is used it becomes more important to use a timestepping approach that reflects the spatial error present. This is particularly true for problems with very large numbers of points.

For example, in a benchmark parallel EHL calculation 250 million mesh points have been used by extending the techniques used in [18]. In this paper we thus look at the issue of developing an appropriate timestepping solver and error control method. In resolving this issue it is important to cast the discretized equations as a differential-algebraic system. This allows error estimation procedures from differential-algebraic equation (DAE) theory to be combined with both the multilevel multigrid solver and an estimate of the spatial error to obtain an efficient timesteping scheme. DAE theory also provides insight regarding which variables should be used in a local error test and which should be excluded. The DAE framework also makes it easier to understand how to choose the appropriate classes of time-integration methods and thus makes it easier to understand how to extend the approach described below to use high or variable order methods. In developing the approach described below we are aware of the tremendous advances made in the global error control of ODEs (see, e.g., [5, 9, 23, 26]) and think that such methods offer tremendous potential for the future. At present such methods are still sufficiently expensive that we will explore simpler methods based on attempting to keep the temporal error on a smaller scale than the spatial error [1, 22].

The rest of this paper is laid out as follows. In section 3 the equation system describing the problem is explained and the problem is reformulated as a DAE system. In section 4 a brief explanation of the spatial discretization and time-integration errors is supplied while section 5 describes the nonlinear equations solver employed. Section 6 describes the general approach we are using for the variable timestep approach, based on considering the DAE nature of the system. Section 7 describes in detail the error control mechanism employed in controlling the temporal error given the spatial error

present. Various examples of the use of variable timestepping are shown in section 8 where results for three very different cases, typical of the situations found in real transient EHL contacts, are presented, including consideration of the reduction in solution times.

2. Notation. The following notation of EHL variables will be used throughout the rest of the paper.

$\mathcal{A}$	surface roughness amplitude	a	halfwidth of Hertzian contact
$\mathcal{G}$	undeformed surface geometry	H	nondimensional film thickness
H_{00}	film thickness central offset	K	film thickness kernel matrix
P	nondimensional pressure	p_h	maximum Hertzian pressure
p_0	viscosity parameter	R_x	reduced radius of curvature
T	dimensionless time	u_s	sum of velocities of contacts
$\mathcal{W}$	surface roughness wavelength	X_d	surface roughness pattern center
X, Y	dimensionless coordinates	z	viscosity index
α	pressure viscosity index	ϵ	coefficient in Reynolds eqn.
λ	coefficient in Reynolds eqn.	η_0	viscosity at ambient pressure
$\bar{\eta}$	nondimensional viscosity	Φ	variable load factor
ψ	convergence test parameter	ρ_0	density at ambient pressure
$\bar{\rho}$	nondimensional density		

3. PDE system. The equations governing the EHL point contact are given, in nondimensional form, by the following six equations which are derived in standard texts such as [12, Chapter 4] and [38, Chapters 5 and 6]. The first is the Reynolds equation, which is based on fluid mass conservation and a thin film fluid flow approximation:

$$\frac{\partial}{\partial X}\left(\frac{\overline{\rho}\,H^3}{\overline{\eta}\lambda u_s(0)}\frac{\partial P}{\partial X}\right)+\frac{\partial}{\partial Y}\left(\frac{\overline{\rho}\,H^3}{\overline{\eta}\lambda u_s(0)}\frac{\partial P}{\partial Y}\right)-\frac{u_s(T)}{u_s(0)}\frac{\partial(\overline{\rho}\,H)}{\partial X}-\frac{\partial(\overline{\rho}\,H)}{\partial T}=0, \tag{3.1}$$

where λ is given by

$$\lambda=\frac{6\,\eta_0\,R_x^2}{a^3\,p_h}, \tag{3.2}$$

and where the other symbols are defined in section 2. The two rightmost terms of (3.1) are known as the wedge and squeeze terms, respectively. The boundary conditions are that the pressure is zero at all boundaries and at the outflow of the free boundary there are zero pressure derivatives; see [12, Chapter 11].

Second, the film thickness equation,

$$H(X,Y)=H_{00}+\mathcal{G}(X,Y,T)+\frac{2}{\pi^2}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\frac{P(X',Y')\,\mathrm{d}X'\,\mathrm{d}Y'}{\sqrt{(X-X')^2+(Y-Y')^2}}, \tag{3.3}$$

may be used to compute the contact shape for a known applied pressure distribution. This equation consists of three parts: the offset film thickness or mutual approach, H_{00}, which is the separation of the solids assuming zero deformation; given undeformed geometry $\mathcal{G}(X,Y,T)$, which is generally parabolic, although will also include any roughness features on the surfaces; and the deformation term. The deformation integral is derived by assuming linear, elastic deformation of the two semi-infinite half spaces.

Finally, the force balance equation,

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} P(X,Y)\,\mathrm{d}X\,\mathrm{d}Y = \frac{2\,\pi}{3}\,, \tag{3.4}$$

is the constraint equation through which the applied force is implemented in non-dimensional form, as in (1.36)–(1.38) of [38]. By satisfying this conservation equation, the applied load of the case in question is guaranteed to be the calculated total pressure across the numerical solution.

The role of the force balance equation is quite complex in that without this extra equation there is one more degree of freedom than there are equations. The unknown variable is H_{00} which does not, however, appear directly in (3.4). A standard approach in EHL calculations (see, e.g., [38, equation (6.53)]) is to define an iteration for H_{00} using the residual of the discrete approximation to (3.4). This iteration is defined in (5.1) below, for example. A full explanation of how this iteration may be justified is given below in section 5. A brief explanation is that the discrete pressure integral in a discrete version of (3.4) is identical to an integral involving H_{00} and the film thicknesses. This identity may be seen by using the inverse of the discrete film thickness and pressure relationship defined by the discrete version of (3.3); see section 5 below.

A slightly modified form of this equation will be used in section 8.3 in conjunction with time varying loads.

The lubricant model used is that of a generalized Newtonian fluid. The viscosity model is the Roelands equation [29],

$$\overline{\eta}(P) = \exp\left\{\frac{\alpha\,p_0}{z}\left[-1+\left(1+\frac{Pp_h}{p_0}\right)^{z}\right]\right\}. \tag{3.5}$$

The density is defined using the Dowson and Higginson relation [6]:

$$\overline{\rho}(P) = \left(1+\frac{5.8\times10^{-10}\,Pp_h}{1+1.7\times10^{-9}\,Pp_h}\right). \tag{3.6}$$

It is worth remarking that the theoretical properties of the solution to these equations are not well understood. One notable exception is the finite element analysis of Wu and Oden [41, 42, 43] who considered lightly loaded EHL cases. One theoretical issue, addressed by Stenger, for example [33, 34], is that the film thickness $H(x,y)$ is uniquely defined by the pressure $P(x,y)$ via (3.3).

These equations are also precursors of equations involving rough surface geometries and/or non-Newtonian thermal effects; see, e.g., [10, 17].

3.1. Example reversal of entrainment. A typical example of a time dependent EHL problem is the test case of reversal of entrainment described in [30]. This situation corresponds to flow in one direction changing to the corresponding flow in the opposite direction over a short time period. This change is represented as a linear decrease in velocity. For the example from [30] it corresponds to oil entrainment changing from $0.05\mathrm{ms}^{-1}$ to $-0.05\mathrm{ms}^{-1}$ in 0.2s.

A key feature of this example is the saucer of viscous fluid that forms at the point of reversal (0.1s) and proceeds across the domain (toward the new outflow) before the deformation pattern readopts its characteristic horseshoe shape. All physical parameters are as given in [30], with the standard nondimensionalized quantities as given in Table 3.1. Figure 3.1 shows the central and minimum film thicknesses during this

TABLE 3.1
Nondimensionalized parameters for reversal example.

Parameter	Value
Viscosity index, α	2.1×10^{-8} Pa^{-1}
Maximum Hertzian pressure, p_h	0.45GPa
Initial sum of rolling speeds, $u_s(0)$	0.1ms^{-1}
Nondimensional coefficient, λ	0.056
Material parameter, G	2961
Load parameter, W	6.58×10^{8}
Speed parameter, U	1.47×10^{9}
Moes parameter, M	52.2
Moes parameter, L	6.9

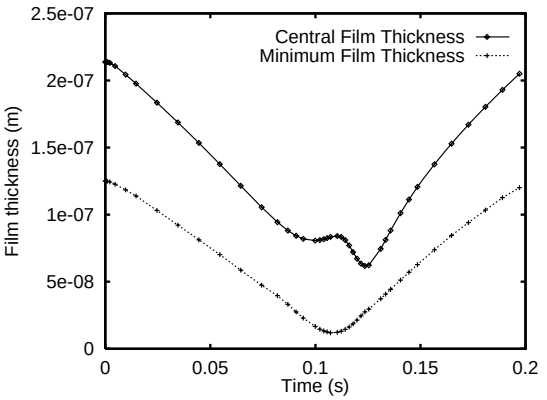


FIG. 3.1. *Central and minimum film thicknesses for reversal of entrainment.*

example. The points on the line indicate where timesteps have been taken. This test case will be used to demonstrate the issues that arise during the development of the timestepping algorithms. Comparisons of numerical solutions against experimental results are given by Goodyer [13].

4. Spatial and temporal discretization. Spatial discretization of the EHL system defined by (3.1)–(3.6) leads to a system of DAEs [3]. Defining the vector of film thicknesses across the whole domain, $\underline{H}$, and its multiple by the density $\underline{\rho}$ in a pointwise manner by

$$\begin{aligned} [\underline{H}]_k &= H_{i,j} \text{ for } k = i + (j-1)\times N_X, \\ \left[\underline{\rho H}\right]_k &= \rho_{i,j} H_{i,j}, \qquad i = 1,\ldots,N_X, \\ &\qquad\qquad\quad j = 1,\ldots,N_Y, \end{aligned} \tag{4.1}$$

with the vector of pressures, $\underline{P}$, likewise, then the first order upwind finite difference in space, implicit Euler in time, discretization of the Reynolds equation (3.1) is given by

$$\begin{aligned} &\frac{(\epsilon^n_{i-1,j}+\epsilon^n_{i,j})(P^n_{i-1,j}-P^n_{i,j}) + (\epsilon^n_{i+1,j}+\epsilon^n_{i,j})(P^n_{i+1,j}-P^n_{i,j})}{2(\Delta X)^2} \\ &+ \frac{(\epsilon^n_{i,j-1}+\epsilon^n_{i,j})(P^n_{i,j-1}-P^n_{i,j}) + (\epsilon^n_{i,j+1}+\epsilon^n_{i,j})(P^n_{i,j+1}-P^n_{i,j})}{2(\Delta Y)^2} \end{aligned}$$

$$-\frac{u_s(T)}{u_{ref}}\frac{\rho^n_{i,j}H^n_{i,j}-\rho^n_{i-1,j}H^n_{i-1,j}}{\Delta X}-\frac{\rho^n_{i,j}H^n_{i,j}-\rho^{n-1}_{i,j}H^{n-1}_{i,j}}{\Delta T}=0, \tag{4.2}$$

where the superscript n denotes values at time t_n. The boundary conditions are that all exterior boundaries have $P = 0$ and the line $j = 1$ is a symmetry condition in the Y direction. It is important to remark that once the entrainment direction has been reversed, the upwind direction of the discretization of the wedge term $\frac{\partial(\overline{\rho}H)}{\partial X}$ needs to be changed to reflect the changing flow direction.

The discretized forms of (3.3)–(3.6) are

$$H_{i,j} = H_{00} + \mathcal{G}_{i,j} + \sum_{k=1}^{N_X}\sum_{l=1}^{N_Y} K_{i,j,k,l}P_{k,l}, \tag{4.3}$$

$$\frac{2\pi}{3} = \Delta X \Delta Y \sum_{i=1}^{N_X}\sum_{j=1}^{N_Y} P_{i,j}, \tag{4.4}$$

$$\overline{\rho}_{i,j} = \frac{0.59\times 10^9 + 1.34 p_h P_{i,j}}{0.59\times 10^9 + p_h P_{i,j}}, \tag{4.5}$$

$$\text{and} \quad \overline{\eta}_{i,j} = \mathrm{e}^{\left\{\frac{\alpha p_0}{z}\left[-1+\left(1+\frac{p_h P_{i,j}}{p_0}\right)^z\right]\right\}}, \tag{4.6}$$

with all symbols as defined previously.

It is important to stress that the dependence of the film thickness $H_{i,j}$ at any point on the pressure values at every mesh point $P_{k,l}$ means that the sparsity pattern of the discrete nonlinear equations is dense in every case. When this dependence is combined with the nonlinear nature of the viscosity and density the result is a computationally challenging problem.

In (4.2) a first order upwind discretization of both first order derivatives is shown. Alternative schemes have been used by Venner and Lubrecht [38] and Wijnant [40]. Both methods again require the leading coefficients of both the wedge and squeeze terms to be one. Optimal use of these second order schemes rely on having a fixed relationship between the spatial and temporal timestep sizes, which, in an adaptive time-integration code, is no longer the case. These second order schemes are, in our experience, somewhat less stable when applied across a wide variety of cases.

4.1. DAE formulation. In order to apply DAE algorithms, theory, and error indicators it is helpful to recast the discretized EHL equations as DAEs. This is important as the form of the DAEs makes a significant difference to the algorithms that can be employed both for solution and for error estimation.

The differential-algebraic form of the discretized EHL equations may be written as

$$\underline{F}_1(\underline{P}, \underline{\overline{\rho}H}) - \left[\overline{\rho}H\right]' = 0, \tag{4.7}$$

the film thickness equation, (3.3), by

$$\underline{F}_2(\underline{P}, \underline{H}) = 0, \tag{4.8}$$

and the density equation, (3.6), by

$$\underline{F}_3(\underline{P}, \underline{\overline{\rho}}) = 0, \tag{4.9}$$

where $'$ denotes differentiation in time. These can then be combined to define a DAE system for $\underline{U}^{\mathrm{T}} = (\underline{P}^{\mathrm{T}}, \underline{H}^{\mathrm{T}})$ by

$$\underline{F}(\underline{U}, \underline{U}', t) = 0. \tag{4.10}$$

In the above case the DAE aspect arises from the lack of a temporal derivative of pressure in (4.7)–(4.9). Issues affecting the differences between solving ODE and DAE systems are discussed in detail in [3]. The authors have computational evidence that the EHL system is index one for regular meshes but need to show that $\frac{\partial \underline{F}_2}{\partial \underline{P}}$ is nonsingular in general.

The index of a DAE is defined as the minimum number of times that all or part of (4.10) must be differentiated with respect to T in order to determine U' as a continuous function of U and T [3]. Thus to find the index of the EHL DAE system given by (4.10), it is necessary to first differentiate (4.8) with respect to T to get

$$\frac{\partial \underline{F}_2}{\partial \underline{P}} \frac{\mathrm{d}\underline{P}}{\mathrm{d}T} + \frac{\partial \underline{F}_2}{\partial \underline{H}} \frac{\mathrm{d}\underline{H}}{\mathrm{d}T} = 0. \tag{4.11}$$

Hence, if $(\frac{\partial \underline{F}_2}{\partial \underline{H}})^{-1}$ exists, this gives an expression for $\frac{\mathrm{d}H}{\mathrm{d}T}$. Similarly, differentiating (4.9) gives

$$\frac{\partial \underline{F}_3}{\partial \underline{P}} \frac{\mathrm{d}\underline{P}}{\mathrm{d}T} + \frac{\partial \underline{F}_3}{\partial \underline{\overline{\rho}}} \frac{\mathrm{d}\underline{\overline{\rho}}}{\mathrm{d}T} = 0, \tag{4.12}$$

which gives an explicit expression for $\frac{\mathrm{d}\overline{\rho}}{\mathrm{d}T}$ if $\frac{\partial \underline{F}_3}{\partial \underline{\overline{\rho}}}$ is nonsingular.

Note that (4.7) may also be written as

$$\frac{\mathrm{d}\underline{\overline{\rho} H}}{\mathrm{d}T} = \underline{H} \frac{\mathrm{d}\underline{\overline{\rho}}}{\mathrm{d}T} + \underline{\overline{\rho}} \frac{\mathrm{d}\underline{H}}{\mathrm{d}T} = \underline{F}_1(\underline{P}, \underline{\overline{\rho} H}); \tag{4.13}$$

then expressions for P', H', and $\overline{\rho}'$ have been obtained in one differentiation step. Hence, the index of the transient EHL problem is one, provided both the aforementioned inverses exist.

The calculation of the density, $\overline{\rho}$, is entirely local, leading to a diagonal Jacobian matrix, $\frac{\partial \underline{F}_3}{\partial \underline{P}} \frac{\mathrm{d}\underline{P}}{\mathrm{d}T}$. This matrix is nonsingular provided that none of the diagonal entries given by

$$\frac{\partial \overline{\rho}_{i,j}}{\partial P_{i,j}} = \frac{\partial}{\partial P_{i,j}} \left(\frac{0.59 \times 10^9 + 1.34 p_h P_{i,j}}{0.59 \times 10^9 + p_h P_{i,j}} \right) \tag{4.14}$$

$$= \frac{0.2006 \times 10^9 p_h}{(0.59 \times 10^9 + p_h P_{i,j})^2} \tag{4.15}$$

are nonzero. For physical, nonnegative pressures, P, this matrix is invertible.

The invertibility of $\frac{\partial F_2}{\partial P}$ is much harder to establish. This is because the kernel matrix K is dense, and hence simple analysis is not possible. A computational investigation into the properties of this matrix showed that it was numerically nonsingular. Unfortunately, due to the very large size of the matrices involved, it was only possible to numerically investigate cases up to grid size 33×33, which remains far short of the fine mesh cases, with more than 1000×1000 points.

5. Nonlinear equations solution method. The methods used for solving for the pressure, the film thickness, and the lubricant properties are now described in turn. In order to obtain solutions to EHL problems quickly it has proved necessary [36, 38] to use multilevel and multigrid methods. Standard texts such as Briggs [4] and Trottenberg, Oosterlee, and Schüller [35] provide descriptions of multigrid methods.

In solving the Reynolds equation it is important to make use of different numerical schemes to reflect the different physical situation in each of the three different regions [13]. In the noncontact region a Gauss–Seidel line relaxation scheme is used; in the contact region a Jacobi line scheme is employed (see Nurgat, Berzins, and Scales [27]) rather than the distributed scheme of Venner [36]. In the cavitation region the Christopherson approach is used [7] to set all calculated negative pressures to zero.

The film thickness calculation, (3.3) once discretized, has the form given in (4.3). Hence for every mesh point, (i, j), the deformation term is a multi-summation of the pressures at all the other points in the computational domain. This calculation is therefore extremely expensive to compute exactly as it is $\mathcal{O}(N_X^2 N_Y^2)$ operations. We therefore use the multilevel multi-integration technique of Brandt and Lubrecht [2], which is described fully by Venner and Lubrecht [38], to reduce this calculation to $\mathcal{O}(N_X N_Y \ln N_X \ln N_Y)$ operations.

The calculation of H_{00} in (4.3) is accomplished by relaxation of the force balance equation (3.4), according to

$$H_{00} \leftarrow H_{00} - c\left(\frac{2\pi}{3} - \Delta X \Delta Y \sum_{i=1}^{N_X}\sum_{j=1}^{N_Y} P_{i,j}\right), \tag{5.1}$$

for all mesh points (i, j), where c is a small relaxation parameter; see [38, p. 148] and [12, p. 365].

At first sight it is unclear exactly how this equation may be justified. Write (4.3) and (4.4) in vector form as

$$\underline{H} = H_{00}\ \underline{e} + \underline{G} + K\underline{P}, \tag{5.2}$$

$$\frac{2\pi}{3} = \Delta X \Delta Y \underline{e}^T \underline{P}, \tag{5.3}$$

where $\underline{e}$ is the vector whose components are all one, $\underline{G}$ is the vector whose components are $\mathcal{G}_{i,j}$, and K is the matrix whose entries are composed of the elements $K_{i,j,k,l}$. Define the vector $\underline{y}$ as the solution of the system of equations

$$K\underline{y} = \underline{e}. \tag{5.4}$$

Hence

$$\underline{y}^T = \underline{e}^T\ K^{-1}. \tag{5.5}$$

It is worth remarking that although we do not know if K has an inverse we can easily attempt to compute $\underline{y}$ numerically. It then follows that the constraint equation (5.3) may be written by using (5.2) and (5.5) as

$$\frac{2\pi}{3} = \Delta X \Delta Y\ \underline{y}^T(\underline{H} - H_{00}\ \underline{e} - \underline{G}). \tag{5.6}$$

Although it would be possible to store and save $\underline{y}^T$ and $\underline{e}$, it is easier to use an iterative method. Applying a standard Newton–Jacobi-type procedure to calculate H_{00} from this equation with relaxation factor λ then gives

$$H_{00} \leftarrow H_{00} - c\left(\frac{2\pi}{3} - \Delta X \Delta Y \underline{y}^T(\underline{H} - H_{00}\underline{e} - \underline{G})\right), \tag{5.7}$$

where

$$c = \frac{\lambda}{\Delta X \Delta Y \underline{y}^T \underline{e}}. \tag{5.8}$$

Equations (5.1) and (5.7) are easily seen to be equivalent using (5.2). Computational experience suggests that c must be small and be chosen with some care.

One smoothing cycle is said to be the sequence of updating all the pressures, $P_{i,j}$, and film thickness values, $H_{i,j}$, along with the corresponding density and viscosity values.

In any implicit time-integration process it is important to ensure that the starting guess on a timestep is a good prediction. We therefore use linear extrapolation of the pressure, film thickness, and H_{00} solutions from the previous two steps. This prediction typically brings the root mean square residual at the start of the timestep down by an order of magnitude and possibly suggests that higher order methods might make an impact in time integration. Although it is possible to calculate $\underline{H}^{(0)}(t_n)$ directly from the predicted pressure solution, $\underline{P}^{(0)}(t_n)$, this does not lead to any noticeable improvements.

The solution method employed, described in [27] and [13], iteratively solves for pressure before updating the film thickness. One addition for the time dependent case is that the method for testing for convergence of the nonlinear iteration is the well-known (see, e.g., [3]) convergence test of [31]. In this test the iteration cycle on a timestep is continued until

$$\frac{\psi}{1-\psi}\left\|\underline{H}^{(m+1)}(t_n) - \underline{H}^{(m)}(t_n)\right\| < 0.33tol, \tag{5.9}$$

where tol is the time-integration error tolerance, and m is the iteration number. This also requires the definition of an estimate for the rate of convergence, ψ,

$$\psi = \left(\frac{\left\|\underline{H}^{(m+1)}(t_n) - \underline{H}^{(m)}(t_n)\right\|}{\left\|\underline{H}^{(1)}(t_n) - \underline{H}^{(0)}(t_n)\right\|}\right)^{\frac{1}{m}}, \tag{5.10}$$

with a suitable norm, $\|\cdot\|$, which we define below. An additional source of difficulty for EHL solvers is that it is often necessary to solve a steady state problem to calculate the initial conditions.

6. Local error estimation. For transient EHL calculations the choice of correct timestep size is critical. If the timestep is too large then important physical features may be missed should they fall between successive steps. Equally, choosing a very small stepsize may, at best, lead to a large amount of computational work for very small changes in the solution and, at worst, result in solutions diverging, for example, due to the magnification of temporal gradients. This is due not to the stability of the problem but to the convergence properties of the nonlinear solver described in detail in [13]. The strategy used here comes from [13] but a similar approach has been

used by Gaskell et al. [11] more recently. There are also issues unique to EHL-based multigrid solvers. For example, should the timestep become very small, then any corrections made may amplify, rather than reduce, the errors in the solution unless very small underrelaxation parameters are used in the multigrid solver.

6.1. Choice of variables. One of the central issues in the numerical solution of DAEs is the proper estimation of the error and which variables to include or exclude in the error test [3]. In this case there is a choice as to whether it is the errors in $\underline{P}$ or in $\underline{H}$ which are controlled. Intuitively, because the system given by (4.7) is solved for $\underline{P}$ and (4.8) is solved for $\underline{H}$ using this $\underline{P}$, it seems sensible to control the errors in $\underline{P}$. However, it is the contact region where the most change is taking place, and this is dominated by the wedge and squeeze terms in (3.1) which involve derivatives of $\underline{H}$. Experiments have confirmed that controlling these errors requires significantly less work per timestep and fewer timesteps are required. The error tests will therefore be formulated for variable $\underline{H}$. The difference between the magnitudes of the local errors in H and P is shown in the next subsection.

6.2. Calculating the local error estimate. Once the convergence test of (5.9) has been satisfied, a local error calculation is undertaken to establish a new timestep size. The size and nonlinearity of the systems of nonlinear equations to be solved at each timestep preclude the use of LU linear equation solvers to estimate the DAE local error by means of an extra backsolve [3, p. 147]. Instead, a multigrid-based iterative approach will be used to estimate the local error over the step. In estimating the local error we rely on the types of methods used by Brenan et al. [3] in our software, while recognizing that there are many alternatives.

First let us consider the Reynolds equation (4.7). Following the derivation given in [3, section 5.4], the system given by (4.7) can be considered, for the implicit Euler scheme used here, as

$$\frac{\overline{\rho_M}(\underline{P}_{n+1})\underline{H}_{n+1} - \overline{\rho_M}\,(\underline{P}_n)\,\underline{H}_n}{\Delta T} - \underline{F}_1(\underline{P}_{n+1}, \underline{H}_{n+1}) = 0, \tag{6.1}$$

where $\overline{\rho_M}\,(\underline{P})$ is a diagonal matrix with the ith diagonal entry consisting of the function $\overline{\rho}(.)$ evaluated using the ith pressure value. The true solution at timestep $n+1$ satisfies

$$\frac{\overline{\rho_M}\,(\underline{P}(t_{n+1}))\,\underline{H}\,(t_{n+1}) - \overline{\rho_M}\,(\underline{P}(t_n))\,\underline{H}(t_n) - \underline{TE}}{\Delta T} - \underline{F}_1(\underline{P}(t_{n+1}), \underline{H}(t_{n+1})) = 0, \tag{6.2}$$

where $\underline{TE}$ is the local truncation error, defined by

$$\underline{TE} = [\overline{\rho_M}\,(\underline{P})\,\underline{H}]''\,\frac{(\Delta T)^2}{2}. \tag{6.3}$$

In estimating the truncation error, $\underline{TE}$, our experiments have shown that the variation of the density is not as important as that of the film thickness. One may therefore consider treating the truncation error simply as $\overline{\rho_M}\underline{H}''$ and estimate it using the usual difference between predicted and corrected values; see [3]. Thus the estimated value denoted by $\underline{TE_a}$ is

$$\underline{TE_a} = \frac{\overline{\rho_M}(\underline{P_{n+1}})}{2}(\underline{H}_{n+1} - \underline{H}^{(0)}_{n+1}). \tag{6.4}$$

In estimating the growth in local error from t_n to t_{n+1} we assume that the error at t_n is zero and consequently that $\underline{P}_n = \underline{P}(t_n)$ and that $\underline{H}_n = \underline{H}(t_n)$. The local errors in pressure and film thickness are defined by $\underline{le\,H}$ and $\underline{le\,P}$ as

$$\begin{aligned} \underline{le\,P} &= \underline{P}_{n+1} - \tilde{\underline{P}}_{n+1}, \\ \underline{le\,H} &= \underline{H}_{n+1} - \tilde{\underline{H}}_{n+1}, \end{aligned} \tag{6.5}$$

where $\tilde{\underline{P}}_{n+1}$ and $\tilde{\underline{H}}_{n+1}$ are the values of the local solution to the equations with the computed values at t_n as the initial values.

This therefore leads to the following equation for the local errors $\underline{le\,H}$ and $\underline{le\,P}$:

$$\begin{aligned} \frac{\overline{\rho_M}(\tilde{\underline{P}}(t_{n+1}))\tilde{\underline{H}}\,(t_{n+1}) - \overline{\rho_M}(\tilde{\underline{P}}(t_n))\tilde{\underline{H}}(t_n)}{\Delta T} - \underline{F}_1(\tilde{\underline{P}}(t_{n+1}), \tilde{\underline{H}}(t_{n+1})) &= \underline{TE_a}, \\ \underline{F}_2(\tilde{\underline{P}}(t_{n+1}), \tilde{\underline{H}}(t_{n+1})) &= 0. \end{aligned} \tag{6.6}$$

This equation may then be solved for $\tilde{\underline{P}}$ and $\tilde{\underline{H}}$ using the standard EHL multigrid algorithm with right-hand side $\frac{\overline{\rho}}{2\Delta T}(\underline{H}_{n+1} - \underline{H}^{(0)}_{n+1})$.

In the case of the film thickness equation (4.3) since there is no transient term present, it can easily shown that neither the geometry nor H_{00} is present in the local error calculation, hence giving

$$\underline{le\,H} - K\underline{le\,P} = 0. \tag{6.7}$$

This gives us a relationship between the local errors in $\underline{H}$ and $\underline{P}$, where K is the film thickness integration kernel matrix.

The cost of this local error calculation on a timestep is that two or three more multigrid V-cycles are carried out to obtain solutions, $\tilde{\underline{P}}$ and $\tilde{\underline{H}}$, to the local error problem. This number of additional cycles is about one fifth of the total number of multigrid cycles.

Once these new solutions are calculated, an estimate of the total local error in $\underline{H}$ may be defined as

$$\|\underline{le\,H}\|_\omega = \|\tilde{\underline{H}}_{n+1} - \underline{H}_{n+1}\|_\omega. \tag{6.8}$$

The chosen norm, $\|\cdot\|_\omega$, is a weighted root mean square L_2-norm, as used in the software package DASSL, described in [3], defined by

$$\|\underline{H}\|_\omega = \sqrt{\frac{1}{N_X N_Y}\sum_{i=1}^{N_X}\sum_{j=1}^{N_y}\left(\frac{H_{i,j}}{\omega_{i,j}}\right)^2}, \tag{6.9}$$

with weights $\omega_{i,j}$ defined by

$$\omega_{i,j} = ATOL + H^{(0)}_{i,j} RTOL. \tag{6.10}$$

These weights are themselves, therefore, given in terms of the predicted solution at that mesh point on that timestep, $H^{(0)}_{i,j}$, and the absolute and relative error tolerances, $ATOL$ and $RTOL$, which will be defined in the next section.

Returning to the relationship between the local errors in P and H given in (6.7) and taking norms give

$$\|\underline{le\,H}\|_\omega \le \|K\|\,\|\underline{le\,P}\|_\omega \tag{6.11}$$

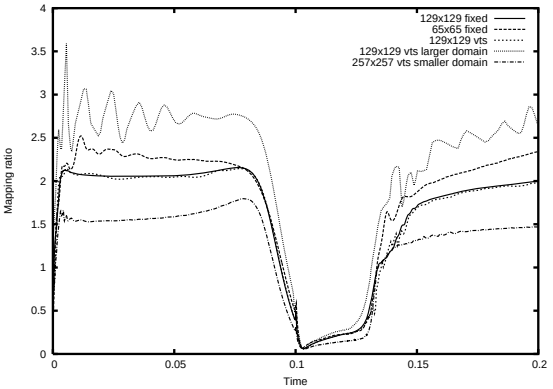


FIG. 6.1. *Ratio of* $\|\underline{le\, H}\|_\omega / \|\underline{le\, P}\|_\omega$.

for an appropriate matrix norm $\|.\|$. Examples of estimates of the values of this norm are shown below for the problem described in section 3.1. Figure 6.1 shows that the ratio of these errors satisfies

$$\|\underline{le\, H}\|_\omega / \|\underline{le\, P}\|_\omega \leq 4 \tag{6.12}$$

for calculations such as those in section 8. It is straightforward to monitor the ratio defined by the left-hand side of (6.12) during the calculation.

The local error test employed is then

$$\|\tilde{\underline{H}}_{n+1} - \underline{H}_{n+1}\|_\omega \leq 1. \tag{6.13}$$

Once this is true the stepsize change ratio is calculated by

$$r = (2\|\underline{le\, H}\|_\omega)^{\frac{-1}{k+1}}\,, \tag{6.14}$$

with k being the order of the method [32]. The new stepsize should then be chosen to be

$$\Delta T_{n+1} = r\Delta T_n, \tag{6.15}$$

subject to some standard limitations concerning maximum permissible stepsize. For example, if the error is small, e.g., $r > 1.5$ in (6.14), then the stepsize may be increased for the next timestep. If $r < 0.9$, then either the stepsize is reduced for the following step or, if the current step is considered to have failed, the current timestep may be retaken with a new stepsize. In between these extremes the stepsize is left unchanged. Stepsize increases are allowed only every two steps, but decreases are allowed on every step as is common in a number of time-integration codes. The main issue now is how to choose $RTOL$ and $ATOL$ so that they reflect the underlying spatial discretization errors.

7. Space and time error estimation and balancing. The general issue of controlling the global error in time dependent PDE calculations remains problematic despite many recent successful algorithms based upon adjoint methods; see [5, 26]. Adjoint methods are also important for EHL problems in that they provide a means of controlling the error in quantities of interest such as friction [20]. The key issue

for large scale engineering calculations is the cost of the estimation algorithms. The counterargument is that without incurring this cost there is no sure way of understanding the accuracy of the solution. Nevertheless for problems for which even computing any solution is a formidable challenge and an expensive use of computer time the option of using the adjoint-based error control algorithms cost is unattractive. For example, the algorithm of [5] requires one preliminary integration with a coarse tolerance, storage of solution values and then two adjoint integration techniques before the main integration is started.

In this section we consider less expensive options which take into account the magnitude of the spatial error based on the approach of [1, 22] but modified still further to reduce computational costs.

Let the continuous equation system, defined by (3.1)–(3.6), have an exact solution $u(x, y, t)$ together with a vector of values of this solution at spatial grid points at time t $\underline{u}(t)$, and the discretized equation system, defined by (4.10), have exact solution $\underline{U}(t)$. If, at time t, the numerical approximation to the solution of the system is $\tilde{\underline{U}}(t)$, then the total error, $E(t)$, is defined by

$$\begin{aligned} \underline{E}(t) &= \underline{u}(t) - \tilde{\underline{U}}(t) \\ &= (\underline{u}(t) - \underline{U}(t)) + (\underline{U}(t) - \tilde{\underline{U}}(t)) \\ &= \underline{e}(t) + \underline{g}(t), \end{aligned} \tag{7.1}$$

where $\underline{e}(t) = \underline{u} - \underline{U}$ represents the vector of spatial discretization errors on the mesh points, and $\underline{g}(t) = \underline{U} - \tilde{\underline{U}}$ is the global error in the time integration. Given that a solution has been discretized in space to a particular degree of accuracy, $\underline{e}(t)$, it is not worth solving the transient part to a much higher degree of accuracy, but equally this transient error $\underline{g}(t)$ must not degrade the spatial accuracy.

The control of the global error using adjoint methods is an active area; see [5, 9, 23, 26]. The optimal choice of timestep is governed by successfully relating the spatial error of the solution, with the time error. It has been shown (see, e.g., [1, 22]) that controlling the local (temporal) error per step, so that the local growth in time of the spatial error dominates, provides a practical reliable algorithm for balancing these errors. The disadvantage is that the strong theoretical basis of the adjoint methods is lacking. This form of error control, in all the experiments we have conducted, has done a good job of balancing the spatial and temporal errors. However, in common with the results of [1], it is the case that it is relatively expensive in terms of the number of timesteps taken.

In order to reduce computation times a modified approach is thus taken so that instead of controlling the local error in time so that it is a fraction of the growth in the spatial error we control the local error so that it is a fraction of the spatial discretization error. The approach used is like the extrapolation approach of section 3.1 of [22] in that the fine mesh solution is used to estimate the error in a coarser mesh and then extrapolation is used to estimate the error on the fine mesh. Using the approach in these papers yields a simple extrapolation-based estimate of the first order spatial error in the fine mesh solution given by

$$e_{i,j}(t) \approx \tilde{H}_{i,j}(t) - \tilde{H}_{I,J}(t), \tag{7.2}$$

where (i,j) is the fine grid point coincident with the coarse grid point (I,J).

The error control algorithm of [22, equation (3.10)] is then given by defining

$$\|\underline{le\, H}\|_\omega < \varepsilon \left\|\underline{e}_{n+1} - \underline{e}_n\right\|_\omega, \tag{7.3}$$

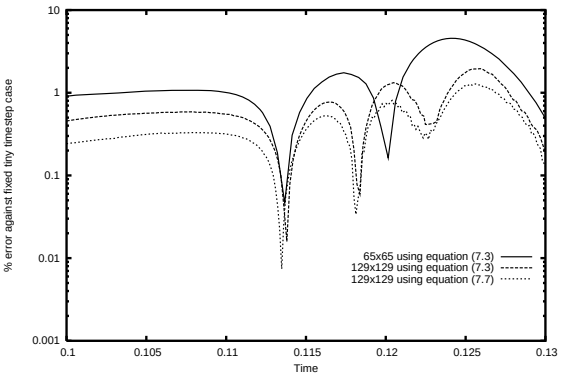


FIG. 7.1. *Comparison of errors in the minimum film thickness through reversal using different error control strategies.*

where $\|\cdot\|_\omega$ is a weighted root mean square L_2-norm, as used in (6.9). Using the approximation

$$\|\underline{e}_{n+1} - \underline{e}_n\|_\omega \approx \Delta t \left\| \frac{\partial \underline{e}}{\partial t} \right\|_\omega \tag{7.4}$$

allows the error control approximation to be written as a local error per unit step. Assuming that the spatial error may be expanded in powers of Δx and that $\Delta x{=}\Delta y$,

$$\left\| \frac{\partial \underline{e}}{\partial t} \right\|_\omega \approx \Delta x^p \left\| \frac{\partial \underline{\tilde{E}}}{\partial t} \right\|_\omega , \tag{7.5}$$

where $p = 1$ is the power of Δx in the spatial error; thus the error control can be approximated, as in [1, 22], by

$$\|\underline{le\, H}\|_\omega < \varepsilon_0 \, \Delta x^p \, \Delta t \left\| \frac{\partial \underline{\tilde{E}}}{\partial t} \right\|_\omega , \tag{7.6}$$

where $\varepsilon_0 = 0.1$ or 0.3. Although the efficiency could be increased by using a larger value of ε_0, a possible alternative to (7.3) is then given by

$$\|\underline{le\, H}\|_\omega < \varepsilon_1 \, \Delta x^p \, \|\underline{\tilde{E}}\|_\omega . \tag{7.7}$$

This approach adapts the timestep so that the local time error is a fraction of the spatial error estimate defined as in (7.7).

In order to assess the effectiveness of this approach an experiment was conducted using the two error controls on the test problem described in section 3.1. In this experiment a high-accuracy solution is computed using meshes of both 65×65 and 129×129 points and a small uniform timestep of $\Delta X^f/32$, where ΔX^f is from the finer resolution case. In comparing the methods the values of $\varepsilon_0 = 0.1$ and $\varepsilon_1 = 0.1$ were used on the 129×129 case. A comparison solution with the 65×65 point case is also included as this shows the effect of increasing mesh resolution. Figure 7.1 shows the percentage "errors" against this high accuracy solution in the key quantity of H_{cen}, the minimum film thickness, as the contact goes through the nonlinear reversal

TABLE 7.1
Computational performance comparison between different error control methods.

Grid dimension	Error control method	Timesteps taken	Iterations required
129×129	Fixed - small ΔT	36998	324065
129×129	Variable - using (7.7)	36628	111580
129×129	Variable - using (7.3)	140	2640

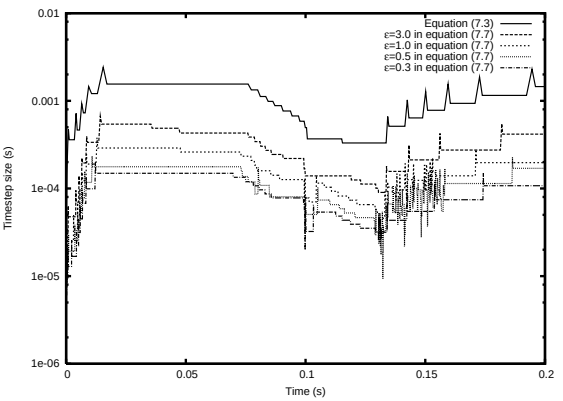


FIG. 7.2. *Comparison of timestep sizes for reversal of entrainment.*

phase of the calculation. The results show that in the first half of the simulation the errors using (7.7) are about 50% smaller than those using (7.3) relative to the tiny timestep case, but in the second half they are comparable. The figure concentrates on the important region around reversal. The computational cost of the three methods is demonstrated in Table 7.1. This table shows that the modified error control defined by (7.3) is a fraction of the cost of the approach defined by (7.7). This cost differential is reflected by the timesteps used by the two approaches as is shown in Figure 7.2. This figure compares the timestep sizes for the method of (7.3) (denoted as our method) with $\varepsilon = 0.1$ with the method of (7.7) with values of ε between 0.3 and 3. These values of ε are slightly larger than those used in the error balancing experiments of [1, 22]. The significant feature of these comparison experiments is that the error in a key quantity of interest, the central film thickness, is computed equally accurately in all cases, showing that the temporal error has been kept small. It is interesting to also see the variation in the timestep as the parameter ε in (7.3) is modified. This is shown in Figure 7.3 where it can be seen that the proposed method of step selection gives similar patterns of change. We have seen that sometimes using (7.7) may result in periods of relatively small steps without gaining additional accuracy. This seems to be related to the interplay between spatial and temporal error resulting in convergence difficulties for the multigrid solver.

The error control algorithm is implemented in the choice of the tolerances $RTOL$ and $ATOL$ being chosen at time t to be dynamically defined by

$$ATOL = \frac{1}{10}\sqrt{\frac{1}{N_X^C N_Y^C}\sum_{I=1}^{N_X^C}\sum_{J=1}^{N_Y^C}\left[\tilde{H}_{i,j}(t) - \tilde{H}_{I,J}(t)\right]^2} \tag{7.8}$$

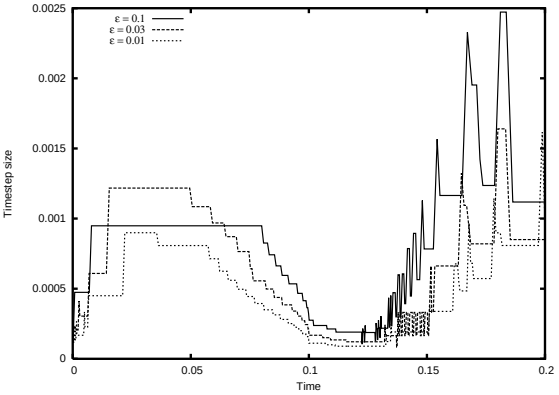


FIG. 7.3. *Comparison of timestep sizes when changing ε in* (7.3).

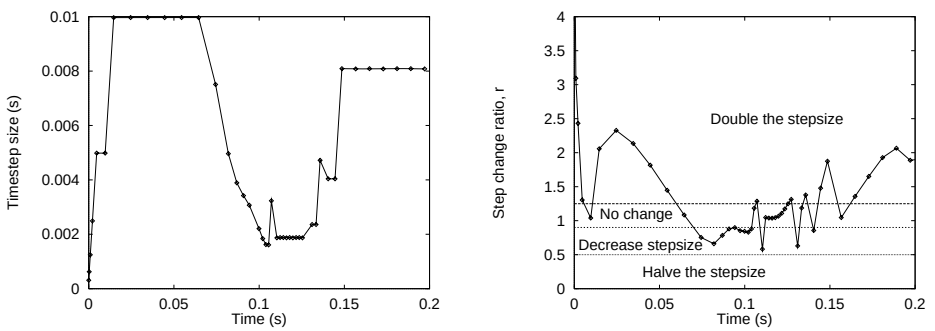
and

$$RTOL = ATOL \tag{7.9}$$

for fine mesh points (i, j) with coincident coarse points (I, J).

8. Examples. In this section we present examples of some typical, challenging, transient EHL cases. The first example in section 8.1 is that of reversal of entrainment. Second in section 8.2 is the overrolling of a single surface roughness feature. In section 8.3 we consider the situation where the load is varying. Finally, in section 8.4 the presence of a measured surface roughness pattern is simulated.

8.1. Reversal of entrainment. A clear example of the benefits of using variable timesteps can be seen in the case of the reversal of entrainment problem defined in section 3.1. In Figure 8.1(a) the points on the line indicate where timesteps have been taken. It can be seen that these are clustered around reversal and just after, as was desired. The actual timestep sizes can be seen in Figure 8.1(b) alongside the values of r, the stepsize change ratio. It was explained above that there can be advantages in not changing the stepsize too often. Choosing the new stepsize based on an a priori error test cannot guarantee that the new stepsize will be valid for more than one timestep. Thus having the range of values for r in (6.14) where the stepsize



(a) Timestep size *(b) Stepsize change ratio*

FIG. 8.1. *Timestep sizes and stepsize change ratios for reversal.*

TABLE 8.1
Computational performance comparison between fixed and variable timestepping codes.

Grid dimension	Fixed/ variable	Time taken (s)	Iterations required
65×65	Fixed	677	3386
65×65	Variable	458	1763
129×129	Fixed	1948	5139
129×129	Variable	1260	1604

TABLE 8.2
Film thickness (FT) comparisons at two reference times between fixed and variable timestepping codes.

Film thickness $\times 10^{-8}$ m		$t = 0.1$ s		$t = t_{min} \approx 0.108$ s	
		Min FT	Cen FT	Min FT	Cen FT
65×65	fixed	1.71	9.34	1.24	9.66
65×65	variable	1.81	9.41	1.28	9.80
129×129	fixed	2.51	9.29	1.97	9.87
129×129	variable	2.52	9.33	1.99	9.95

remains unchanged is important. The size of this region also governs how often the stepsize can change because if it is too small then the stepsize may be successively increased and decreased. This "chattering" effect, well known in the ODE community, may cause instabilities in the solution. This region is considered, for example, by Shampine and Gordon [32] and Hairer, Nørsett, and Wanner [19]. It is the range of values calculated for r in the error test, for which r should be set to 1 in (6.15).

The use of variable timestepping may require more cycles per timestep, but through the larger stepsizes these are being done considerably less often. It is also possible to limit the number of iterations per timestep if the convergence of individual steps is failing to satisfy the convergence test (5.9) quickly enough.

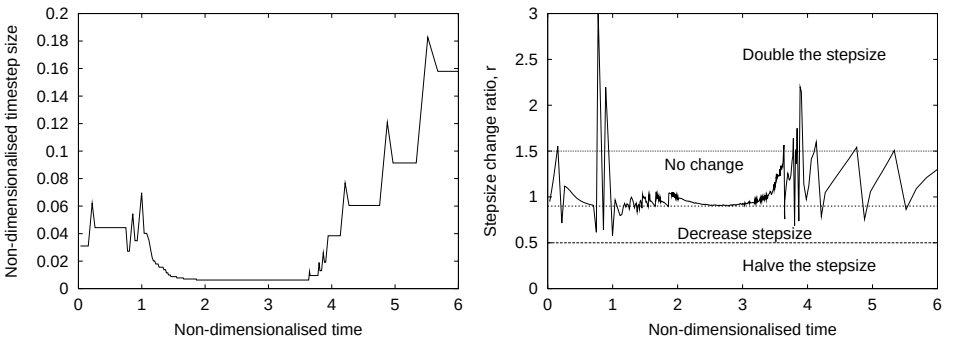
One of the stated aims for the use of variable timestepping was a reduction in the required computational time. Table 8.1 shows the computational time comparison between run times for fixed step ($\Delta T = \Delta X$) and variable timestep sizes. To confirm the validity of the results obtained, central and minimum film thicknesses are compared in Table 8.2 at two reference times: the point of reversal ($t = 0.1$ s) and the time that the minimum film thickness is achieved ($t = t_{min} \approx 0.108$ s). At these times the minimum film thickness is an order of magnitude less than the initial steady state. It is clear that the significant saving in computational work described produces results of similar accuracy.

8.2. Surface features. Following the experimental work of Kaneta, Sakai, and Nishikawa [21], Venner and Lubrecht [37] solved this transient example of a transverse ridge proceeding from left to right through the domain. For the local error control method to be considered effective it should be able to capture periods of rapid change in surface roughness cases where the relative importance of the asperity changes as it progresses through the contact. The undeformed geometry is given by

$$\mathcal{G}(X,Y,T) = \frac{X^2}{2} + \frac{Y^2}{2} - \mathcal{A} \times 10^{-10\left(\frac{X-X_d}{\mathcal{W}}\right)^2} \cos\left(2\pi \frac{X - X_d}{\mathcal{W}}\right), \tag{8.1}$$

TABLE 8.3
Parameters used for transverse ridge example, after Venner and Lubrecht [37].

Parameter	Value	Parameter	Value
E'	1.17×10^{11} Pa	Moes parameter, M	233
α	2.2×10^{-8} Pa^{-1}	Moes parameter, L	5.42
η_0	1.22 Pa s	material parameter, G	2.62×10^{3}
u_s	0.0215 m s^{-1}	speed parameter, U	8.8×10^{-12}
R_x	0.0127 m	load parameter, W	2.0×10^{-6}
a	1.84×10^{-4} m	ridge amplitude, $\mathcal{A}$	0.075
p_h	0.54×10^{9} Pa	ridge wavelength, $\mathcal{W}$	0.7



(a) Timestep size *(b) Stepsize change ratio, r*

FIG. 8.2. *Timestep sizes and stepsize change ratio during overrolling of a transverse ridge using variable timestepping.*

where $\mathcal{A}$ is the dimensionless amplitude of the ridge, $\mathcal{W}$ is the dimensionless wavelength of the ridge, and $X_d(T)$ is the dimensionless position of the ridge given by

$$X_d(T) = X_d(0) + \frac{2u_2T}{u_s}. \tag{8.2}$$

The parameters are as given in Table 8.3 with the ridge entering from the left-hand side of the domain, with $X_d(0) = -2.5$, and then progressing through the domain. They correspond to a ridge of height 0.2 μm and a width of 0.07 mm, approximately what was used by Kaneta, Sakai, and Nishikawa, and precisely what was used by Venner and Lubrecht. Initially the ridge is outside the area of influence on the solution, and the timestep is expected to be larger than the optimal timestep as the ridge passes through the contact region. Once the ridge has entered the cavitation region no further transient effects should be present, and hence the solutions should return to the steady state conditions, and the timestep size should increase.

The physical solutions obtained were as shown by Venner and Lubrecht [37], and hence the only part of interest is the timestep information. In Figure 8.2(a) the timestep size is shown. As expected it does rise to a maximum which then falls as the ridge enters the contact area at $T = 1.5$. This remains fairly constant until the ridge enters the cavitation region just after $T = 3.5$. Once out of the contact area the solution returns to steady state conditions and the timestep size increases dramatically. Consideration of the stepsize change parameter, r, in Figure 8.2(b) reveals how satisfactory the selected timestep size is, as the ridge passes through the contact area. The desire to increase the stepsize both initially and after overrolling is contrasted against the reduction in timestep size between $T = 1$ and $T = 2$.

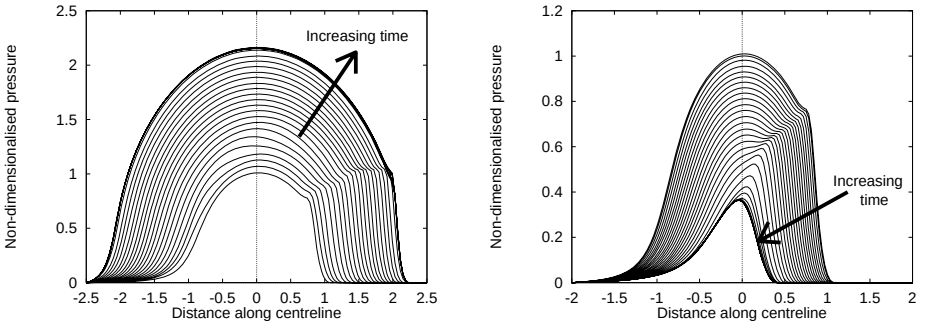


FIG. 8.3. (i) *Shock loading from* 0.45 *to* 0.90 *GPa in* 0.02 *s and* (ii) *unloading from* 0.45 *to* 0.22 *GPa in* 0.02 *s.*

8.3. Shock loading. The example of shock loading models the kind of change to the physical conditions experienced by the teeth on gears during meshing. Two kinds will be demonstrated in this section: shock loading and shock unloading. The load changes are of much greater amplitude than the oscillating loads investigated by Wijnant [40], and the timescale is very short.

The examples to be considered here both start from the initial conditions of the reversal example, shown in Table 3.1. The loading will take place by increasing the maximum Hertzian pressure to double its initial value for loading, or by halving it for the unloading case. Since this calculation is done using nondimensionalized versions of the equations as explained in section 3, it is the initial load against which the parameters are defined. In the nondimensionalization process the applied load is normally assumed to be constant, and thus, in the case of a varying load, a change to the conservation equation, (3.4), is necessary. This entails a modified right-hand side, hence changing the target sum of pressures. Thus (3.4) becomes

$$\frac{2\pi}{3}\Phi = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} P(X,Y)\,\mathrm{d}X\mathrm{d}Y, \tag{8.3}$$

where the variable load, Φ, is given by

$$\Phi = \left\{ \begin{array}{ll} 1, & t \le 0\text{s}, \\ 1 + 450t, & 0 \le t \le 0.02\text{s}, \\ 10, & t > 0.02\text{s}, \end{array} \right\} \text{for increasing loading,} \tag{8.4}$$

$$\Phi = \left\{ \begin{array}{ll} 1, & t \le 0\text{s}, \\ 1 - 45t, & 0 \le t \le 0.02\text{s}, \\ \frac{1}{10}, & t > 0.02\text{s}, \end{array} \right\} \text{for decreasing loading.} \tag{8.5}$$

Pressure solutions for the two cases are very different. The centerline pressure is shown at selected timesteps in Figure 8.3, with the arrow indicating the direction of increasing time of solution plots. The increase in both the size of the contact area and maximum pressure is evident.

The unloading example has a very different behavior, as shown in the right side of Figure 8.3, because the problem has dropped out of the EHL regime to just being hydrodynamically lubricated instead. This means that the pressure spike has completely disappeared, and the deformation is now minimal, with no sidelobes present at all.

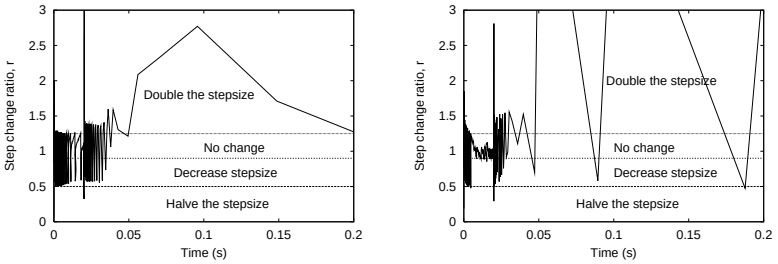


FIG. 8.4. *Shock loading and unloading: Stepsize change ratios.*

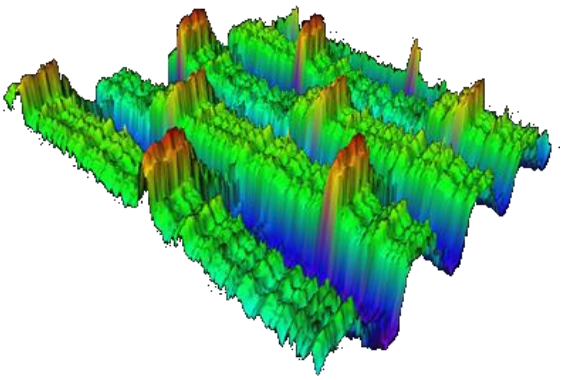


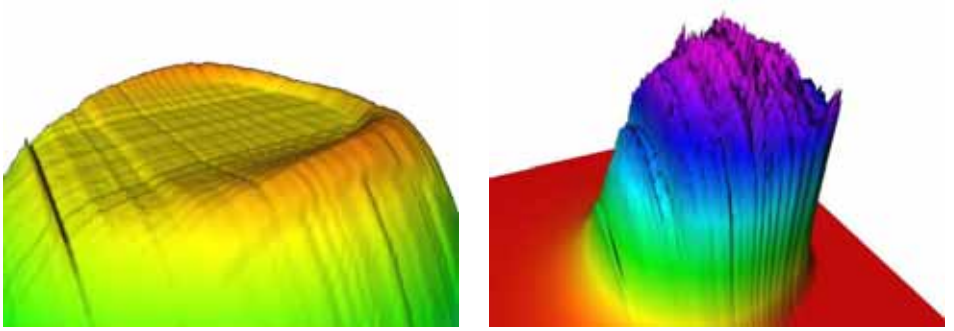
FIG. 8.5. *Roughness pattern applied across contact.*

The timestep size is chosen based on the local error estimate which in turn depends on derivatives of the solution. It is therefore expected that when the system has reached the final steady state solution beyond 0.02 s the timestep should be much larger than in the initial stages. For the shock loading example the stepsize change ratio, r, is shown on the left of Figure 8.4. It can be clearly seen that the timestep does not increase until after the end of the loading period.

The stepsize change ratios for the shock unloading example is shown on the right side of Figure 8.4. Similar behavior to the shock loading case can be observed, although with a significantly less smooth increase in the timestep size.

8.4. Real surface roughness. A recent facet of EHL research has been the solution of problems with real surface roughness by a number of research groups; see, e.g., [8]. On a microscopic scale real surfaces of contacts are not defined by parabolas, nor are any features, such as bumps and dents, perfectly geometric shapes. In fact the surface is covered in small asperities. In an EHL contact the deformation of the solid will seek to smooth out these irregularities, and hence the pressure profile will be related to the local surface geometry. In fact for each individual asperity being flattened it is possible to see that it has its own EHL contact problem—just on a much smaller physical scale.

For this example we have imposed the measured roughness pattern shown in Figure 8.5 onto one of the surfaces. This therefore rolls through the contact over time, in the same manner as the surface feature simulated in section 8.2. An example, with the amplitude of the roughness applied at a nominal 25%, is shown in Figure 8.6.



(a) Contact shape (b) Pressure profile

FIG. 8.6. *Roughness pattern applied across contact.*

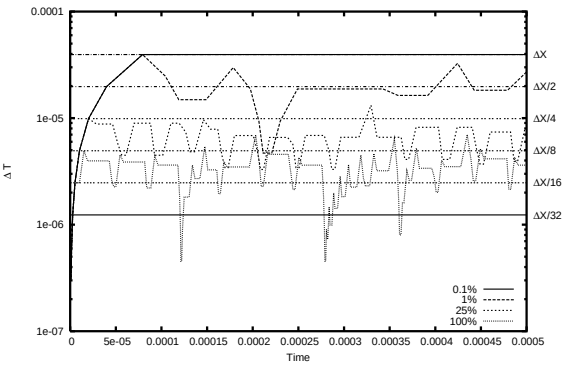


FIG. 8.7. *Timestep size for different amplitudes of roughness.*

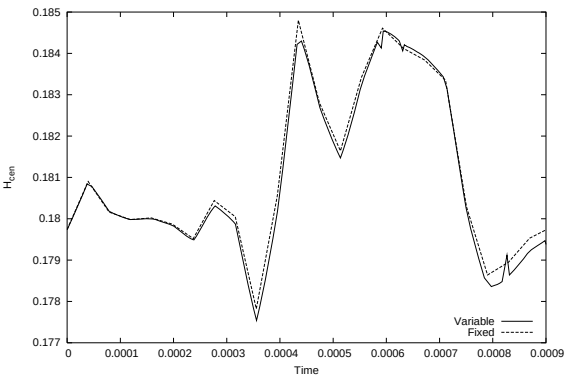


FIG. 8.8. *Central film thickness using fixed and variable timestepping approaches.*

In the previous examples it has been shown how the use of variable timestepping increases the efficiency of the solver without damaging the accuracy. For surface roughness cases the situation is slightly different in that variable timestepping will allow the tracking of the progression of each asperity through the domain. This is shown well in Figure 8.7 where the timestep size is monitored for increasing roughness

amplitudes. It can be seen how the selected timestep size diminishes as the contacts become less smooth, as we would expect.

The increase in accuracy that may be attained through the use of temporal adaptivity is shown in Figure 8.8, which shows the contrast in film thickness solution values at the center of the contact region. It is clearly seen how the fixed timestep approach both diverges from the variable timestepping solution and also misses out on the subtimestep scale features.

9. Conclusions. In this paper we have shown how a variable timestep approach has been used to provide an efficient method for solving transient EHL problems. It has been explained how the adaptive methods proposed fit into the standard multilevel schemes used for numerical solutions to EHL problems.

Variable timestepping has been shown, by experiments, to substantially reduce the required work while maintaining the same level of solution accuracy. The overhead in calculating new stepsizes is small, relative to the increase in performance. Changing the stepsize away from ΔT, within predefined limits, has been seen to pose no problems for the solver.

Examples of representative transient EHL situations have been considered, covering three different physical changes. The problem of reversal of entrainment was showcased and compared against the experimental results of Scales et al. [30]. The progression of the disc of highly viscous lubricant through the center of the contact just after reversal was automatically picked up by the variable timestep approach, and the results were shown to be just as accurate as for fixed timesteps.

The problem of surface roughness has been considered. It was seen how the overrolling of a single surface feature required small steps while the asperity passes through the center of the contact. One conclusion to be drawn from this example could be that for cases with real surface roughness, the timestep could be expected to be constant throughout. In the example of a full surface roughness pattern it has been seen how this level of constant timestep is related to the amplitude of the roughness pattern and may be much smaller than the mesh size. It is in cases like this where grid adaptation [15, 17] will be very beneficial.

The other case considered was that of shock loading, where the loading on the contact changed dramatically over a very short length of time. Again it was seen how the timesteps were very small as the major changes in the loading changed, but as the system reached a steady state the allowable timestep size increased.

The grid adaptation techniques shown to be beneficial in [15, 17] have not been included, hence allowing the advantages of the variable timestepping methods to be considered independently of other factors. Similarly the advantages of parallelism [14, 18] have also been neglected since they have no impact on the accuracy of the solution obtained.

Future work in this area should consider the use of adjoint-based approaches with regard to the direct estimation and control of errors in quantities such as the central and minimum film thicknesses, or other quantities of interest such as the friction [17, 20].

Acknowledgments. Thanks to Roger Fairlie, Laurence Scales of Shell Global Solutions, and the EPSRC CASE scheme.

REFERENCES

[1] M. Berzins, *Temporal error control for convection-dominated equations in two space dimensions*, SIAM J. Sci. Comput., 16 (1994), pp. 558–580.

[2] A. Brandt and A. A. Lubrecht, *Multilevel matrix multiplication and fast solution of integral equations*, J. Comput. Phys., 90 (1990), pp. 348–370.

[3] K. E. Brenan, S. L. Campbell, and L. R. Petzold, *Numerical Solution of Initial-Value problems in Differential-Algebraic Equations*, Classics Appl. Math. 14, SIAM, Philadelphia, 1995.

[4] W. L. Briggs, V. E. Hensen, and S. McCormick, *A Multigrid Tutorial*, 2nd ed., SIAM, Philadelphia, 2000.

[5] Y. Cao and L. Petzold, *A posteriori error estimation and global error control for ordinary differential equations by the adjoint method*, SIAM J. Sci. Comput., 26 (2004), pp. 359–374.

[6] D. Dowson and G. R. Higginson, *Elasto-hydrodynamic Lubrication, The Fundamentals of Roller and Gear Lubrication*, Permagon Press, Oxford, Great Britain, 1966.

[7] D. Dowson and C. M. Taylor, *Cavitation in bearings*, in Annual Review of Fluid Mechanics Vol. 11, Annual Reviews, Palo Alto, CA, 1979, pp. 35–66.

[8] C. D. Elcoate, H. P. Evans, T. G. Hughes, and R. W. Snidle, *Transient elastohydrodynamic analysis of rough surfaces using a novel coupled differential deflection method*, Proc. Instn. Mech. Engrs. J. Engineering Tribology Part J, 215 (2001) pp. 319–337.

[9] D. Estep, *A posteriori error bounds and global error control for approximation of ordinary differential equations*, SIAM J. Numer. Anal., 32 (1995), pp. 1–48.

[10] R. Fairlie, C. E. Goodyer, M. Berzins, and L. E. Scales, *Numerical modelling of thermal effects in elastohydrodynamic lubrication solvers*, in Tribological Research and Design for Engineering Systems, Proceedings of the 29th Leeds-Lyon Symposium on Tribology, D. Dowson et al., eds., Elsevier, Amsterdam, 2003, pp. 675–683.

[11] P. H. Gaskell, P. K. Jimack, M. Sellier, and H. M. Thompson, *Efficient and accurate time-adaptive multigrid simulations of droplet spreading*, Internat. J. Numer. Methods Fluids, 45 (2004) pp. 1161–1186.

[12] R. Gohar, *Elastohydrodynamics*, 2nd ed., Imperial College Press, London, 2001.

[13] C. E. Goodyer, *Adaptive Numerical Methods for Elastohydrodynamic Lubrication*, Ph.D. thesis, University of Leeds, Leeds, UK, 2001.

[14] C. E. Goodyer, M. Berzins, P. K. Jimack, and L. E. Scales, *Grid-based numerical optimisation in a problem solving environment*, in Proceedings of the All Hands Meeting 2003, S. Cox, ed., EPSRC, Swindon, UK, 2003, pp. 854–861.

[15] C. E. Goodyer, R. Fairlie, M. Berzins, and L. E. Scales, *Adaptive mesh methods for elastohydrodynamic lubrication*, in ECCOMAS CFD 2001: Computational Fluid Dynamics Conference Proceedings, Institute of Mathematics and Its Applications, Minneapolis, MN, 2001.

[16] C. E. Goodyer, R. Fairlie, M. Berzins, and L. E. Scales, *Adaptive techniques for elastohydrodynamic lubrication solvers*, in Tribology Research: From Model Experiment to Industrial Problem, Proceedings of the 27th Leeds-Lyon Symposium on Tribology, G. Dalmaz et al., eds., Elsevier, Amsterdam, 2001, pp. 709–719.

[17] C. E. Goodyer, R. Fairlie, D. E. Hart, M. Berzins, and L. E. Scales, *Calculation of friction in steady-state and transient EHL simulations*, in Transient Processes in Tribology, Proceedings of the 30th Leeds-Lyon Symposium on Tribology, G. Dalmaz et al., eds., Elsevier, Amsterdam, 2004, pp. 579–590.

[18] C. E. Goodyer, J. Wood, and M. Berzins, *A parallel grid based PSE for EHL problems*, in Applied Parallel Computing, Proceedings of PARA'02, Lecture Notes in Comput. Sci. 2367, J. Fagerholm et al., eds., Springer-Verlag, New York, 2002, pp. 523–532.

[19] E. Hairer, S. P. Nørsett, and G. Wanner, *Solving Ordinary Differential Equations* I: *Nonstiff Problems*, Springer-Verlag, New York, 1980.

[20] D. E. Hart, M. Berzins, C. E. Goodyer, P. K. Jimack, and L. E. Scales, *Adjoint error estimation for EHL-like models*, Internat. J. Numer. Methods Fluids, 47 (2005), pp. 1069–1075.

[21] M. Kaneta, T. Sakai, and H. Nishikawa, *Optical interferometric observations of the effects of a bump on point contact EHL*, ASME J. Tribology, 114 (1992), pp. 779–784.

[22] J. Lawson, M. Berzins, and P. M. Dew, *Balancing space and time errors in the method of lines for parabolic equations*, SIAM J. Sci. Statist. Comput., 12 (1991), pp. 573–594.

[23] A. Logg, *Multi-adaptive time integration*, Appl. Numer. Math., 48 (2004), pp. 339–354.

[24] A. A. Lubrecht, W. E. ten Napel, and R. Bosma, *Multigrid, an alternative method of calculating film thicknesses and pressure profiles in elastohydrodynamically lubricated line contacts*, ASME J. Tribology, 108 (1986), pp. 551–556.

[25] A. A. Lubrecht, W. E. ten Napel, and R. Bosma, *Multigrid, an alternative method of solution for two-dimensional elastohydrodynamically lubricated point contact calculations*, ASME J. Tribology, 109 (1987), pp. 437–443.
[26] K.-S. Moon, A. Szepessy, R. Tempone, and G. Zouraris, *Hyperbolic differential equations and adaptive numerics*, in Theory and Numerics of Differential Equations (Durham, 2000), Universitext, Springer-Verlag, Berlin, 2001, pp. 231–280.
[27] E. Nurgat, M. Berzins, and L. E. Scales, *Solving EHL problems using iterative, multigrid and homotopy methods*, ASME J. Tribology, 121 (1999), pp. 28–34.
[28] A. I. Petrusevich, *Fundamental conclusions from the contact-hydrodynamic theory of lubrication*, Izv. Akad. Nauk. SSSR (OTN), 2 (1951), p. 209.
[29] C. J. A. Roelands, *Correlational Aspects of the Viscosity-Temperature-Pressure Relationship of Lubricating Oils*, Ph.D. thesis, Technische Hogeschool Delft, Delft, The Netherlands, 1966.
[30] L. E. Scales, J. E. Rycroft, N. R. Horswill, and B. P. Williamson, *Simulation and observation of transient effects in elastohydrodynamic lubrication*, SP-1182, in SAE International Fuels and Lubricants Meeting, Dearborn, MI, 1996, pp. 23–34.
[31] L. F. Shampine, *Implementation of implicit formulas for the solution of ODEs*, SIAM J. Sci. Statist. Comput., 1 (1980), pp. 103–118.
[32] L. F. Shampine and M. K. Gordon, *Computer Solution of Ordinary Differential Equations: The Initial Value Problem*, W. H. Freeman and Co., San Francisco, CA, 1975.
[33] F. Stenger, *Numerical Methods Based on Sinc and Analytic Functions*, Springer Ser. Comput. Math. 20, Springer-Verlag, New York, 1993.
[34] F. Stenger, *private communication*, 2004.
[35] U. Trottenberg, C. Oosterlee, and A. Schüller, *Multigrid*, Academic Press, New York, 2001.
[36] C. H. Venner, *Multilevel Solution of the EHL Line and Point Contact Problems*, Ph.D. thesis, University of Twente, Endschende, The Netherlands, 1991.
[37] C. H. Venner and A. A. Lubrecht, *Numerical simulation of a transverse ridge in a circular EHL contact under rolling/sliding*, ASME J. Tribology, 116 (1994), pp. 751–761.
[38] C. H. Venner and A. A. Lubrecht, *Multilevel Methods in Lubrication*, Elsevier, Amsterdam, 2000.
[39] B. Watremetz, F. Colin, C. H. Venner, and A. A. Lubrecht, *Optimum time step in a transient EHL contact*, in Transient Processes in Tribology, Proceedings of the 30th Leeds-Lyon Symposium on Tribology, G. Dalmaz et al., eds., Elsevier, Amsterdam, 2004, pp. 591–600.
[40] Y. Wijnant, *Contact Dynamics in the Field of Elastohydrodynamic Lubrication*, Ph.D. thesis, University of Twente, Endschende, The Netherlands, 1998.
[41] S. R. Wu, *A penalty formulation and numerical approximation of the Reynolds-Hertz problem of elastohydrodynamic lubrication*, Internat. J. Engrg. Sci., 24 (1986), pp. 1001–1013.
[42] S. R. Wu and J. T. Oden, *A note on some mathematical studies on elastohydrodynamic lubrication*, Internat. J. Engrg. Sci., 25 (1987), pp. 681–690.
[43] S. R. Wu and J. T. Oden, *Convergence and error estimates for finite element solutions of elastohydrodynamic lubrication*, Comput. Math. Appl., 13 (1987), pp. 583–593.

SIAM J. SCI. COMPUT.
Vol. 28, No. 2, pp. 651–681
© 2006 Society for Industrial and Applied Mathematics

THE RIEMANN PROBLEM FOR THE EULER EQUATIONS WITH NONCONVEX AND NONSMOOTH EQUATION OF STATE: CONSTRUCTION OF WAVE CURVES*

SIEGFRIED MÜLLER† AND ALEXANDER VOß†

Abstract. The Riemann problem is considered for the investigation of nonclassical wave phenomena in Bethe–Zel'dovich–Thompson (BZT)-fluids and fluids which undergo a phase transition, i.e., the isentropes suffer kinks at the saturation boundaries. These fluids are modeled by a general equation of state that also takes into account regions of negative nonlinearity where the isentropes are nonconvex in the pressure-volume plane. Here the physical model is based on thermodynamical equilibrium. Both nonsmoothness as well as nonconvexity significantly influence the composition of the wave curves which are crucial for determining the solution for the Riemann problem for the Euler equations. This work gives explicit criteria for the composition of these wave curves by elementary wave types such as shocks, simple waves, and composite waves. Numerous examples for different wave curve compositions are presented.

Key words. Riemann problem, Euler equations, phase transition, negative nonlinearity, BZT-fluids

AMS subject classifications. 35L65, 35L67, 58K05, 76N15

DOI. 10.1137/040619909

1. Introduction.

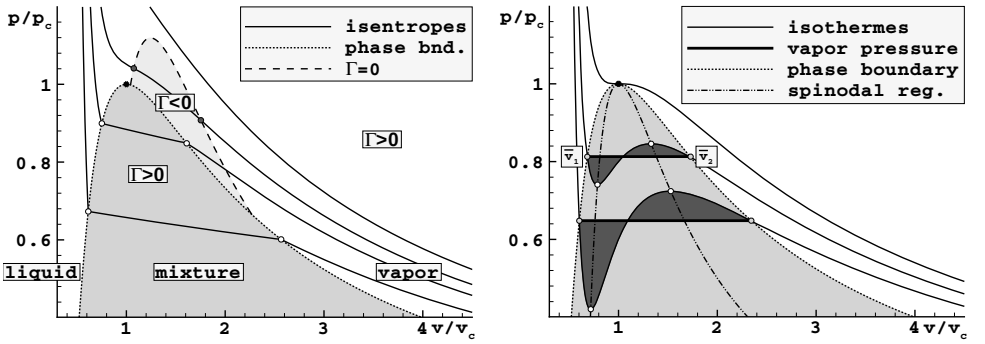
The dynamics of compressible flows has a strong influence on the design of aircraft and turbomachinery. In many applications the fluid is modeled by a perfect gas. In the range of perfect gas theory only two types of waves are possible and allowed by the entropy inequality—the [illegible] and the [illegible]. A compression shock is a discontinuity where the pressure of the fluid increases while the shock is passing, whereas in an expansion wave the pressure decreases.

However, the perfect gas model is no longer appropriate when dealing with so-called BZT-fluids, named after Bethe, Zel'dovich, and Thompson. These fluids, e.g., high molecular fluorcarbons and hydrocarbons, are characterized by a large heat capacity and are considered a subgroup of so-called retrograde fluids. BZT-fluids exhibit a region of negative nonlinearity near the curve of saturated vapor in phase space. This region is characterized by the nonconvexity of the isentropes in the pressure-volume plane where the fundamental derivative of gas dynamics Γ becomes negative due to the negative curvature of the isentropes; see Figure 1. From experiments, e.g., [BBKN83], it is known that there may occur nonclassical effects such as [illegible] and [illegible], as well as waves composed of adjacent shocks and simple waves. Here an expansion shock is a discontinuity where the pressure of the fluid decreases while the shock is passing, whereas in a compression wave the pressure increases. In short, most of the classical inequalities and effects are reversed in this area.

Below the critical point in phase space (see again Figure 1), liquid and vapor coexist in a so-called mixture region. Due to the involved phase changes new phe-

*Received by the editors December 1, 2004; accepted for publication (in revised form) December 13, 2005; published electronically May 12, 2006.
http://www.siam.org/journals/sisc/28-2/61990.html

†Institut für Geometrie und Praktische Mathematik, RWTH Aachen, D-52056 Aachen, Germany (mueller@igpm.rwth-aachen.de, voss@igpm.rwth-aachen.de). The work of the second author was performed with funding by the Deutsche Forschungsgemeinschaft in the Priority Research Program "Analysis and Numerics for Conservation Laws" and the DFG Leibniz Program.



(a) Isentropes for a BZT fluid.

(b) Isotherms and Maxwell's equal area rule for a BZT fluid.

FIG. 1. *Phase space projected onto pressure-volume plane.*

nomena such as [illegible] and [illegible] have been observed; cf. [DTMS79, TK83, TCK86, TCM$^+$87].

The aforementioned phenomena can be analytically investigated by the exact solution of a Riemann problem for compressible fluid flow where a general equation of state (EOS) is employed, capable of reflecting the negative nonlinearity of BZT-fluids as well as phase transitions and mixture states, respectively. Moreover, such an exact Riemann solver is helpful in many other respects. For instance, it may serve as a validation tool for numerical schemes since it is known that some of them do not catch all effects. Additionally, it can be used to determine initial conditions for experiments exhibiting the aforementioned phenomena.

The crucial point in solving the Riemann problem for compressible fluids is the construction of the wave curves. So far it has been an open problem how to [illegible] a practical method for the computation of the wave curves from the known theoretical construction, and thus to determine the exact Riemann solution in the general setting [illegible] phase transition. For this purpose we extend here the well-known concept of wave curve composition and construction (cf. [Wen72a, Wen72b]) to the [illegible] case for the first time. Following Liu's concept [Liu75] to characterize the admissibility of wave types by the monotonicity of the corresponding wave speeds, we additionally have to take into account that the wave curves may suffer kinks and therefore wave speeds may jump. Here we present a complete set of conditions characterizing those states on a wave curve where the wave type changes. Then these conditions are employed to compose a wave curve out of different elementary waves such as shock waves, simple waves, and composite waves in a [illegible] way. The construction is supplemented by numerous examples for different compositions. Details can be found at length in [Voß05]. Note that this paper is primarily concerned with the design of an algorithm for the computation of the wave curves. In particular, no convergence result is claimed. The existence, uniqueness, and regularity have been investigated in full detail in earlier works by Liu [Liu75] as well as Iguchi and LeFloch [IL03] for general systems of hyperbolic conservation laws.

Finally, we remark that the construction is based on certain assumptions. Since these assumptions are properly addressed in different parts of the paper we will summarize them here for the convenience of the reader: (1) We assume that the underlying system of conservation laws, namely, the Euler equations, modeling inviscid,

compressible fluid flow, is hyperbolic. Because of this assumption the different wave families do not coincide in the space-time continuum. Consequently, we exclude states corresponding to three-phase mixtures. (2) The nonlinear wave families are assumed to linearly degenerate only at isolated points. The flux functions also exhibit discontinuous derivatives at isolated points. This implies that the wave curves corresponding to the nonlinear fields are transversal to the submanifolds of degeneration points and kinks where the curvature of the isentropes vanishes or suffers a kink. (3) The presented construction of the wave curves is only admissible if the isentropes do not cross each other in the pressure-volume plane. This is guaranteed provided that the Grüneisen coefficient is positive, i.e., the material expands upon heating at constant pressure. A well-known exception is water that contracts upon heating near 0° C and 1 bar. (4) According to the above assumptions the solution of the Riemann problem may be reduced to constructing wave curves and determining an intersection point of the wave curves corresponding to the nonlinear fields in the pressure-velocity plane. To ensure uniqueness of the intersection point the wave curves have to be monotone. This holds true provided that the introduced by Smith [Smi79] on the EOS is satisfied; see [MP89]. (5) The wave curves need to be parameterized. Here we employ the pressure because this quantity varies monotonically along the wave curves as long as the introduced by Smith [Smi79] on the EOS is satisfied; see [MP89]. So far there is no evidence that the weak condition is ever violated. (6) The phase transition is assumed to take place instantaneously; i.e., the system is in equilibrium. Therefore, nonequilibrium phenomena such as metastable states are not considered here. Taking these effects into account would spoil the self-similarity of the Riemann solution. (7) Throughout the construction the will be imposed. Therefore, nonclassical shocks will not be included; cf. [HL00, Flo02].

1.1. Physical background. We first recall the modification of a cubic EOS, for instance, the EOS, in order to take into account the aformentioned physical regions. We remark that this approach is not restricted to the van der Waals EOS but can be applied, for instance, to more appropriate cubic EOSs such as the Redlich–Kwong or Peng–Robinson EOSs.

The van der Waals EOS is one of the simplest model equations. It is given by

$$\mathtt{p}(\mathtt{v},\mathtt{T}) = \frac{\mathcal{R}\,\mathtt{T}}{\mathtt{v}-\mathtt{b}} - \frac{\mathtt{a}}{\mathtt{v}^2}, \tag{1.1}$$

with pressure $\mathtt{p}$, specific volume $\mathtt{v}$, temperature $\mathtt{T}$, specific gas constant $\mathcal{R}$, internal pressure $\mathtt{a}/\mathtt{v}^2$, and covolume $\mathtt{b}$. This is an incomplete EOS and has to be completed by a caloric EOS for the internal energy $\mathtt{e}$ and the entropy $\mathtt{s}$,

$$\mathtt{e}(\mathtt{v},\mathtt{T}) = \mathtt{e}_0 + \int_{\mathtt{T}_0}^{\mathtt{T}} \mathtt{c}_\mathtt{v}(\mathtt{T})\ \mathtt{dT} - \frac{\mathtt{a}}{\mathtt{v}}, \qquad \mathtt{s}(\mathtt{v},\mathtt{T}) = \mathtt{s}_0 + \int_{\mathtt{T}_0}^{\mathtt{T}} \frac{1}{\mathtt{T}}\,\mathtt{c}_\mathtt{v}(\mathtt{T})\ \mathtt{dT},$$

such that the thermodynamic relation

$$\mathtt{de} = -\mathtt{p}\ \mathtt{dv} + \mathtt{T}\ \mathtt{ds} \tag{1.2}$$

holds. Here $\mathtt{c}_\mathtt{v}$ denotes the heat capacity at constant volume, $\mathtt{e}$ the internal energy, and $\mathtt{e}_0$ the heat of formation. Note that a BZT-fluid is characterized by a large heat capacity, i.e., $\mathcal{R}/\mathtt{c}_\mathtt{v} \ll 1$.

The pressure equation (1.1) shows an unphysical inclination in the mixture area. If we consider an isotherm for a temperature below the critical one (see Figure 1(b) for a BZT-fluid), we observe two local extrema, called spinodal points. They correspond to the endpoints of supersaturated vapor (maximum) and overexpanded liquid (minimum), respectively. Between these two points the derivative $\mathtt{p_v}(\mathtt{v},\mathtt{T})$ is positive and, hence, physically excluded. In order to provide physically meaningful data, we calculate the equilibrium or vapor pressure $\bar{\mathtt{p}}$, as a function of the temperature using the well-known Maxwell's construction principle. Summarized, for every temperature below the critical one, two volumes $\bar{\mathtt{v}}_1$ and $\bar{\mathtt{v}}_2$ and the equilibrium pressure $\bar{\mathtt{p}}$ are computed such that the equal-area-rule

$$\int_{\bar{\mathtt{v}}_1}^{\bar{\mathtt{v}}_2} \mathtt{p}(\mathtt{v},\mathtt{T})\ \mathtt{dv} = \bar{\mathtt{p}}\,(\bar{\mathtt{v}}_2 - \bar{\mathtt{v}}_1), \qquad\qquad \mathtt{p}(\bar{\mathtt{v}}_1,\mathtt{T}) = \mathtt{p}(\bar{\mathtt{v}}_2,\mathtt{T}) = \bar{\mathtt{p}},$$

holds. Performing this for all $\mathtt{T} < \mathtt{T_c}$ yields the saturated liquid and vapor curves, i.e., the phase boundaries, and hence the mixture area according to the underlying EOS. Then we compose a new pressure EOS by taking the equilibrium pressure for parameters $(\mathtt{v},\mathtt{T})$ inside the mixture region and the original (1.1) otherwise.

In the same manner we modify the corresponding caloric EOS and the entropy $\mathtt{s}$. Inside the mixture region these quantities are calculated as convex combinations of the equilibrium states at the phase boundary, i.e.,

$$\bar{\mathtt{e}} = (1-\nu)\,\bar{\mathtt{e}}_1 + \nu\,\bar{\mathtt{e}}_2, \qquad \bar{\mathtt{s}} = (1-\nu)\,\bar{\mathtt{s}}_1 + \nu\,\bar{\mathtt{s}}_2, \qquad \nu = \frac{\bar{\mathtt{v}} - \bar{\mathtt{v}}_1}{\bar{\mathtt{v}}_2 - \bar{\mathtt{v}}_1},$$

where ν is the steam mass fraction. For simplicity, we choose $\mathtt{c_v}$ above to be constant. In particular, these linear combinations also satisfy the thermodynamic relation (1.2).

For an extensive discussion concerning the physical properties of retrograde and BZT-fluids we refer to [Cra91, Tho91]. The treatment of the mixture regions and the derivation of the Maxwell construction from evaluating the Gibbs-function at the saturation curves can be found in [Bae88].

In the present paper we restrict our attention to systems being in equilibrium, which is tantamount to assuming instantaneous phase transitions. Following [MP89], such an equilibrium model is reasonable if the time for transition is short relative to the time scale of the flow. When this is not the case, then nonequilibrium phenomena such as the metastable state play a role.

1.2. Mathematical background. We now briefly summarize the fundamental concepts and notions for treating systems of hyperbolic conservation laws in one spatial dimension. Among them is a discussion on strict hyperbolicity and genuine nonlinearity in a generalized sense; i.e., we take into account smooth as well as Lipschitz continuous flux functions.

We remark that recent mathematically oriented literature, e.g., [Daf00] or [Flo02], tends to use "well-known" terms like [illegible] or [illegible] in a more general sense which is thus detached from their original, physical meaning. This makes allowance for the ongoing research in the field of hyperbolic conservation laws, for instance, in the case of nonclassical shock waves or retrograde fluids used in gas dynamics, leading to formerly thought "impossible" physical effects [Tho91].

A system of conservation laws

$$\mathtt{q}_t + \mathtt{f}(\mathtt{q})_x = 0, \quad \mathtt{q}(t,x) \in V,\ x \in \mathbb{R},\ t > 0, \tag{1.3}$$

with state vector $\mathtt{q} : \mathbb{R}_+ \times \mathbb{R} \to V \subset \mathbb{R}^m$ and flux function $\mathtt{f} : V \to \mathbb{R}^m$ is said to be a strictly hyperbolic system if the Jacobian $\partial_{\mathtt{q}}\mathtt{f}$ of a [illegible] flux $\mathtt{f}$ admits m real and distinct eigenvalues

$$\lambda_1(\mathtt{q}) < \cdots < \lambda_m(\mathtt{q}) \quad \forall \mathtt{q} \in V.$$

They are referred to as characteristic speeds, associated to the basis of right eigenvectors $\{\mathtt{r}_k(\mathtt{q})\}_{k=1,\dots,m}$. Additionally, they are supposed to be normalized; i.e., $\mathtt{l}_l^T \mathtt{r}_k = \delta_{l,k}$, where $\{\mathtt{l}_l\}_{l=1,\dots,m}$ denote the corresponding left eigenvectors. We emphasize that the assumption of strict hyperbolicity is crucial for solving the Riemann problem by determining the intersection of wave curves. However, there are physical applications where nonstrictly hyperbolic PDEs are involved; see, for instance, [IMPT88]. If initial data at time $t = 0$ is prescribed by a given function $\mathring{\mathtt{q}}$ consisting of piecewise constant data

$$\mathring{\mathtt{q}}(x) = \begin{cases} \mathtt{q}_l, & x < 0, \\ \mathtt{q}_r, & x > 0, \end{cases} \tag{1.4}$$

the initial value problem is referred to as the Riemann problem. As will be seen later, the solution of a Riemann problem, i.e., the type of occurring waves patterns, their compositions, and speeds, heavily depends on the [illegible] of the characteristic speeds λ_k along trajectories tangent to $\mathtt{r}_k$.

The pair $(\lambda_k, \mathtt{r}_k)$ is called the k-characteristic field. It is called genuinely nonlinear if the quantity

$$\zeta_k(\mathtt{q}) = [\nabla_{\mathtt{q}} \lambda_k \cdot \mathtt{r}_k](\mathtt{q})$$

does not vanish, i.e., $\zeta_k \not\equiv 0$. It is said to be linearly degenerated when $\zeta_k \equiv 0$. With this in mind, the definition of genuine nonlinearity can be read as a directional derivative of λ_k along an integral curve defined by the eigenvector $\mathtt{r}_k$.

In general, the flux function is assumed to be either smooth or Lipschitz continuous. Concerning the Euler equations and a cubic EOS we have a C^1-function away from phase boundaries which is just continuous on the saturation lines. At these states the derivatives suffer from kinks. In order to speak of hyperbolic systems and genuinely nonlinear characteristic fields even in the case of Lipschitz continuous flux functions, one has to adapt the notions above. This has been done in [CLT01], based on the definition of generalized derivatives and related important analytical results still valid for Lipschitz functions, which can be found in [Cla83] and [Fil88].

One can show that the initial data of a Cauchy or Riemann problem is propagated along straight (characteristic) lines in the x-t plane. Hence, it admits self-similar (weak) solutions of the form $\mathtt{q} = \mathtt{q}(\tilde{\xi})$ with $\tilde{\xi} = \frac{x}{t}$ and the solution additionally satisfies the boundary conditions $\mathtt{q}(-\infty) = \mathtt{q}_l$ and $\mathtt{q}(\infty) = \mathtt{q}_r$. According to [Daf00] such a self-similar weak solution, $\mathtt{q} = \mathtt{q}(\tilde{\xi})$, is a bounded and measurable function $\mathtt{q} \in \mathtt{BV}_{\mathrm{loc}}$. In particular, it is differentiable on $(-\infty, +\infty)$ almost everywhere and one-sided limits $\mathtt{q}(\tilde{\xi}) = \mathtt{q}(\tilde{\xi}-) = \mathtt{q}(\tilde{\xi}+)$ exist, except possibly on a countable set of $\tilde{\xi}$. Furthermore, it solves the ordinary differential equation

$$[-\tilde{\xi}\, I + [\partial_{\mathtt{q}}\mathtt{f}](\mathtt{q})]\,(\mathtt{d}_{\tilde{\xi}}\mathtt{q}) = 0, \tag{1.5}$$

in the sense of measures. We again refer to [Daf00] for details on the correlation of $\mathtt{BV}$ and the space of measures. The interval $(-\infty, +\infty)$ can be decomposed into

three disjoint sets $\mathcal{N}, \mathcal{H}$, and $\mathcal{R}$, where $\mathcal{N}$ denotes the set on which $\mathtt{q}$ is constant, i.e., $\mathtt{d}_{\tilde{\xi}}\mathtt{q} = 0$, on (at most) a countable union of disjoint open intervals. $\mathcal{H}$ is the (at most) countable set of points of jump discontinuities of $\mathtt{q}$ and for each $\tilde{\xi} \in \mathcal{H}$ the Rankine–Hugoniot condition holds. At least, the (possibly empty) set $\mathcal{R}$ consists of points of continuity of $\mathtt{q}$, where at each $\tilde{\xi} \in \mathcal{R}$ we have

$$\lambda_k(\mathtt{q}(\tilde{\xi})) = \tilde{\xi} \quad \text{for some } k \in \{1, \dots, m\}. \tag{1.6}$$

If $\mathtt{q}$ is also differentiable at $\tilde{\xi} \in \mathcal{R}$, (1.5) implies

$$\mathtt{d}_{\tilde{\xi}}\mathtt{q} = \tilde{c}\,\mathtt{r}_k(\mathtt{q}), \tag{1.7}$$

with some function $\tilde{c} = \tilde{c}(\tilde{\xi})$, satisfying

$$[\nabla_{\mathtt{q}}\lambda_k \cdot \mathtt{r}_k](\mathtt{q})\,\tilde{c} = \zeta_k(\mathtt{q})\,\tilde{c} = 1. \tag{1.8}$$

The latter directly follows from differentiation of (1.6) and indicates that $\mathtt{q}(\tilde{\xi})$ is a point of genuine nonlinearity of the k-characteristic family.

Finally, the above discussion shows that a self-similar solution is composed of constant states, shocks, and simple waves.

1.3. Outline of the paper. We first address the Riemann problem of gas dynamics in section 2. Here, the general structure of a solution in terms of wave curves is given, providing the basis for introducing the specific wave parts, i.e., rarefaction, shock, and composite curves. They are discussed in the context of genuinely nonlinear, nongenuinely nonlinear, and nonsmooth characteristic fields. Then, section 3 contains the construction of wave curves—our main focus. This is basically a discussion on the termination of wave curves if certain criteria are violated, for instance, admissibility criteria, and the problem of continuation. We specify 20 possible termination conditions and provide an example for each case. Section 4 describes how to map the constructed solution from phase space onto the x-t plane. This is not obvious as not every part of the wave curve induces a wave profile visible in the x-t plane. We end with a conclusion in section 5 and a collection of figures in section 6, corresponding to the previously discussed examples.

2. The Riemann problem of gas dynamics. By now, the Riemann problem is well understood for the Euler equations that model equilibrium hydrodynamics. For an overview we refer to the review article by Menikoff and Plohr [MP89].

In Lagrangian coordinates the vector $\mathtt{q}$ and the flux function $\mathtt{f}(\mathtt{q})$ are given by

$$\mathtt{q} = (\mathtt{v}, \mathtt{u}, \epsilon)^T, \qquad \mathtt{f}(\mathtt{q}) = (-\mathtt{u}, \mathtt{p}, \mathtt{pu})^T,$$

with the velocity $\mathtt{u}$ and the total energy $\epsilon = \mathtt{e} + \mathtt{u}^2/2$. This system is closed by a general EOS satisfying the relations

$$\mathtt{p} = \mathtt{p}(\mathtt{v}, \mathtt{s}), \quad \mathtt{p}_{\mathtt{v}}(\mathtt{v}, \mathtt{s}) < 0.$$

We assume that the isentrope is a strictly monotone function in order to ensure strict hyperbolicity of the underlying system of equations. By this assumption we exclude states corresponding to three-phase mixtures. In the phase diagram in the $\mathtt{p}$-$\mathtt{T}$ plane these states are characterized by the triple point. There the sound speed vanishes and therefore we lose strict hyperbolicity. According to Menikoff and Plohr [MP89, p. 82],

"however, this loss of strict hyperbolicity does not have the important consequences for the topology of wave curves in compressible fluid flow as it has for other nonstrictly-hyperbolic systems." For systems with more degenerate wave families we refer, for instance, to [Mil03, Tor03].

Furthermore we assume that the isentropes do not cross each other in the pressure-volume plane. This is guaranteed provided that the Grüneisen coefficient

$$\mathcal{G} = -\frac{\mathtt{v}}{\mathtt{T}}\,\mathtt{e}_{\mathtt{sv}}(\mathtt{v},\mathtt{s})$$

is positive.

The characteristic speeds $\lambda_k(\mathtt{q})$, $k = 1, 2, 3$, the corresponding right eigenvectors, and the speed of sound turn out to be

$$\begin{aligned}
\lambda_1 &= -\sqrt{-\mathtt{p}_{\mathtt{v}}(\mathtt{v},\mathtt{s})} = -\mathtt{c}/\mathtt{v} < \lambda_2 = 0 < \lambda_3 = +\sqrt{-\mathtt{p}_{\mathtt{v}}(\mathtt{v},\mathtt{s})} = \mathtt{c}/\mathtt{v},\\
\mathtt{r}_{1,3} &= \frac{1}{\lambda_{1,3}}\,(-1, \lambda_{1,3}, \mathtt{p} + \lambda_{1,3}\mathtt{u})^T, \quad \mathtt{r}_2 = (-\mathtt{p}_{\mathtt{e}}(\mathtt{v}\,\mathtt{e}), 0, \mathtt{p}_{\mathtt{v}}(\mathtt{v},\mathtt{e}))^T,\\
\mathtt{c}^2 &= -\mathtt{v}^2\,\mathtt{p}_{\mathtt{v}}(\mathtt{v},\mathtt{s}),
\end{aligned}$$

respectively. The properties of the corresponding characteristic k-fields are determined by

$$\zeta_{1,3} = \Gamma/\mathtt{v}, \qquad\qquad \zeta_2 = 0,$$

where Γ denotes the fundamental derivative of gas dynamics. This quantity is a measure for the curvature of the isentropes and is defined by

$$\Gamma = \frac{\mathtt{v}^3}{2\mathtt{c}^2}\,\mathtt{p}_{\mathtt{vv}}(\mathtt{v},\mathtt{s}) = -\frac{\mathtt{v}}{2}\frac{\mathtt{p}_{\mathtt{vv}}(\mathtt{v},\mathtt{s})}{\mathtt{p}_{\mathtt{v}}(\mathtt{v},\mathtt{s})};$$

cf. Figure 1. Since the term $\mathtt{p}_{\mathtt{v}}$ is assumed to be negative, the sign of Γ coincides with the convexity properties of the isentropes. Hence, for a convex EOS, i.e., $\Gamma > 0$, the nonlinear fields are genuinely nonlinear, whereas in the case of a nonconvex EOS they may locally degenerate and there exist states $\bar{\mathtt{q}}$ in phase space such that $\zeta_k(\bar{\mathtt{q}}) = 0$.

Obviously, the Euler equations have only two linearly nondegenerated wave families, namely, $k = 1, 3$. This fact is crucially exploited in the construction of the Riemann solution; see section 2.2.

Note that, according to standard assumptions of equilibrium thermodynamics, we may equivalently express the pressure also in terms of specific volume and internal energy. For an extensive discussion on complete and incomplete EOSs see again [MP89].

2.1. The general structure of a Riemann solution. As is generally known there is a one-to-one correlation between a Riemann solution considered in the space-time plane and a set of parameterized, one-dimensional (wave) curves in phase space. Hence, solving the Riemann problem in space-time can be shifted to the construction of wave curves in such a way that there is a path along these curves connecting the initial states $\mathtt{q}_l$ and $\mathtt{q}_r$. The basic idea of this construction principle is to consider curves corresponding to the k-characteristic fields, $k = 1, \ldots, m$, which are connected at points called intermediate states; i.e., these states connect a k-wave pattern with a $k+1$-wave pattern. To be more precise, we introduce the notation $\mathtt{W}_k = \mathtt{W}_k(\mathtt{q}, \xi)$ denoting the wave curve corresponding to the k-characteristic field emanating from

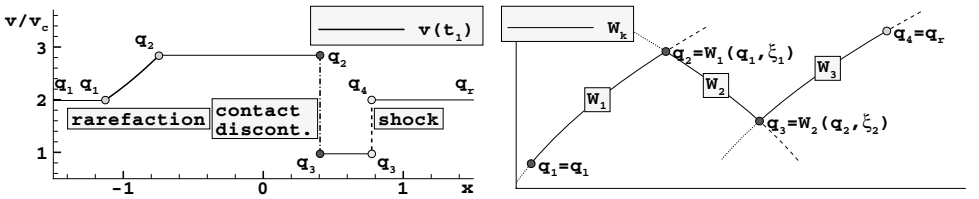
the state $\mathtt{q} \in \mathbb{R}^m$ and parameterized by $\xi \in \mathbb{R}$. These curves are composed of all states which can be connected to their origins with some specific wave type in space-time. Then we are looking for a unique set of parameter values ξ_k, $k = 1, \ldots, m$, characterizing the intermediate states $\mathtt{q}_{k+1} \in \mathbb{R}^m$ in phase space such that the conditions

$$\begin{aligned} \mathtt{q}_l &= \mathtt{W}_1(\mathtt{q}_l, 0) \equiv \mathtt{q}_1, \\ \mathtt{q}_{k+1} &= \mathtt{W}_k(\mathtt{q}_k, \xi_k), \quad k = 1, \ldots, m-1, \\ \mathtt{q}_r &= \mathtt{W}_m(\mathtt{q}_m, \xi_m) \equiv \mathtt{q}_{m+1}, \end{aligned}$$

or, equivalently,

$$\mathtt{W}_m(\mathtt{W}_{m-1}(\ldots(\mathtt{W}_1(\mathtt{q}_l, \xi_1), \xi_2), \ldots, \xi_{m-2}), \xi_{m-1}), \xi_m) = \mathtt{q}_r, \tag{2.1}$$

hold. Then the existence of a solution of the Riemann problem is tantamount to the existence of a set of parameters. [illegible] implies that there is exactly [illegible] parameter set. [illegible]



(a) Specific volume $\mathtt{v}(\cdot, t)$. (b) Wave curves.

FIG. 2. *Riemann solution in time-space and in phase space.*

The general procedure is illustrated by Figure 2 showing a classical solution of a Riemann problem for the Euler equation corresponding to a convex EOS.

The wave curve intersection in phase space is sketched in Figure 2(b) and the corresponding wave profiles in the x-t plane are shown in Figure 2(a). There are three elementary waves: (i) a simple wave connecting the states $\mathtt{q}_1 = \mathtt{q}_l$ and $\mathtt{q}_2$; (ii) a contact discontinuity from $\mathtt{q}_2$ to $\mathtt{q}_3$; and (iii) a shock between $\mathtt{q}_3$ and $\mathtt{q}_4 = \mathtt{q}_r$. The parameterized wave curves $\mathtt{W}_k = \mathtt{W}_k(\mathtt{q}_k, \xi)$ for $k \in \{1, 2, 3\}$ in this situation are sketched in Figure 2(b). For instance, one branch of wave curve $\mathtt{W}_1$ consists of all states which can be connected to $\mathtt{q}_1$ by a simple wave. As the first wave profile of the solution is a simple wave, the state $\mathtt{q}_2$ must be contained in the rarefaction branch of $\mathtt{W}_1$. Regarding this example we have

$$\mathtt{q}_1 = \mathtt{q}_l, \qquad \mathtt{q}_2 = \mathtt{W}_1(\mathtt{q}_1, \xi_1), \qquad \mathtt{q}_3 = \mathtt{W}_2(\mathtt{q}_2, \xi_2), \qquad \mathtt{q}_4 = \mathtt{W}_3(\mathtt{q}_3, \xi_3) = \mathtt{q}_r,$$

i.e., each intermediate state ($k = 2, 3$) and the right state are elements of the previous wave curve and ξ_k denotes the respective curve parameter. Note that states may coincide ($\mathtt{q}_k = \mathtt{q}_{k+1}$) and that this results in a wave that cannot be observed in the x-t plane.

Considered in the x-t space, a wave profile is just a composition of elementary wave profiles, i.e., a sequence of separated elementary waves (as in the example above), or sonic shocks adjoining simple waves (composite waves), or split waves when wave speeds are such that waves separate in the x-t space. Note that without knowledge of the [illegible] the latter split wave patterns cannot be distinguished from separated

elementary waves. Note also that the composition of a wave curve can be much more complex than the wave profile in x-t.

The construction of the wave curve $\mathtt{W}_k$ is subject to the properties of the k-characteristic fields. If it is linearly degenerated, the wave curve $\mathtt{W}_k$ is composed of one wave type only, namely, contact discontinuities. Otherwise, if it is nonlinear, the wave curve may be composed of several wave types which we here call [illegible]. Each wave part is a shock wave part, a simple wave part, or a composite wave part in the sense of Liu. For instance, a sonic shock wave profile with an adjacent continuous simple wave in x-t corresponds to a simple wave part with a following composite wave part embedded in the whole wave curve. The composition of wave curves out of wave parts and their corresponding wave profile in the x-t space is the main idea described in the following.

To account for the different parts of the wave curve $\mathtt{W}_k$ we first split it into two [illegible] $\mathtt{W}_{k+}$ and $\mathtt{W}_{k-}$ corresponding to positive and negative parameter direction relative to the origin of the wave curve. Then each wave branch is composed of [illegible] $\mathtt{W}_{k\pm}^{[j]}$ that might be an elementary wave or an elementary wave part corresponding to a composite wave. Then the wave curve can be written as the composition

$$\mathtt{W}_k = \mathtt{W}_{k+}^{[j]} \cup \mathtt{W}_{k-}^{[j]} = \{\mathtt{W}_{k+}^{[j]} \mid j \in J_{k+}\} \cup \{\mathtt{W}_{k-}^{[j]} \mid j \in J_{k-}\}.$$

Here the sets $J_{k\pm} = \{1, \dots, j_{k\pm,\max}\}$ denote index sets according to the number of involved wave parts of each branch. The associated intermediate states are referred to by $\mathtt{q}_{k\pm}^{[j]}$ and should satisfy

$$\mathtt{q}_{k\pm}^{[j]} = \mathtt{W}_{k\pm}^{[j]}(\mathtt{q}_{k\pm}, \xi_{k\pm}^{[j]}) = \mathtt{W}_{k\pm}(\mathtt{q}_{k\pm}, \xi_{k\pm}^{[j]}) \tag{2.2}$$

with appropriate parameters $\xi_{k\pm}^{[j]}$. In case of the Euler equations this parameterization is given by the pressure but it is not restricted to that.

The imposed equality concerning the right-hand side expression in (2.2) synchronizes the parameterization of $\mathtt{W}_{k\pm}$ and $\mathtt{W}_{k\pm}^{[j]}$. The various wave notions are sketched in Figure 3.

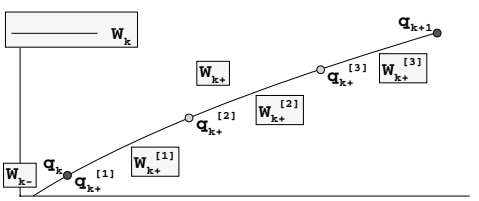
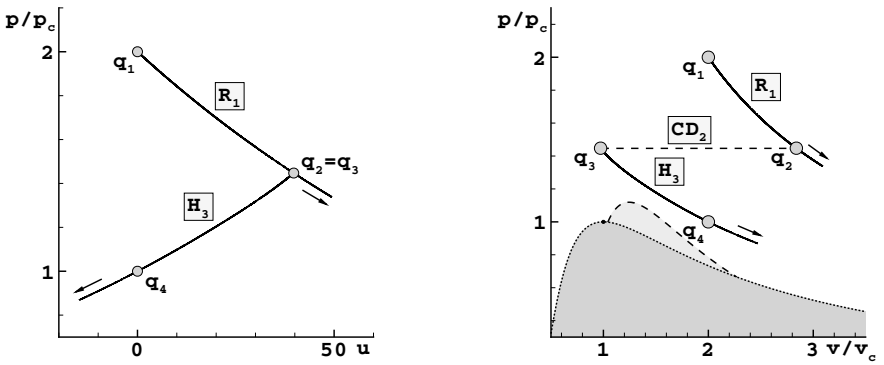


FIG. 3. *Wave curves, branches, and parts.*

The monotonicity properties of the characteristic speed or shock speed along the wave curves is important for their composition. If we speak of wave speed along some wave curve, wave branch, or wave part, we always address the speed corresponding to the respective wave type.

2.2. Existence and uniqueness of solutions. As is well known, the 2-characteristic family of the Euler equations is linearly degenerated and the 2-wave is always a contact discontinuity. The Rankine–Hugoniot condition implies that the pressure $\mathtt{p}$ and the velocity $\mathtt{u}$ have to be constant along the curve $\mathtt{W}_2$; i.e., the pressure and

the velocity do not jump on contact discontinuities. Hence, projecting the wave curves onto the p-u plane reveals all information concerning existence and uniqueness of a Riemann solution, since existence means the intersection of the 1- and 3-wave curves and uniqueness means that there is exactly one intersection point. Figures 4(a) and 4(b) sketch wave curves projected onto the p-u and p-v planes, taken from the example shown in Figure 2. Here the solution consists of a 1-rarefaction, a 2-contact discontinuity, and a 3-shock.



(a) p-u-plane. (b) p-v-plane.

FIG. 4. *Wave curve projection onto* p-u *and* p-v *planes.*

In general the question of existence can be answered by looking at the asymptotic properties of the EOS, while uniqueness of a solution depends on the monotonicity properties of the wave curves in the p-u plane. Furthermore, the transversality of the wave curves is required to solve the Riemann problem. In [MP89] the authors discuss in greater detail asymptotic and monotonicity properties while taking into account regions where $\Gamma < 0$ and where phase transition occurs. They concluded that a solution for any Riemann problem exists and that it has standard structure, except that the 1- and 3-waves may be composite or split waves.

2.3. Wave curve structure. The composition of the wave curves is strongly correlated to the genuine nonlinearity of the characteristic fields. The different cases are subsequently listed and illustrated by Figure 5. At this time, we focus on characteristic fields which are not linearly degenerated.

(C) The "convex" case: Characteristic fields are genuinely nonlinear at all states. Then each wave curve consists of a rarefaction branch in one parameter direction and a shock branch in the other direction; i.e., each branch consists of exactly one wave part. In this situation a solution of the Riemann problem is composed of simple waves and shocks (besides contact discontinuities).

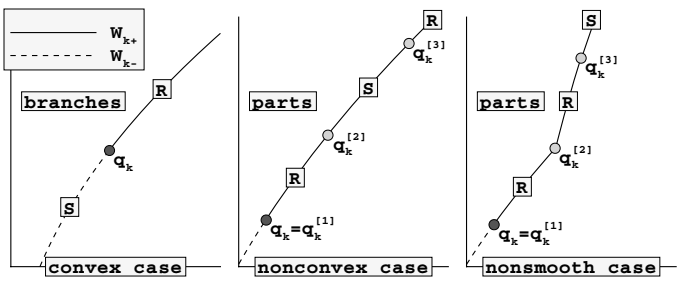
(NC) The "nonconvex" case: The genuine nonlinearity of the characteristic fields is violated at isolated states along wave curves, or precisely, along rarefaction curves. Then a wave branch may be composed of different wave parts. Figure 5 shows a typical k-wave, where a rarefaction is connected to a shock part, and the shock is connected to another rarefaction part. Notice, that the different waves are adjacent in the x-t plane; i.e., the wave speeds are equal at the states belonging to the two wave parts.

In case of the Euler equations this situation may occur for a real gas EOS such as van der Waals and a BZT-fluid in order to obtain states, where the

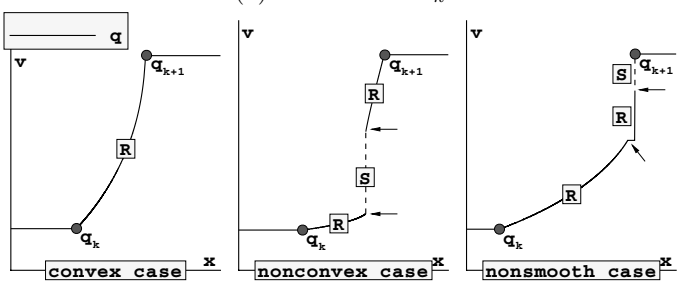
fundamental derivative Γ vanishes and the characteristic field degenerates.

(NS) The "nonsmooth" (and possibly nonconvex) case: Similar to the nonconvex case, there exist isolated states along the wave curves at which the differentiability property fails. At those states the curves are just continuous. This leads to wave curves which may also be composed of different wave parts, but due to the possible kinks the wave speeds jump between two wave parts. Concerning a solution in the x-t plane this may result in additional constant intermediate states, filling the gaps caused by the discontinuities in wave speeds. Figure 5 sketches a k-wave where a rarefaction curve is split into two parts and an adjacent shock is attached to the second rarefaction part.
Regarding the Euler equations we obtain such wave curves if we take into account phase transitions. They cause a jump in the characteristic speeds.

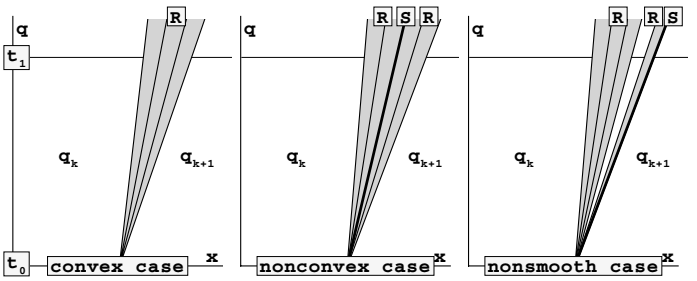
In the end all ingredients are available in order to explicitly construct a solution of the Riemann problem by composing admissible wave parts and we will obtain, in particular, the wave structures sketched in Figure 5.



(a) Wave curve $\mathtt{W}_k$.



(b) Solution of a Riemann problem, k-wave.



(c) Characteristics, k-wave.

FIG. 5. *Solution structure of Riemann problems corresponding to genuinely nonlinear, nongenuinely nonlinear, and nonsmooth nonlinear characteristic fields.*

2.4. Wave curves. In the following we summarize classical existence results involving [illegible] flux functions for the different wave curves, which can be found in [Flo02] and [Daf00]. In [CLT01] the authors prove the existence of rarefaction and shock curves for genuinely nonlinear systems and Lipschitz continuous flux functions. Their results are mainly based on the work of Filippov [Fil88] concerning ordinary differential equations with discontinuous coefficients and results from the theory of nonsmooth analysis given by Clarke [Cla83].

2.4.1. Simple waves. We have discussed before that for differentiable, self-similar solutions $\mathtt{q} = \mathtt{q}(\tilde{\xi})$, with $\tilde{\xi} = x/t$, the relations (1.7) and (1.8) hold. Now, we fix one genuinely nonlinear state $\mathtt{q}_0 \in V$, i.e., $\zeta_k(\mathtt{q}_0) \neq 0$, and consider the ordinary differential equation in the [illegible] $\mathtt{q}$

$$\mathtt{q}_{\tilde{\xi}} = \zeta_k(\mathtt{q})^{-1}\, \mathtt{r}_k(\mathtt{q}), \quad \mathtt{q}(\tilde{\xi}_0) = \mathtt{q}_0, \quad \tilde{\xi}_0 = \lambda_k(\mathtt{q}_0), \tag{2.3}$$

for one fixed but arbitrary $k \in \{1, \dots, m\}$. Then there exists a smooth solution $\mathtt{q}(\tilde{\xi})$ as long as the characteristic field is genuinely nonlinear (along this curve). Furthermore, integration of $\mathtt{d}_{\tilde{\xi}}\lambda_k$ together with the initial conditions yields $\lambda_k(\mathtt{q}(\tilde{\xi})) = \tilde{\xi}$. Note that the singularity in the term $\zeta_k(\mathtt{q})^{-1}$ for points of linear degeneracy stems from the special parameter choice of $\tilde{\xi}$.

Conversely, there exists a unique k-rarefaction wave curve $\mathtt{q}_k^{\mathtt{R}}(\xi)$ through $\mathtt{q}_0$, defined by

$$\mathtt{d}_\xi \mathtt{q}_k^{\mathtt{R}} = c\, \mathtt{r}_k(\mathtt{q}_k^{\mathtt{R}}), \quad \mathtt{q}_k^{\mathtt{R}}(\xi_0) = \mathtt{q}_0,$$

where $c \neq 0$ is an arbitrary Lipschitz continuous function. As long as the k-characteristic field remains genuinely nonlinear along $\mathtt{q}_k^{\mathtt{R}}$, the curve also solves (2.3) with the parameterization $\tilde{\xi}(\xi) = \lambda_k(\mathtt{q}_k^{\mathtt{R}}(\xi))$. For later use, we introduce the slightly more complicated notion $\mathtt{q}_k^{\mathtt{R}}(\xi, \mathtt{q}_0)$, when we need to emphasize the origin $\mathtt{q}_0$ of the wave curve.

A k-simple wave, connecting two states $\mathtt{q}_0$ and $\mathtt{q}_1 = \mathtt{q}_k^{\mathtt{R}}(\xi_1, \mathtt{q}_0)$ with a fixed parameter ξ_1, is then given by

$$\mathtt{q}(t,x) = \begin{cases} \mathtt{q}_0, & x \leq t\,\lambda_k(\mathtt{q}_0), \\ \mathtt{q}_k^{\mathtt{R}}(\xi(\tilde{\xi}), \mathtt{q}_0), & t\,\lambda_k(\mathtt{q}_0) < x < t\,\lambda_k(\mathtt{q}_1), \\ \mathtt{q}_1, & x \leq t\,\lambda_k(\mathtt{q}_1), \end{cases}$$

where the curve $\mathtt{q}_k^{\mathtt{R}}$ and the function $\xi(\tilde{\xi}) = \xi(x/t)$ are understood in the sense above. Moreover, if we have $\zeta(\mathtt{q}_0) \gtrless 0$ and choose $c > 0$, then $\lambda_k(\mathtt{q}_k^{\mathtt{R}}(\xi)) \gtrless \lambda_k(\mathtt{q}_0)$ locally if and only if $\xi \gtrless \xi_0$.

2.4.2. Shock waves. In the spirit of the last section, we fix one state $\mathtt{q}_0 \in V$ and collect all states $\mathtt{q} \in V$ that may be connected to $\mathtt{q}_0$ by a weak k-shock; i.e., the Rankine–Hugoniot jump condition holds,

$$\mathtt{f}(\mathtt{q}) - \mathtt{f}(\mathtt{q}_0) = \sigma\, (\mathtt{q} - \mathtt{q}_0), \tag{2.4}$$

with speed $\sigma = \sigma(\mathtt{q}, \mathtt{q}_0)$. Here a shock is called weak if limitations on the shock strength $|\mathtt{q} - \mathtt{q}_0| < \delta$ have to be imposed, where δ depends on $\partial\mathtt{f}(\mathtt{q}_0)$, $\partial^2\mathtt{f}(\mathtt{q}_0)$, and the modulus of continuity of $\partial^2\mathtt{f}$. In the following we always consider weak shocks. The set of shocks is called the Hugoniot locus of $\mathtt{q}_0$ and it represents the union of m shock curves. For each state $\mathtt{q}_0 \in V$ there is a unique C^2-curve $\mathtt{q}_k^{\mathtt{H}}(\xi)$ in the state

space V, called the k-shock curve through $\mathtt{q}_0$, and a C^2-function $\sigma_k(\xi)$, both defined for ξ in some neighborhood of ξ_0, with the following property: A state $\mathtt{q}$ can be connected to $\mathtt{q}_0$ by a k-shock of speed σ if and only if $\mathtt{q} = \mathtt{q}_k^{\mathtt{H}}(\xi)$, $\sigma = \sigma_k(\xi)$ for some ξ. Furthermore, we have $\mathtt{q}_k^{\mathtt{H}}(\xi_0) = \mathtt{q}_0$ and $\sigma_k(\xi_0) = \lambda_k(\mathtt{q}_0)$.

Again, the extended notion $\mathtt{q}_k^{\mathtt{H}}(\xi, \mathtt{q}_0)$, $\sigma_k(\xi, \mathtt{q}_0)$ shall be employed in the future when we want to point out the origin $\mathtt{q}_0$ of this shock curve.

At a state with $\zeta(\mathtt{q}_0) \gtrless 0$, a k-shock that connects $\mathtt{q}_0$ to a state $\mathtt{q}_1 = \mathtt{q}_k^{\mathtt{H}}(\xi, \mathtt{q}_0)$ satisfies the Lax entropy condition

$$\begin{aligned} \lambda_k(\mathtt{q}_0) > \sigma > \lambda_k(\mathtt{q}_1), \\ \lambda_{k+1}(\mathtt{q}_1) > \sigma > \lambda_{k-1}(\mathtt{q}_0) \end{aligned} \tag{2.5}$$

if and only if $\xi \lessgtr \xi_0$.

2.4.3. Composite waves. So far, we have constructed rarefaction and shock curves in case of genuinely nonlinear characteristic fields. In case of a linear degenerated state we cannot necessarily continue a rarefaction curve, since the solution may fold in the x-t plane. Analogously, a shock curve violates the Lax entropy condition if the shock speed drops below the characteristic speed. One can show that in the latter case the shock curve must have crossed at least one linear degenerated state. Hence, all phenomena are related to the existence of linear degenerated states and it is natural to extend the general situation to genuinely nonlinear characteristic fields, which are linearly degenerated on an $(m-1)$-dimensional hyperplane. This was first done by Gelfand [Gel59], [Gel63], for the scalar case and was generalized to hyperbolic systems in m dimensions by Wendroff [Wen72a], [Wen72b] and by Liu [Liu74], [Liu75]. In fact, they introduce a new type of wave curve, namely, the composite wave curve, which allows continuation of a wave curve.

We consider the situation where we have [illegible], i.e., where $\zeta_k = \nabla_{\mathtt{q}} \lambda_k \cdot \mathtt{r}_k$ vanishes locally at certain points of the phase space. To be precise, we assume that there is an $(m-1)$-dimensional hypersurface $\mathcal{M} \subset V$, where $\zeta_k|_{\mathcal{M}} = 0$ and the trajectories $\mathtt{q}_k^{\mathtt{R}}$ intersect $\mathcal{M}$ only at isolated points; i.e., $\mathtt{d}_\xi^2 \lambda_k(\mathtt{q}_k^{\mathtt{R}}(\xi)) \neq 0$ for all $\mathtt{q} \in \mathcal{M}$. Then we can continue a rarefaction curve if we consider the composite locus consisting of all roots $\mathtt{q} \in \mathcal{B}(\bar{\mathtt{q}})$ of

$$\Phi(\mathtt{q}, \xi^{\mathtt{R}}) = \mathtt{f}(\mathtt{q}) - \mathtt{f}(\mathtt{q}_k^{\mathtt{R}}(\xi^{\mathtt{R}})) - \lambda_k(\mathtt{q}_k^{\mathtt{R}}(\xi^{\mathtt{R}}))\,(\mathtt{q} - \mathtt{q}_k^{\mathtt{R}}(\xi^{\mathtt{R}})), \tag{2.6}$$

where $\mathtt{q}_k^{\mathtt{R}} = \mathtt{q}_k^{\mathtt{R}}(\xi^{\mathtt{R}}, \mathtt{q}_0)$ is a rarefaction curve, which degenerates at $\bar{\mathtt{q}} = \mathtt{q}_k^{\mathtt{R}}(\bar{\xi}, \mathtt{q}_0) \in \mathcal{M}$, i.e., $\zeta(\bar{\mathtt{q}}) = 0$.

At this stage it is not clear, however, that the composite locus is a [illegible]. It simply collects all states near the degeneration point that can be connected to any state on the rarefaction curve by a k-shock with a particular shock speed. Notice that (2.6) is actually the Rankine–Hugoniot condition where the shock speed equals the characteristic speed (sonic shock) and that, in particular, the composite locus contains the trivial solution $\mathtt{q} = \mathtt{q}_k^{\mathtt{R}}(\xi^{\mathtt{R}})$, i.e., the rarefaction curve itself.

Liu [Liu75] constructs the composite curve starting from $\bar{\mathtt{q}}$ by collecting all [illegible] states on Hugoniot curves, starting at states on the associated rarefaction curve. This way, the rarefaction curve is not part of the composite curve, but it is not clear that Liu's construction principle results in a unique curve, which additionally continues the rarefaction curve.

For this purpose the authors show in [MV05] the existence of a parameterizable composite curve $\mathtt{q}_k^{\mathtt{C}}$ in a local neighborhood of $\bar{\mathtt{q}}$ as being part of the composite locus. In particular, if one locally approaches $\bar{\mathtt{q}}$ on the rarefaction curve, then

the corresponding state on the composite curve approaches $\bar{\mathtt{q}}$ as well and the wave curve can be continued. Moreover, for any shock pair $[\mathtt{q}, \mathtt{q}^*]$ with $\mathtt{q} \in \mathtt{q}_k^{\mathtt{C}}$ and the associated state $\mathtt{q}^* \in \mathtt{q}_k^{\mathtt{R}}$ the curve $\mathtt{q}_k^{\mathtt{C}}$ is tangent to $\mathtt{q}^{\mathtt{H}}(\mathtt{q}^*)$ at $\mathtt{q}$. In particular, if $\sigma(\mathtt{q}, \mathtt{q}^*) = \lambda_k(\mathtt{q}^*) = \lambda_k(\mathtt{q})$, then $\mathtt{q}_k^{\mathtt{C}}$ is tangent to $\mathtt{q}_k^{\mathtt{R}}$ at $\mathtt{q}$; cf. [Liu75].

The composite curve provides the possibility to continue a rarefaction part of a wave curve beyond a state of linear degeneracy. It collects shock states which can be connected to some associated state, commonly denoted by $\mathtt{q}^*$, from the rarefaction curve. Now two questions arise: (i) In which sense are shocks collected in a composite curve admissible? (ii) How should we proceed, if there is no further corresponding rarefaction state?

The first question is answered within the frame of the [illegible] introduced below. The second issue addresses the fact that the associated rarefaction states run toward the origin of the rarefaction curve if we follow the composite curve. There might be the situation where we reach the origin of the rarefaction curve. In this case the solution is a simple shock and the wave curve is continued with a Hugoniot curve; we will come to this case later.

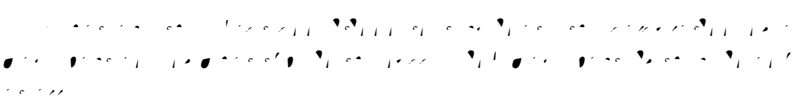
2.4.4. The Liu entropy criterion. The Liu entropy criterion was first presented in [Liu75] and concerns shock states, which may be connected by shock curves. Here a k-shock of speed $\sigma_k(\xi)$, connecting a state $\mathtt{q}_0$ to a state $\mathtt{q} = \mathtt{q}_k^{\mathtt{H}}(\xi, \mathtt{q}_0)$ for some ξ, is said to satisfy the Liu entropy condition if

$$\sigma_k(\xi) \leq \sigma_k(\hat{\xi}) \quad \forall \hat{\xi} \text{ between } \xi_0 \text{ and } \xi. \tag{2.7}$$

A k-shock connecting the states $\mathtt{q}_0$ and $\mathtt{q} = \mathtt{q}_k^{\mathtt{H}}(\xi, \mathtt{q}_0)$ and satisfying the Liu criterion (2.7) also satisfies the Lax criterion (2.5) if we suppose $\mathtt{l}^T(\mathtt{q})\,[\mathtt{d}_\xi \mathtt{q}_k^{\mathtt{H}}](\mathtt{q}) > 0$ to hold at ξ. If the k-characteristic family is genuinely nonlinear, then k-shocks satisfy the Liu criterion if and only if they satisfy the Lax criterion. Actually, the Liu condition is stronger than the Lax condition.

3. Construction of wave curves. The main goal of this chapter is to introduce a general construction principle for each wave branch that is part of an entropy solution to the Riemann problem. The idea of composing wave curves by curves of different types can be found in the work of Wendroff [Wen72b] and Liu [Liu75], and it holds for the nongenuinely nonlinear case. Here, we modify the EOS taking into account the mixture region and thus phase boundaries to the EOS and extend the construction to wave curves containing countable kinks; see [Voß05].

Since a wave branch is assembled from wave parts this basically means providing a set of conditions which successively determines the types of the wave parts and their parameter range. In particular, for a numerical implementation, the construction becomes an iterative and discrete process and the wave curve is assembled part by part and each part state by state. Seen from this point of view, the governing question reads as follows:



The answer is given in section 3.1, where we discuss all possible terminating conditions and derive the continuation of the wave branch. We will see that the monotonicity properties of the wave speeds are the crucial ingredients. We illustrate each case by an appropriate Riemann problem.

3.1. Terminating conditions for wave parts. Regarding this section the working frame is as follows:

- We always consider one specific wave branch $\mathtt{W}_k(\mathtt{q}_k, \xi)$ from its origin $\mathtt{q}_k$ to the final state $\mathtt{q}_{k+1}$, $k \in \{1, \dots, m\}$, and write $(\cdot)_k$ instead of the more complicated notation $(\cdot)_{k^\pm}$.
- This branch is composed of wave parts $\mathtt{W}_k^{[j]}$—at least of one. Refer to Figure 5 for a comparison of the possible cases.
- All wave parts are admissible, namely, shock speeds are decreasing and characteristic speeds are increasing, [illegible] the respective curve parts.
- The speeds of all wave profiles, reconstructed from the wave curve, are increasing; i.e., $\mathtt{q}(\cdot, t)$ is still a function.

Assume $\mathtt{q} = \mathtt{W}_k^{[j]}(\mathtt{q}_k, \xi)$ denotes a state belonging to the jth part of one branch of the wave curve $\mathtt{W}_k$. Then, in the first step toward a complete description of all possibilities, we have to distinguish four main cases characterizing states where the wave part starts to be no longer admissible:

$(\cdot)_{\mathtt{[m]}}$: The monotonicity `[m]` of the corresponding wave speed is violated from state $\mathtt{q}$; i.e., in case of a positive parameterized branch, the monotonicity is broken for all states $\mathtt{W}_k^{[j]}(\mathtt{q}_k, \xi + \varepsilon)$, $\varepsilon > 0$.

$(\cdot)_{\mathtt{[p]}}$: The wave part hits a phase boundary. Then $\mathtt{q}$ is assumed to be a state of the wave part in the old phase; i.e., the phase of all states with increasing parameter value is different. Subsequently, we break this case down into two (respectively, four) single cases and replace the marker `[p]` by one or two signs, related to the sign of the characteristic speed derivative along the curve and the sign of the jump in the wave speed.

$(\cdot)_{\mathtt{[W]}}$: The wave profile in the x-t plane corresponding to the state $\mathtt{q}$ is slower than the wave profile belonging to some previous wave part characterized by a smaller parameter value; i.e., the wave is [illegible] by a previous part. We write `[R]`, `[H]`, or `[C]` instead of `[W]`, depending on specific details of the overtaking process, such as the type of the previous or next wave part.

$(\cdot)_{\mathtt{[e]}}$: We reach the final state of the wave curve $\mathtt{W}_k$, i.e., $\mathtt{q} = \mathtt{q}_{k+1}$ and $\xi = \xi_{k+1}$. In this case the construction ends `[e]` and the wave curve is complete.

Some remarks concerning the conditions and the construction may be helpful. In general, if a wave part reaches a phase boundary or crosses the manifold of linear degenerated states $\mathcal{M}$, say, at parameter $\bar{\xi}$, we refer to a quantity a (along the curve) before and behind, respectively, by

$$a_- = \lim_{\xi \to \bar{\xi}_-} a(\xi), \qquad a_+ = \lim_{\xi \to \bar{\xi}_+} a(\xi).$$

In this sense, if the wave part is terminated at $\mathtt{q}_-$ and the curve is continued with a new wave part, the state $\mathtt{q}_+$ must be the origin of this succeeding wave part to ensure that the whole wave curve is continuous; see Figure 6.

Conditions $(\cdot)_{\mathtt{[p]}}$ and $(\cdot)_{\mathtt{[m]}}$ may coincide; i.e., there is a phase transition [illegible] the wave speed is not monotone. In that case we speak only of condition $(\cdot)_{\mathtt{[p]}}$. Conversely,

FIG. 6. *Discontinuous quantities along wave curves.*

if $(\cdot)_{\texttt{[m]}}$ is addressed, the wave curve is smooth and there is no phase transition.

Both conditions, $(\cdot)_{\texttt{[p]}}$ and $(\cdot)_{\texttt{[m]}}$ affect the admissibility of the wave part. In a certain sense this a local information and each state from a wave part can be checked without knowing information of previous wave parts. This is in contrast to the global condition $(\cdot)_{\texttt{[W]}}$, where the whole of the wave curve up to this state plays a role. On account of this fact one has to keep a record of previous wave parts such that the wave speed of all occurring wave profiles is increasing. To handle these speeds appropriately the idea of a stack is helpful. Such a stack works like a book pile. One can put some books on top of it ("push") or remove some books from the top ("pop"). Furthermore, only the book on the top level is available.

In our situation this reads as follows: some conditions push a wave speed on the stack, and others pop a wave speed. Checking whether the wave part is overtaken by a previous wave part means comparing the wave speed against the top level speed on the stack.

In the subsequent discussion we refer to a list of figure sets collected in section 6. Each set corresponds to a Riemann problem for the Euler equations. Here we use the van der Waals EOS with the BZT-fluid PP10 (pf-perhydrofluorene $C_{13}F_{22}$). We always choose the origin $\mathtt{q}_k$ and the last state $\mathtt{q}_{k+1}$ of a wave branch as initial conditions and show

(a) the solution $\mathtt{q}$ in space for some fixed time t,
(b) the wave speed along the wave branch,
(c) the entropy along the wave branch, and
(d) the wave curve projected onto p-v plane.

According to section 2.1, each wave part is superscripted by $(\cdot)^{[j]}$ and its type is denoted by R, H, or C. In case of composite parts and overtaking shock parts we add the index of the associated rarefaction part, respectively, the index of the second shock state (which is different from the origin of the wave part). For instance, we write $\mathtt{C}^{[4,2]}$ as an indication that rarefaction part $\mathtt{R}^{[2]}$ is associated to the composite part $\mathtt{C}^{[4]}$, which is the fourth wave part.

As we always consider only one branch of the k-characteristic family, we omit the wave index k and replace it, if necessary, by the terminating condition.

Note that in case of $\Gamma \gtrless 0$ the Bethe–Weyl theorem implies that the entropy $\mathtt{s}$ and the shock speed σ are monotone along a shock curve. The entropy always increases across a shock. If Γ may vanish, i.e., the characteristic field is nongenuinely nonlinear, the entropy is an extremum if and only if the shock speed is an extremum. This ensures that no shock states are embodied in the (Liu-admissible) shock curves at which the entropy decreases. Liu additionally shows that the entropy also increases at composite shock states; cf. [Liu75].

Because the entropy jumps are very small in the presented examples, we scale the shown curves by a factor corresponding to the entropy at state $\mathtt{q}^{[1]}$. In most cases we consider the 1-wave curve; i.e., a shock runs to the left and the entropy increases from state $\mathtt{q}^{[i]}$ to $\mathtt{q}^{[i+1]}$. If $k = 3$, then a shock runs to the right and the shock state $\mathtt{q}^{[i]}$ the shock is $\mathtt{q}^{[i+1]}$. Thus, the entropy must decrease along the wave curve; cf. case $\mathtt{H}_{\texttt{[+]}}$, Figure 17.

We remark that the observation of positive entropy increments across shocks is not surprising in the case of smooth flux functions. However, the construction principle reflects this property even in the occurrence of phase transitions.

3.2. Rarefaction curve parts. A rarefaction part is constructed as long as the characteristic speed is increasing and the wave part is smooth.

$\mathtt{R}_{\mathtt{[e]}}$: In the simplest case the wave branch is composed of exactly one wave part. Hence we terminate the rarefaction part $\mathtt{R}^{[1]}$ when it reaches the right state $\mathtt{q}_r = \mathtt{q}^{[2]}$, as shown in Figure 7 (see the appendix). In particular, no succeeding wave part is needed, as the wave curve is complete.
Notice that the bump near $\mathtt{q}^{[1]}$ in the characteristic speed diagram is caused by the nearby region of negative nonlinearity. Approaching this area with the wave curve leads to a local maximum in the wave speed; see the next case $\mathtt{R}_{\mathtt{[m]}}$.

$\mathtt{R}_{\mathtt{[m]}}$: The rarefaction curve $\mathtt{R}^{[1]}$ crosses the manifold of linear degenerated states $\mathcal{M}$ at state $\mathtt{q}^{[2]}$ and the characteristic speed reaches a maximum. In this case, the wave curve is continued with a composite curve, here $\mathtt{C}^{[2,1]}$. Figure 8 shows this situation. In particular, the speed diagrams clearly indicate the adjacent shock state $\mathtt{q}^*$ on the rarefaction part.

$\mathtt{R}_{\mathtt{[p]}}$: When the rarefaction part crosses a phase boundary the situation becomes more complicated, since we have to determine the monotonicity behavior of the characteristic speed in the new phase , the sign of the jump in the wave speed. In order to distinguish the different cases we replace our abbreviation [p] by a pair of signs: the first refers to the sign of λ' in the new phase and the second to the sign of the jump $\lambda_+ - \lambda_-$.
Obviously, there are four phase boundary situations possible.

$\mathtt{R}_{\mathtt{[++]}}$: The easiest case is given when the first rarefaction part $\mathtt{R}^{[1]}$ reaches a phase boundary, here at state $\mathtt{q}^{[2]}$, and we find that the characteristic speed λ is increasing in the new phase (first plus) and the jump in speed is positive (second plus); see Figure 9. Then we start a new rarefaction part $\mathtt{R}^{[2]}$ from $\mathtt{q}^{[2]}$ since the characteristic speed is still increasing. The solution consists of a split rarefaction fan with a constant state in between, namely, the state $\mathtt{q}^{[2]} = \mathtt{q}_-^{[2]} = \mathtt{q}_+^{[2]}$ at the phase boundary.
Technically, we have the choice to continue the current wave part $\mathtt{R}^{[1]}$ or terminate it and start a second rarefaction part in the new phase. As we will see later, e.g., in case $\mathtt{H}_{\mathtt{[R]}}$, the last characteristic speed of $\mathtt{R}^{[1]}$ must be kept in mind (push $\lambda_-^{[2]}$), and for that reason we choose the second alternative and continue the wave curve with the new rarefaction part $\mathtt{R}^{[2]}$, starting at the phase boundary. Accordingly, a simple wave part never embodies a kink state.

$\mathtt{R}_{\mathtt{[-+]}}$: We next treat the case where the characteristic speed is not increasing behind the phase boundary, but the jump in speed is positive; cf. Figure 10, state $\mathtt{q}^{[2]}$. Due to $\lambda'_+ < 0$ we cannot continue with a rarefaction part but it is possible to consider a shock curve with the boundary state as origin. Then the shock speed also decreases and thus the wave part is admissible in the sense of Liu. The jump in speed is positive and because of this the new wave profile, the shock, is faster than the rarefaction fan. Hence $\mathtt{q}(\cdot, t)$ is indeed a function. As before, we also have to push $\lambda_-^{[2]}$.
Note that the wave curve starts inside the region of negative nonlinearity and grows with increasing pressure; i.e., the simple wave is , , .

$\mathtt{R}_{\mathtt{[+-]}}$, $\mathtt{R}_{\mathtt{[--]}}$: The remaining cases $\mathtt{R}_{\mathtt{[+-]}}$ and $\mathtt{R}_{\mathtt{[--]}}$ (Figures 11 and 12) are very similar. Both have in common that the jump in characteristic speeds at state $\mathtt{q}^{[2]}$ is negative. Thus neither a rarefaction part nor a shock part starting at this (boundary) state may continue the wave curve, because the second wave profile would be slower than the last rarefaction state and $\mathtt{q}(\cdot, t)$ would become

multivalued. In order to overcome this problem we are looking for shocks whose shock speeds equal the speed of some state from the rarefaction part; i.e., we continue the wave curve by a composite part. The shock is adjacent to the rarefaction fan in the x-t plane by construction and the speed graph is continuous.

Note that the wave curves from Figures 11 and 12 differ by the fact that they leave the mixture phase at regions with different nonlinearity.

3.3. Shock and composite curve parts. Regarding the terminating conditions, shock and composite curves are very similar. This follows from the fact that, by construction, each composite state is also an element of a Hugoniot curve being tangential to the composite curve. This implies that if a condition holds at some composite state associated to the rarefaction state $\mathtt{q}^*$, an analogous condition must hold for shock curves if we start the wave branch at $\mathtt{q}^*$, and vice versa.

We assumed that shock and composite curves locally exist. They are admissible in the sense of Liu as long as the shock speed is decreasing along the curve.

$\mathtt{H}_{[e]}$: A single shock is shown in Figure 13. The shock speed is decreasing along the shock part $\mathtt{H}^{[1]}$ and thus it is Liu-admissible. Note that the shock curve crosses the region of negative nonlinearity but without causing a local minimum in shock speed. This is a qualitative difference between rarefaction and shock curves.

$\mathtt{C}_{[e]}$: The situation of a simple composite part is already treated in case $\mathtt{R}_{[m]}$; see Figure 8.

$\mathtt{H}_{[m]}$: If the shock speed equals the characteristic speed, here at $\mathtt{q}^{[2]}$, then we call it a [illegible]; see Figure 14. Thus, the solution considered in the x-t plane is composed of a shock with an adjacent rarefaction fan. In a way this is the reverse case of a composite wave. Here, the shock speed reaches a local minimum at the sonic state, i.e., $\sigma' = 0$, and continuing the shock curve $\mathtt{H}^{[1]}$ would lead to nonadmissible shock states. Therefore we continue the wave curve with a rarefaction part, that is, $\mathtt{R}^{[2]}$.

Note that the shock part may contain linearly degenerated and phase boundary states without getting nonadmissible.

$\mathtt{C}_{[m]}$: Similar to the previous case, the shock speed along a composite part may also coincide with the characteristic speed and reach a minimum. In this case the resulting shock is adjacent to a rarefaction fan on each side; cf. Figure 15, state $\mathtt{q}^{[3]}$. As in case $\mathtt{H}_{[m]}$ the composite part may contain linearly degenerated and phase boundary states.

$\mathtt{C}_{[b]}$: It may happen that the state on the associated rarefaction part of a composite part coincides with the origin of the rarefaction. Figure 16 shows an instance where we have $\mathtt{q}^* = \mathtt{q}^{[1]}$. In this situation the composite part ends and the wave curve is continued with a shock curve from $\mathtt{q}^{[3]}$. Recall that we already have $\mathtt{q}^{[3]} \in \mathtt{q}^{\mathtt{H}}(\mathtt{q}^*)$ and we just continue this shock curve. In order to indicate that this new shock part contains shock states related to $\mathtt{q}^{[1]}$, we add a second index referring to this state, i.e., we speak of the part $\mathtt{H}^{[3,1]}$.

Note, that this situation can easily be constructed if we consider the case $\mathtt{C}_{[m]}$, and start the wave curve construction on a state on the first rarefaction part $\mathtt{R}^{[1]}$ [illegible] the last associated state $\mathtt{q}^*$; this is between states $\mathtt{q}^*$ and $\mathtt{q}^{[2]}$.

$\mathtt{H}_{[+]}$, $\mathtt{H}_{[-]}$: In contrast to the characteristic speed along a simple wave the shock speed along a shock curve crossing a phase boundary is continuous. We do not terminate the shock part as long as the shock speed is decreasing, i.e.,

as long as the shock is Liu-admissible. Thus a shock part crossing a phase boundary is only terminated at the boundary if $\sigma_+ > \sigma_-$. On account of this we just have the two cases $\mathtt{H}_{[+]}$ and $\mathtt{H}_{[-]}$, referring to the monotonicity properties of λ at $\mathtt{q}_+$. In both cases we start a new wave part at state $\mathtt{q}_+$, either a rarefaction part ($\lambda'_+ > 0$), or a shock part otherwise ($\lambda'_+ < 0$). Examples are shown in Figures 17 and 18. Each second wave part creates a wave profile moving faster than the first shock due to the discontinuity in wave speed with positive jump. As before, the intermediate state is the state at the phase boundary, here $\mathtt{q}^{[2]}$.

In analogy to the cases $\mathtt{R}_{[++]}$ and $\mathtt{R}_{[-+]}$ we have to push the wave speed $\sigma_-^{[2]}$, since the jump in speed is positive.

$\mathtt{C}_{[+]}$, $\mathtt{C}_{[-]}$: We treat these conditions implicitly in case $\mathtt{H}_{[C]}$.

$\mathtt{H}_{[R]}$: This is the first case where we require the speed on the stack. In order to establish a situation where we use the top level speed we first have to push this speed at some earlier instant in the wave curve construction. The configuration is sketched in Figure 19.

We start with a rarefaction part $\mathtt{R}^{[1]}$ which reaches a phase boundary at state $\mathtt{q}^{[2]}$. At this state we determine the case $\mathtt{R}_{[-+]}$, such that we have to continue with a shock part, here $\mathtt{H}^{[2]}$. Before we do this we push speed $\lambda_-^{[2]}$ as we have explained before. The shock part crosses the phase boundary causing the kink in the speed diagram, but as the speed decreases we continue the shock curve. At some state $\mathtt{q}^{[3]}$ the shock speed equals the last characteristic speed of the rarefaction state $\mathtt{q}_-^{[2]}$, which is the top level speed on the stack. Continuing the shock curve would lead to shocks being slower than the rarefaction states, as we discussed before. Because the speed on the stack is caused by the first rarefaction part (and we chose to separate the two rarefaction parts) we can continue with a composite part $\mathtt{C}^{[3,1]}$ from $\mathtt{q}^{[3]}$, associated to $\mathtt{R}^{[1]}$. Finally, we have to remove the speed from the stack, and the case $\mathtt{H}_{[R]}$ is complete.

$\mathtt{H}_{[H]}$: This case can be handled similarly to the previous one. Figure 20 and, in particular, the speed diagram show the takeover of the second shock part $\mathtt{H}^{[2]}$ by the first part $\mathtt{H}^{[1]}$. Indeed, a short calculation shows that if $\sigma(\mathtt{q}^{[1]}, \mathtt{q}^{[2]}) = \sigma(\mathtt{q}^{[2]}, \mathtt{q}^{[3]})$, $\mathtt{q}^{[2]} \in \mathtt{q}^{\mathtt{H}}(\mathtt{q}^{[1]})$, and $\mathtt{q}^{[3]} \in \mathtt{q}^{\mathtt{H}}(\mathtt{q}^{[2]})$, we also have $\mathtt{q}^{[3]} \in \mathtt{q}^{\mathtt{H}}(\mathtt{q}^{[1]})$. In other words, the shock curves intersect at $\mathtt{q}^{[3]}$ and we continue the wave curve with a further shock part, labeled $\mathtt{H}^{[3,1]}$, which is part of the Hugoniot curve emanating from $\mathtt{q}^{[1]}$. This example shows a so-called [illegible], where the fluid condenses behind the shock.

$\mathtt{H}_{[C]}$, $\mathtt{C}_{[+]}$, $\mathtt{C}_{[-]}$: The final situation concerning shock curves is the takeover of a shock part by a composite part; cf. Figure 21. The given example is more complicated by nature as we need to establish (i) a configuration with a discontinuity in wave speed in order to push the speed, (ii) a rarefaction part in order to construct the composite part, and (iii) a shock curve which will be overtaken.

This case possibly provides the most complicated wave curve in this setting, as it incorporates two composite parts. We start again in the liquid phase with the rarefaction part $\mathtt{R}^{[1]}$ up to the phase boundary at $\mathtt{q}^{[2]}$ where we establish the second rarefaction part $\mathtt{R}^{[2]}$. The characteristic speed $\lambda_-^{[2]}$ is pushed. As part $\mathtt{R}^{[2]}$ reaches the saturated vapor boundary at state $\mathtt{q}^{[3]}$ we continue with a very small composite part $\mathtt{C}^{[3,2]}$. This part ends due to reaching the origin $\mathtt{q}^{[2]} = \mathtt{q}^*(\mathtt{q}^{[4]})$ of the associated rarefaction part $\mathtt{R}^{[2]}$ and the shock curve $\mathtt{H}^{[4,2]}$

follows. Then the shock speed equals the pushed speed at $\mathtt{q}^{[5]}$ and the shock curve is overtaken by a second composite part $\mathtt{C}^{[5,1]}$ corresponding to the first rarefaction part $\mathtt{R}^{[1]}$. The shock speed along the composite part reaches a minimum at state $\mathtt{q}^{[6]}$ and we end the wave curve with the rarefaction part $\mathtt{R}^{[6]}$. In the x-t plane the solution consists of a shock adjacent to a fan on either side. Note that we have one wave profile where only the parts $\mathtt{R}^{[1]}$, $\mathtt{C}^{[5,1]}$, and $\mathtt{R}^{[6]}$ play a role.

$\mathtt{C}_{[\mathtt{H}]}$, $\mathtt{C}_{[\mathtt{C}]}$: These situations are similar to the ones treated in $\mathtt{H}_{[\mathtt{H}]}$ and $\mathtt{H}_{[\mathtt{C}]}$, and we just have to start with a rarefaction and a composite part instead of a shock part.

The case $\mathtt{C}_{[\mathtt{H}]}$ is treated in Figure 22. On first glance this situation is covered by case $\mathtt{C}_{[\mathtt{b}]}$, but if we look at the suggested continuation, i.e., a shock curve with the first state of the simple wave as origin, we observe that this new shock part would be overtaken instantaneously by the first shock part. Hence we continue the first shock part directly, here denoted by $\mathtt{H}^{[4,1]}$.

We could not construct the case $\mathtt{C}_{[\mathtt{C}]}$ for the Euler equations equipped with a cubic EOS and one for the BZT-fluids provided in [Cra91], though we do not know if it may be possible for some combination of artificial EOS and/or BZT-fluid. Note that one needs at least two rarefaction parts and two composite parts in order to obtain the , , , , configuration

$$\mathtt{R}^{[1]}, \mathtt{C}^{[2,1]}, \mathtt{R}^{[3]}, \mathtt{C}^{[4,3]}, \mathtt{C}^{[5,1]}.$$

This involves at least two regions with negative nonlinearity or one region with negative nonlinearity and an appropriate phase change such as the one involved in the case $\mathtt{R}_{[\mathtt{+-}]}$. None of the situations we came across possess either of these properties.

Table 1 summarizes the discussion above and provides all derived terminating conditions and subsequent wave part types.

4. Mapping of solutions from phase space to space-time. The previous section describes in detail how to compose a wave curve such that all requirements concerning admissible Riemann solutions are met. It remains to show in which way the function $\mathtt{q}(t, x)$, living in the x-t plane, is obtained from the wave curves. As such a solution is scale-invariant it is sufficient to provide the solution in the form $\mathtt{q}(\tilde{\xi})$ with $\tilde{\xi} = \frac{x}{t}$. Moreover, $\mathtt{q}(\tilde{\xi})$ is composed of k-wave profiles, $k = 1, \ldots, m$, where each wave profile corresponds to one wave branch of the k-characteristic family. Hence we need to consider only one k-wave branch in order to do the transformation.

We approach the underlying idea by means of an example. Assume a wave branch is given by the sequence from the example shown in Figure 19,

$$\mathtt{W}_k = \mathtt{R}^{[1]}_{[\mathtt{-+}]} \cup \mathtt{H}^{[2]}_{[\mathtt{R}]} \cup \mathtt{C}^{[3,1]}_{[\mathtt{e}]};$$

then the solution consists of a rarefaction fan with an adjacent shock. The second wave part $\mathtt{H}^{[2]}_{[\mathtt{R}]}$ does not play any role because all states between $\mathtt{q}^{[4]}$ and the last associated state $\mathtt{q}^*$ from $\mathtt{R}^{[1]}$ are bridged by the shock stemming from the composite part $\mathtt{C}^{[3,1]}$.

We equipped the composite notation with an additional index referring to the associated rarefaction part, where, in particular, the state $\mathtt{q}^*$ lives. This linkage helps us figure out the skipped and the involved wave parts, namely, index 2 and indices

TABLE 1
Terminating conditions.

Types of wave parts and conditions in effect			
R	$\lambda' > 0$, curve is smooth		
H	$\sigma' < 0$, $\sigma \neq \sigma^{[\mathtt{top}]}$		
C	$\sigma' < 0$, $\sigma \neq \sigma^{[\mathtt{top}]}$, origin of corresponding R is not reached		
Abbreviations for terminating conditions		Succ.	Remark
$\mathtt{R}_{[\mathtt{e}]}$	external (e) condition fulfilled		
$\mathtt{R}_{[\mathtt{m}]}$	monotonicity (m) of λ broken, i.e., $\lambda' = 0$	C	
$\mathtt{R}_{[++]}$	phase boundary crossed, $\lambda'_+ > 0$, $\lambda_+ > \lambda_-$	R	push λ_-
$\mathtt{R}_{[+-]}$	phase boundary crossed, $\lambda'_+ > 0$, $\lambda_+ < \lambda_-$	C	
$\mathtt{R}_{[-+]}$	phase boundary crossed, $\lambda'_+ < 0$, $\lambda_+ > \lambda_-$	H	push λ_-
$\mathtt{R}_{[--]}$	phase boundary crossed, $\lambda'_+ < 0$, $\lambda_+ < \lambda_-$	C	
$\mathtt{H}_{[\mathtt{e}]}$	external (e) condition fulfilled		
$\mathtt{H}_{[\mathtt{m}]}$	monotonicity (m) broken, i.e., $\sigma' = 0$	R	push σ_-
$\mathtt{H}_{[+]}$	phase boundary crossed, $\lambda'_+ > 0$	R	push σ_-
$\mathtt{H}_{[-]}$	phase boundary crossed, $\lambda'_+ < 0$	H	push σ_-
$\mathtt{H}_{[\mathtt{R}]}$	overtaken by R, i.e., $\sigma = \sigma^{[\mathtt{top}]}$	C	pop
$\mathtt{H}_{[\mathtt{H}]}$	overtaken by H, i.e., $\sigma = \sigma^{[\mathtt{top}]}$	H	pop
$\mathtt{H}_{[\mathtt{C}]}$	overtaken by C, i.e., $\sigma = \sigma^{[\mathtt{top}]}$	C	pop
$\mathtt{C}_{[\mathtt{e}]}$	external (e) condition fulfilled		
$\mathtt{C}_{[\mathtt{m}]}$	monotonicity (m) broken, i.e., $\sigma' = 0$	R	push σ_-
$\mathtt{C}_{[\mathtt{b}]}$	begin (b) of corresponding R reached	H	
$\mathtt{C}_{[+]}$	phase boundary crossed, $\lambda'_+ > 0$	R	push σ_-
$\mathtt{C}_{[-]}$	phase boundary crossed, $\lambda'_+ < 0$	H	push σ_-
$\mathtt{C}_{[\mathtt{H}]}$	overtaken by H, i.e., $\sigma = \sigma^{[\mathtt{top}]}$	H	pop
$\mathtt{C}_{[\mathtt{C}]}$	overtaken by C, i.e., $\sigma = \sigma^{[\mathtt{top}]}$	C	pop

1 and 3, respectively. In short, the overtaking of waves always means skipping wave parts in between!

If we consider the wave construction backwards, i.e., from the fastest waves to the slowest ones, which is the parameter range $\tilde{\xi} = +\infty \to -\infty$, we can take advantage of this linkage information. The fastest wave profile is caused by the last involved wave part $\mathtt{C}^{[3,1]}$. As shock and composite curves collect possible shocks, only the last shock state, or last shock pair $[\mathtt{q}, \mathtt{q}_0]$, respectively, is important. In this case it is the shock between $\mathtt{q}^{[4]}$ and the associated state $\mathtt{q}^*$. We assign a (single valued) parameter range to this wave part, namely, the shock speed

$$\Xi^{[3]} = \{\sigma(\mathtt{q}^{[3]}, \mathtt{q}^*)\}.$$

The definition of a simple wave relates the states along a rarefaction part to their wave speed and thus each rarefaction part belongs to some nonsingular range in the variable $\tilde{\xi}$. As we skip wave part 2 due to the linkage of parts 3 and 1, we continue with part $\mathtt{R}^{[1]}$. This part connects states $\mathtt{q}^{[1]}$ and $\mathtt{q}^{[2]}$, but all occurring states in the fan must be slower than the composite shock between $\mathtt{q}^{[4]}$ and $\mathtt{q}^*$. Hence we take into account only the range of the rarefaction part between $\mathtt{q}^*$ and $\mathtt{q}^{[1]}$, i.e.,

$$\Xi^{[1]} = [\tilde{\xi}^{[1]}, \tilde{\xi}^*].$$

At this point we define two ranges $\Xi^{[i]}$, each corresponding to some wave part. In the nonconvex but smooth situation of the Riemann problem the union of ranges

$$\Xi = \bigcup_{j=1,3} \Xi^{[j]}$$

is a continuous interval, since the wave speed along the whole wave branch is continuous. The nonsmooth case causes jumps in wave speed and as a consequence we obtain gaps in Ξ. But as the wave curve is still continuous, the corresponding state on each side of this gap is the same and we have found the constant intermediate state to fill in the blank between the split wave profiles.

In a nutshell, the general strategy to obtain a function $\mathtt{q}(\tilde{\xi})$ can be described as follows:

- Collect the indices j of all involved wave parts $\mathtt{W}^{[j]}$ in a set J; let $\#J$ denote the number of elements in J. If we have a nonempty wave curve the set J contains at least one index.
- For this purpose iterate backward and use the linkage information to skip single indices.
- Define the corresponding parameter ranges $\Xi^{[j]}$ and add

$$\Xi^{[0]} = \{-\infty\} \quad \text{and} \quad \Xi^{[\#J+1]} = \{+\infty\}.$$

Then, skipping some details, the main point is to decide whether a given parameter $\tilde{\xi}$ lies [illegible] some $\Xi^{[j]}$, or [illegible] some ranges $\Xi^{[j]}$ and $\Xi^{[j+1]}$. In the first case we have a rarefaction part and the value is obtained by means of its definition. In the other case we use the corresponding boundary states of $\Xi^{[j]}$ and $\Xi^{[j+1]}$ (which should be equal) in order to obtain an intermediate state.

5. Conclusion. In this paper we are concerned with the construction of an entropy solution (in the sense of Liu) to the Riemann problem of gas dynamics. Here we take into account EOSs exhibiting regions of negative nonlinearity and phase transition assuming thermodynamical equilibrium. This setting applies to BZT-fluids, which play an important role in the design of turbomachinery. The wave curves corresponding to the nonlinear fields consist of various wave parts which are connected according to a set of rules, based on admissibility criteria for shock and rarefaction curves. The construction principle extends the methods used in the smooth, nongenuinely nonlinear case, first given by Wendroff and Liu. We present an algorithm for computing these wave curves, but we claim no convergence.

We emphasize that the construction can be applied to any hyperbolic system with properties comparable to the Euler equations regarding the characteristic fields and the existence of solutions. In particular, it works in the scalar case, where it yields the Riemann solution based on the convex hull of the flux function.

In addition to the theoretical work, a software library has been implemented containing all necessary modules to provide

- the k-wave curves in phase space;
- the wave profile in the x-t plane;
- various EOSs (among them are the well-known van der Waals, the Redlich–Kwong, and the Peng–Robinson EOSs; all cubic EOSs can be equipped with the Maxwell construction modeling the mixture region and phase boundaries) and
- the BZT-fluids.

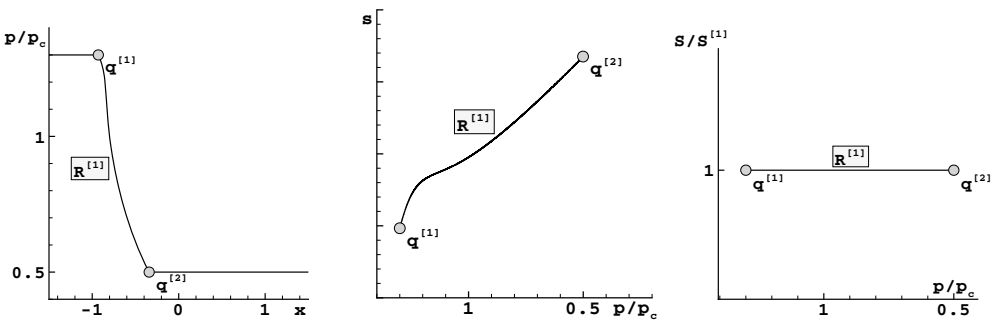
The library can be used within a batch tool in order to generate plots like the ones we used in this paper, or it can be used as a fluid library, for instance, in the flow solver Quadflow; see [BLM04].

The exact Riemann solution has been used as a validation tool for numerical solvers. We have considered several standard schemes, e.g., the Lax–Friedrich and Lax–Wendroff schemes, the Roe scheme, a second order ENO scheme, the HLLC

and HLLE schemes, or the WAF scheme with MinBee and SuperBee limiters (see [Toro99]). The computations have been performed for various spatial resolutions and CFL numbers.

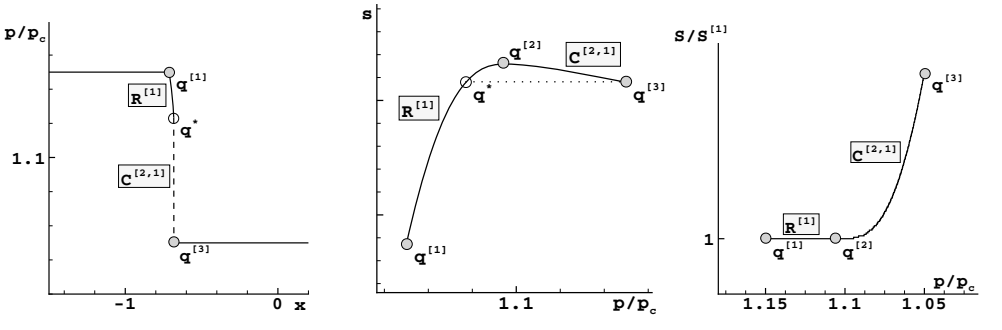
Finally, we remark that our construction is based on the entropy concept of Liu. By this the existence of so-called nonclassical shocks is excluded; see, for instance, [HL00], [Flo02], and the references therein. In the future it would be interesting to extend our construction taking into account these effects.

Appendix.



(a) Solution $\mathtt{q}(\cdot,t)$. (b) Wave speed along $\mathtt{W}_k$. (c) Entropy along $\mathtt{W}_k$.

FIG. 7. *Case* $\mathtt{R}_{[e]}$. *(Attention: Pressure scale inverted in* (b), (c)*.)*



(a) Solution $\mathtt{q}(\cdot,t)$. (b) Wave speed along $\mathtt{W}_k$. (c) Entropy along $\mathtt{W}_k$.

FIG. 8. *Cases* $\mathtt{R}_{[m]}$, $\mathtt{C}_{[e]}$. *(Attention: Pressure scale inverted in* (b), (c)*.)*

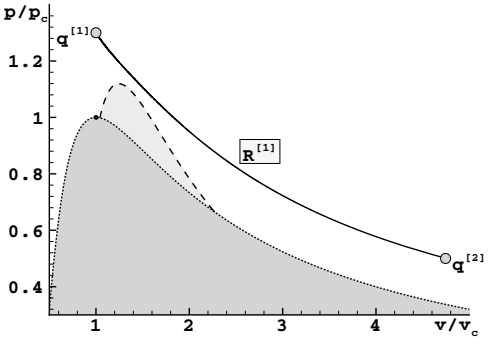


FIG. 7(d). *Wave curve* ($k = 1$).

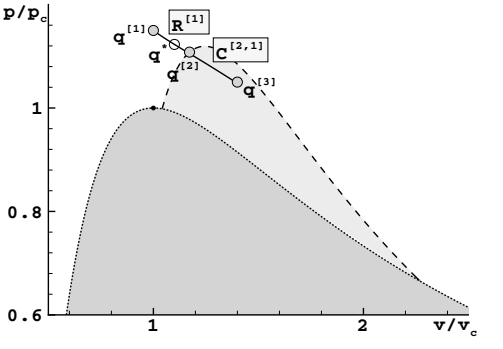
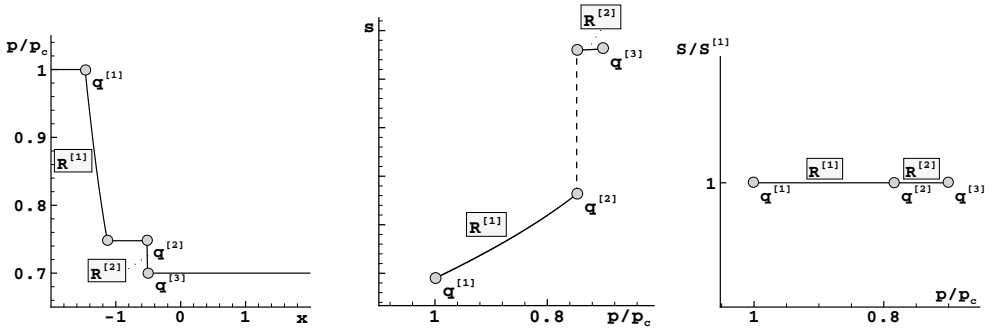
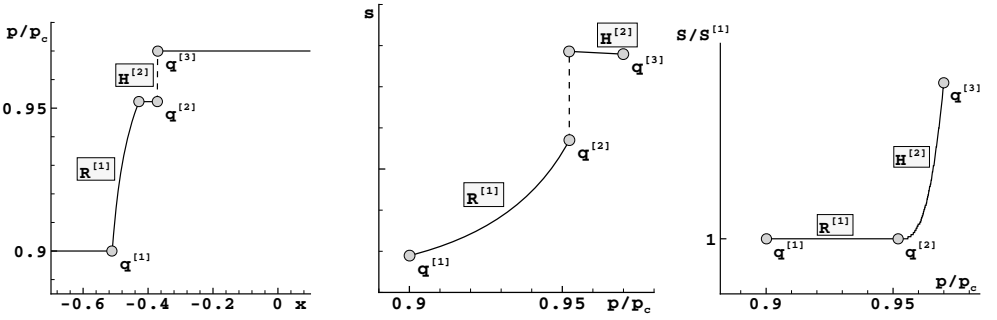


FIG. 8(d). *Wave curve* ($k = 1$).



(a) Solution $\mathtt{q}(\cdot, t)$. (b) Wave speed along $\mathtt{W}_k$. (c) Entropy along $\mathtt{W}_k$.

FIG. 9. *Case* $\mathtt{R}_{[++]}$. *(Attention: Pressure scale inverted in* (b), (c)*.)*



(a) Solution $\mathtt{q}(\cdot, t)$. (b) Wave speed along $\mathtt{W}_k$. (c) Entropy along $\mathtt{W}_k$.

FIG. 10. *Case* $\mathtt{R}_{[-+]}$.

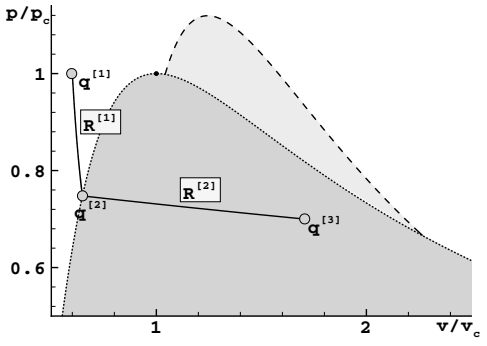


FIG. 9(d). *Wave curve* ($k = 1$).

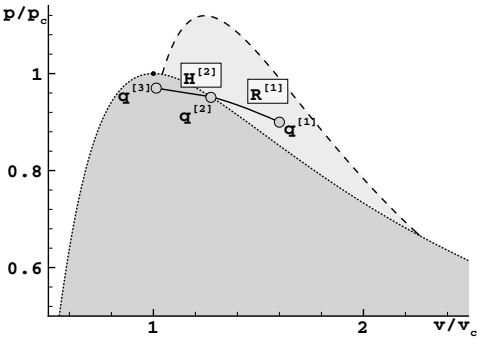
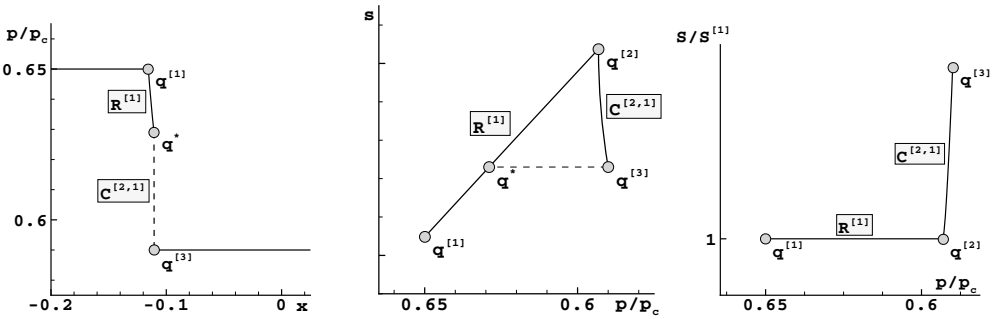
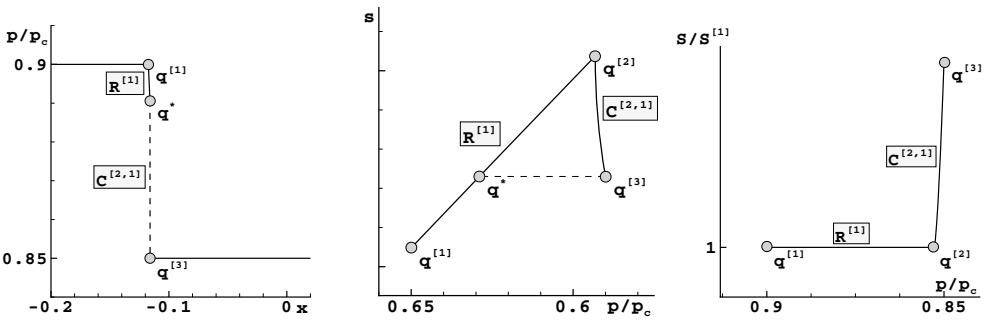


FIG. 10(d). *Wave curve* ($k = 1$).



(a) Solution $\mathtt{q}(\cdot, t)$. (b) Wave speed along $\mathtt{W}_k$. (c) Entropy along $\mathtt{W}_k$.

FIG. 11. *Case* $\mathtt{R}_{[+-]}$. *(Attention: Pressure scale inverted in* (b), (c)*.)*



(a) Solution $\mathtt{q}(\cdot, t)$. (b) Wave speed along $\mathtt{W}_k$. (c) Entropy along $\mathtt{W}_k$.

FIG. 12. *Case* $\mathtt{R}_{[--]}$. *(Attention: Pressure scale inverted in* (b), (c)*.)*

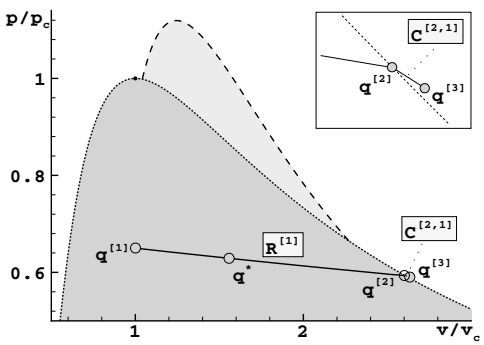


FIG. 11(d). *Wave curve* ($k = 1$).

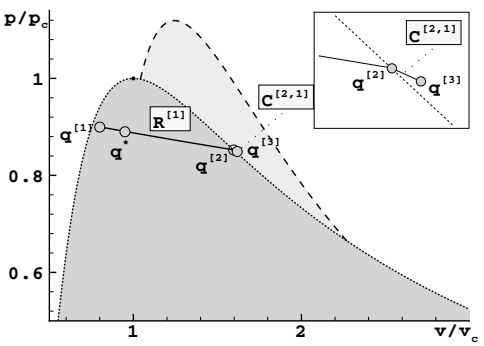
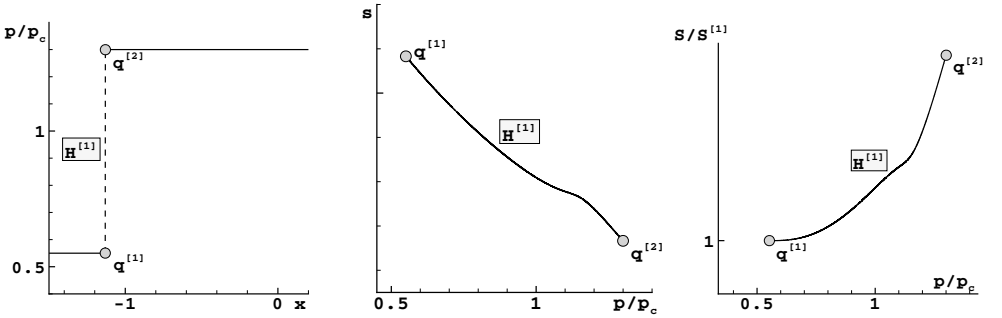
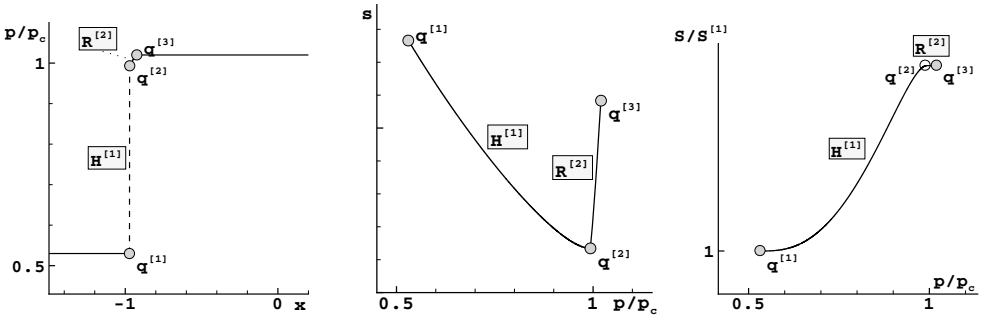


FIG. 12(d). *Wave curve* ($k = 1$).



(a) Solution $\mathtt{q}(\cdot, t)$. (b) Wave speed along $\mathtt{W}_k$. (c) Entropy along $\mathtt{W}_k$.

FIG. 13. *Case* $\mathtt{H}_{[\mathtt{e}]}$.



(a) Solution $\mathtt{q}(\cdot, t)$. (b) Wave speed along $\mathtt{W}_k$. (c) Entropy along $\mathtt{W}_k$.

FIG. 14. *Case* $\mathtt{H}_{[\mathtt{m}]}$.

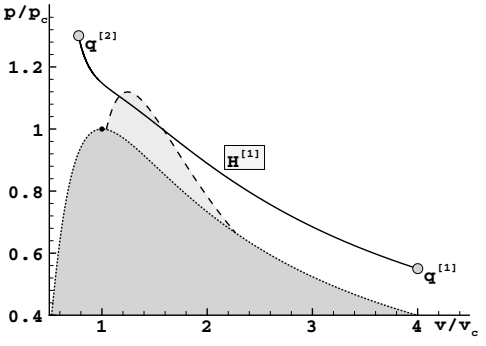


FIG. 13(d). *Wave curve* ($k = 1$).

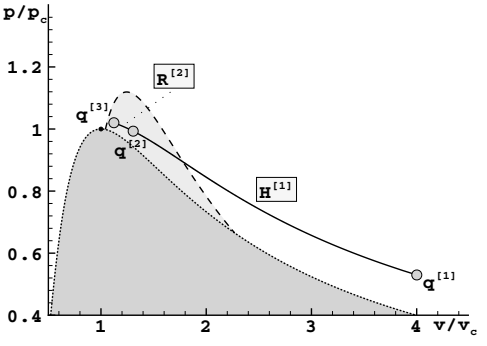
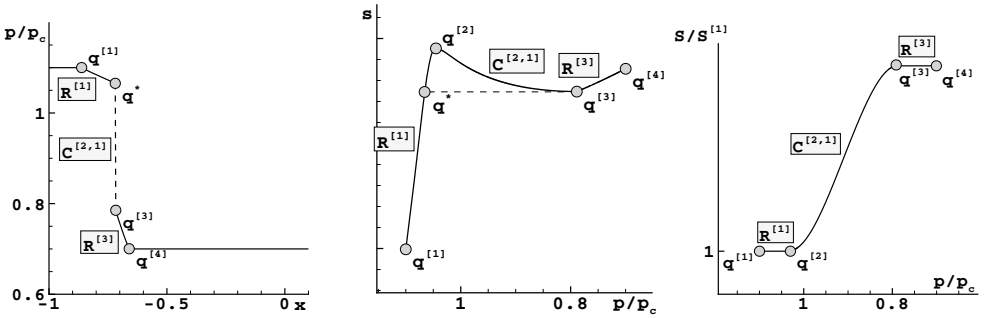
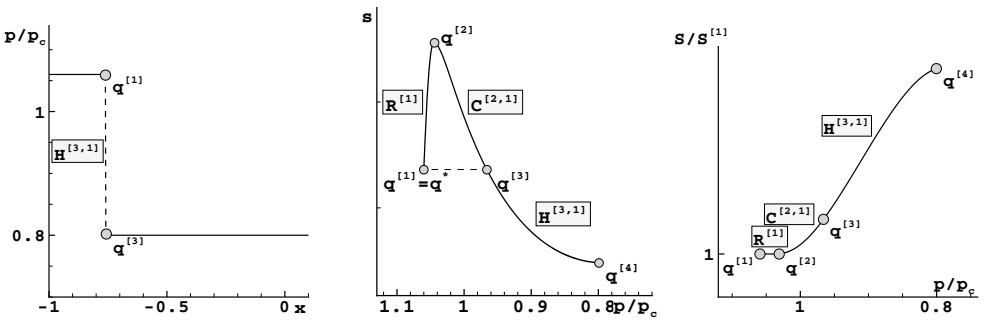


FIG. 14(d). *Wave curve* ($k = 1$).



(a) Solution $\mathtt{q}(\cdot,t)$. (b) Wave speed along $\mathtt{W}_k$. (c) Entropy along $\mathtt{W}_k$.

FIG. 15. *Case* $\mathtt{C}_{[\mathtt{m}]}$. *(Attention: Pressure scale inverted in* (b), (c)*.)*



(a) Solution $\mathtt{q}(\cdot,t)$. (b) Wave speed along $\mathtt{W}_k$. (c) Entropy along $\mathtt{W}_k$.

FIG. 16. *Case* $\mathtt{C}_{[\mathtt{b}]}$. *(Attention: Pressure scale inverted in* (b), (c)*.)*

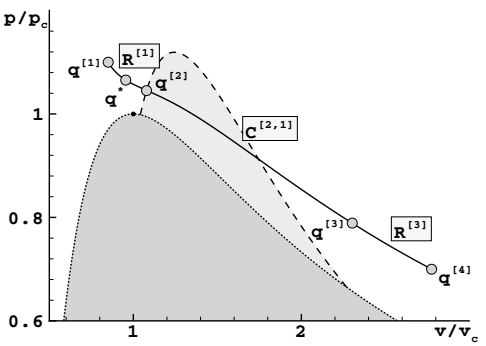


FIG. 15(d). *Wave curve* ($k = 1$).

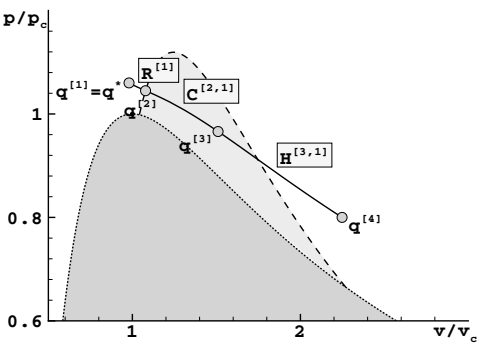
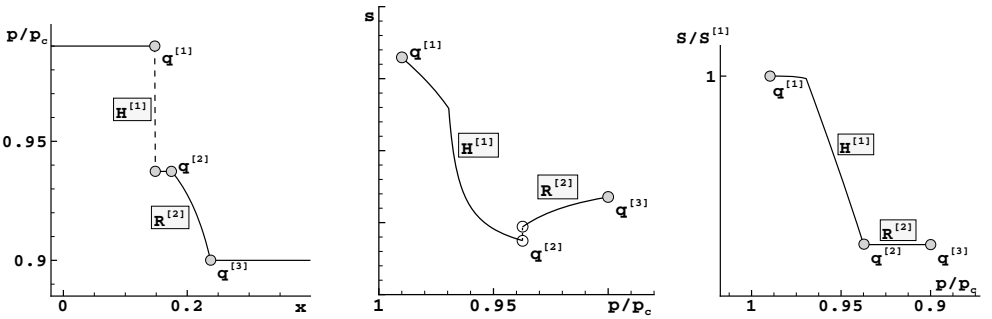
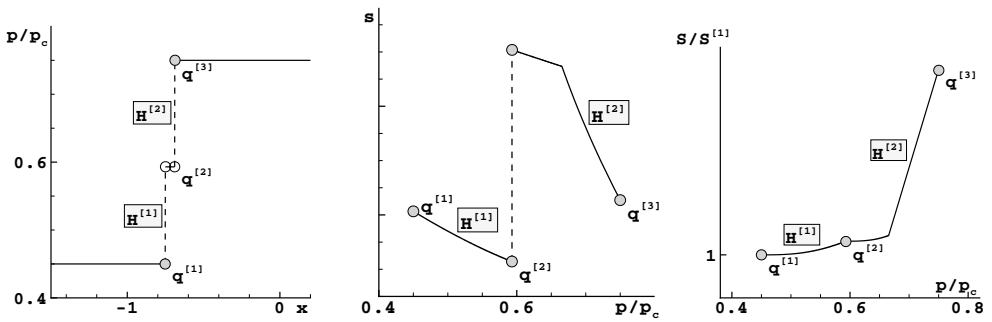


FIG. 16(d). *Wave curve* ($k = 1$).



(a) Solution $\mathtt{q}(\cdot, t)$. (b) Wave speed along $\mathtt{W}_k$. (c) Entropy along $\mathtt{W}_k$.

FIG. 17. *Case* $\mathtt{H}_{[+]}$. *(Attention: Pressure scale inverted in* (b), (c)*.)*



(a) Solution $\mathtt{q}(\cdot, t)$. (b) Wave speed along $\mathtt{W}_k$. (c) Entropy along $\mathtt{W}_k$.

FIG. 18. *Case* $\mathtt{H}_{[-]}$.

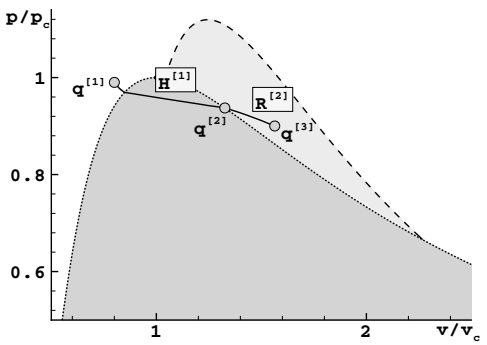


FIG. 17(d). *Wave curve* ($k = 3$).

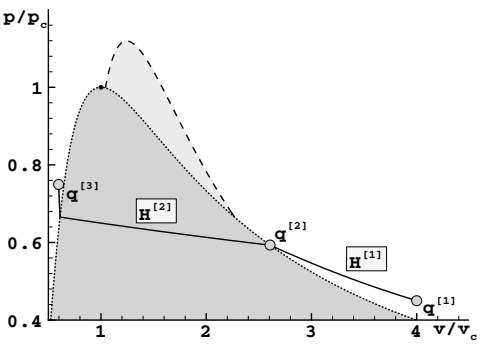
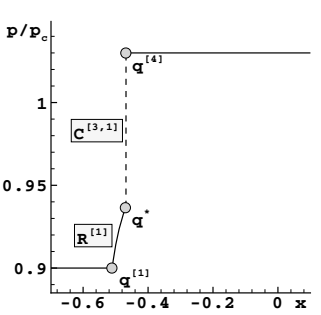
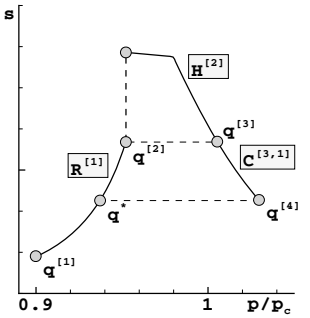


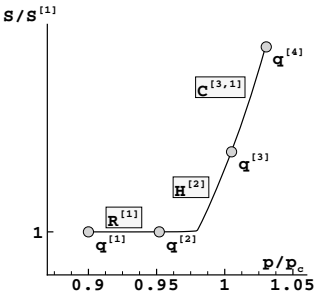
FIG. 18(d). *Wave curve* ($k = 1$).



(a) Solution $\mathtt{q}(\cdot,t)$.

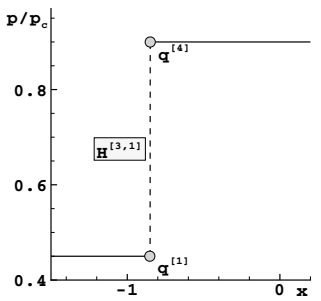


(b) Wave speed along $\mathtt{W}_k$.

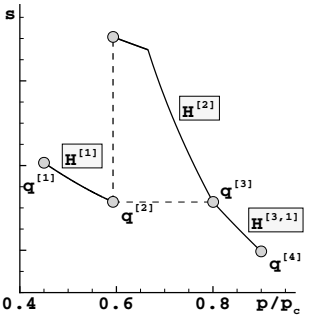


(c) Entropy along $\mathtt{W}_k$.

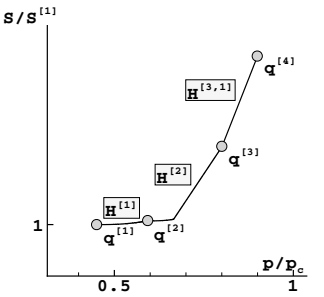
FIG. 19. *Case* $\mathtt{H}_{[\mathtt{R}]}$.



(a) Solution $\mathtt{q}(\cdot,t)$.



(b) Wave speed along $\mathtt{W}_k$.



(c) Entropy along $\mathtt{W}_k$.

FIG. 20. *Case* $\mathtt{H}_{[\mathtt{H}]}$.

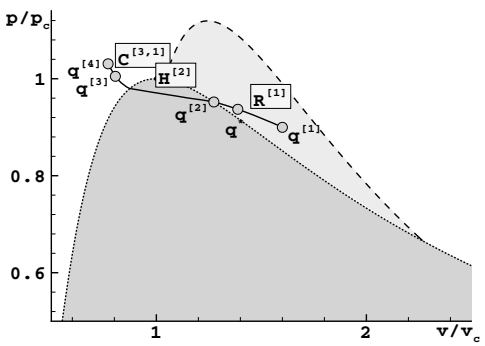


FIG. 19(d). *Wave curve* ($k = 1$).

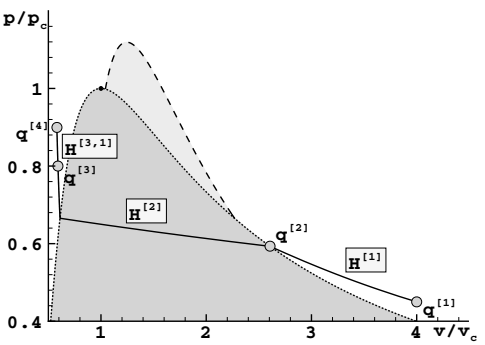
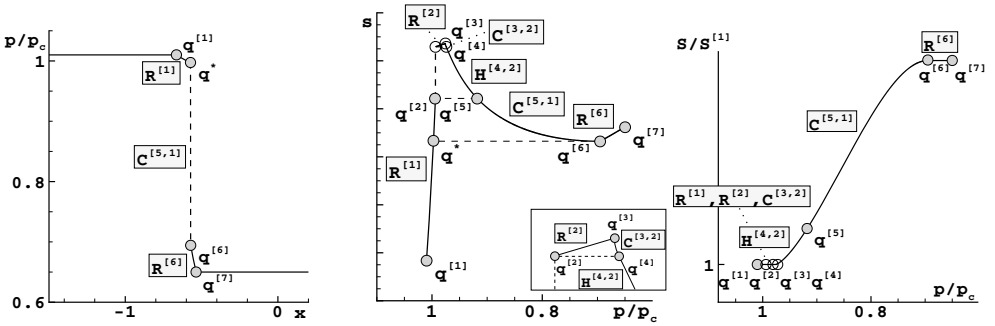
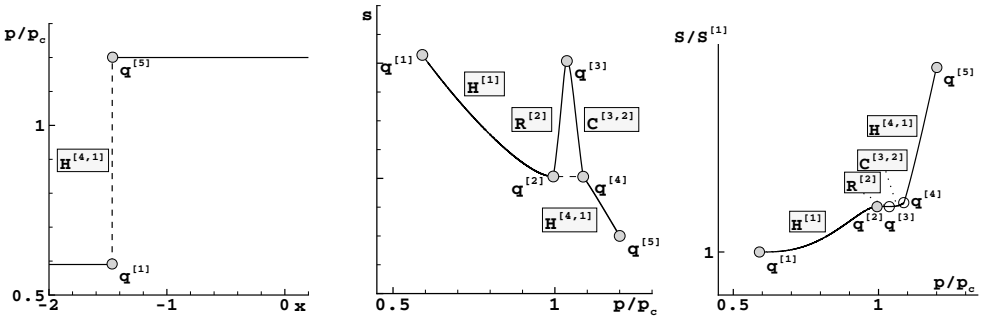


FIG. 20(d). *Wave curve* ($k = 1$).



(a) Solution $\mathtt{q}(\cdot, t)$. (b) Wave speed along $\mathtt{W}_k$. (c) Entropy along $\mathtt{W}_k$.

FIG. 21. *Attention: Pressure scale inverted in* (b), (c).



(a) Solution $\mathtt{q}(\cdot, t)$. (b) Wave speed along $\mathtt{W}_k$. (c) Entropy along $\mathtt{W}_k$.

FIG. 22.

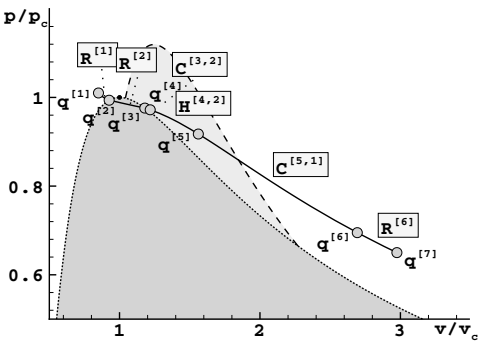


FIG. 21(d). *Wave curve ($k = 1$).*

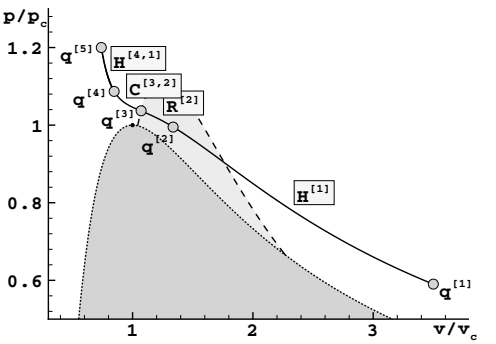


FIG. 22(d). *Wave curve ($k = 1$).*

Acknowledgment. The authors gratefully acknowledge Dr. Ralf Menikoff for many fruitful and inspiring discussions on this entire topic.

REFERENCES

[Bae88] H. D. BAEHR, *Thermodynamik*, Springer-Verlag, New York, 1988.

[BBKN83] A. A. BORISOV, AL. A. BORISOV, S. S. KUTATELADZE, AND V. E. NAKORYAKOV, *Rarefaction shock wave near the critical liquid vapor point*, J. Fluid Mech., 126 (1983), pp. 59–73.

[BLM04] F. BRAMKAMP, P. LAMBY, AND S. MÜLLER, *An adaptive multiscale finite volume solver for unsteady and steady flow computations*, J. Comput. Phys., 197 (2004), pp. 460–490.

[Cla83] F. H. Clarke, *Optimization and Nonsmooth Analysis*, John Wiley and Sons, New York, 1983.

[CLT01] J. Correia, P. G. LeFloch, and M. D. Thanh, *Hyperbolic systems of conservation laws with Lipschitz continuous flux-functions: The Riemann problem*, Bol. Soc. Bras. Mat., 32 (2001), pp. 271–301.

[Cra91] M. S. Cramer, *Nonclassical dynamics of classical waves*, in Nonlinear Waves in Real Fluids, A. Kluwick, ed., Springer-Verlag, New York, 1991, pp. 147–213.

[Daf00] C. M. Dafermos, *Hyperbolic Conservation Laws in Continuum Physics*, Springer-Verlag, New York, 2000.

[DTMS79] G. Dettleff, P. A. Thompson, G. E. A. Meier, and H.-D. Speckmann, *An experimental study of liquefaction shock waves*, J. Fluid Mech., 95 (1979), pp. 279–304.

[Fil88] A. F. Filippov, *Differential equations with discontinuous right-hand sides*, Mathematics and Its Applications (Soviet Series) 18, Kluwer, Dordrecht, The Netherlands, 1988.

[Flo02] P. G. Floch, *Hyperbolic Systems of Conservation Laws*, Birkhäuser, Basel, 2002.

[Gel59] I. Gelfand, *Some problems in the theory of quasilinear equations*, Usp. Mat. Nauk, 14 (1959), pp. 87–158 (in Russian).

[Gel63] I. Gelfand, *Some problems in the theory of quasilinear equations*, Amer. Math. Soc. Transl. (2), 29 (1963), pp. 295–381.

[HL00] B. T. Hayes and P. G. LeFloch, *Nonclassical shocks and kinetic relations: Strictly hyperbolic systems*, SIAM J. Math. Anal., 31 (2000), pp. 941–991.

[IL03] T. Iguchi and P. G. LeFloch, *Existence theory for hyperbolic systems of conservation laws with general flux-functions*, Arch. Ration. Mech. Anal., 168 (2003), pp. 165–244.

[IMPT88] E. Isaacson, D. Marchesin, B. Plohr, and B. Temple, *The Riemann problem near a hyperbolic singularity: The classification of solutions of quadratic Riemann problems* I, SIAM J. Appl. Math., 48 (1988), pp. 1009–1032.

[Liu74] T.-P. Liu, *The Riemann problem for general* 2×2 *conservation laws*, Trans. Amer. Math. Soc., 199 (1974), pp. 89–112.

[Liu75] T.-P. Liu, *The Riemann problem for general systems of conservation laws*, J. Differential Equations, 18 (1975), pp. 218–234.

[Mil03] G.H̃. Miller, *An iterative Riemann solver for systems of hyperbolic conservation laws*, J. Comput. Phys., 193 (2003), pp. 198–225.

[MP89] R. Menikoff and B. J. Plohr, *The Riemann problem for fluid flow of real materials*, Rev. Modern Phys., 61 (1989), pp. 75–130.

[MV05] S. Müller and A. Voß, *Analysis of composite wave curves for non-convex systems of conservation laws*, in Hyperbolic Problems, Theory, Numerics, Applications, Vol. II, F. Asakura, H. Aiso, S. Kawashima, A. Matsumura, S. Nishibata, K. Nishihara, eds., Yokohama Publishers, Yokohama, Japan, to appear.

[Smi79] R. G. Smith, *The Riemann problem in gas dynamics*, Trans. Amer. Math. Soc., 249 (1979), pp. 1–50.

[TCK86] P. Thompson, G. Carafano, and Y.-G. Kim, *Shock waves and phase changes in a large-heat-capacity fluid emerging from a tube*, J. Fluid. Mech., 166 (1986), pp. 57–92.

[TCM+87] P. A. Thompson, H. Chaves, G. E. A. Meier, Y.-G. Kim, and H.-D. Speckmann, *Wave splitting in a fluid of large heat capacity*, J. Fluid Mech., 185 (1987), pp. 385–414.

[Tho91] P. Thompson, *Liquid-vapor adiabatic phase changes and related phenomena*, in Nonlinear Waves in Real Fluids, A. Kluwick, ed., Springer-Verlag, New York, 1991, pp. 147–213.

[TK83] P. Thompson and Y.-G. Kim, *Direct observation of shock splitting in a vapor-liquid system*, Phys. Fluids, 26 (1983), pp. 3211–3215.

[Toro99] E. F. Toro, *Riemann Solvers and Numerical Methods for Fluid Dynamics. A Practical Introduction*, 2nd ed., Springer-Verlag, Berlin, 1999.

[Tor03] M. Torrilhon, *Uniqueness conditions for Riemann problems of ideal magnetohydrodynamics*, J. Plasma Phys., 69 (2003), pp. 253–276.

[Voß05] A. Voß, *Exact Riemann Solutions for the Euler Equations with Nonconvex and Nonsmooth Equation of State*, Ph.D. thesis, RWTH Aachen, Aachen, Germany, 2005, available online at http://darwin.bth.rwth-aachen.de/opus/volltexte/2005/1210/.

[Wen72a] B. Wendroff, *The Riemann problem for materials with nonconvex equations of state,* I: *Isentropic flow*, J. Math. Anal. Appl., 38 (1972), pp. 454–466.

[Wen72b] B. Wendroff, *The Riemann problem for materials with nonconvex equations of state,* II: *General flow*, J. Math. Anal. Appl., 38 (1972), pp. 640–658.

SIAM J. SCI. COMPUT.
Vol. 28, No. 2, pp. 682–700
© 2006 Society for Industrial and Applied Mathematics

SPECTRUM OF THE VOLUME INTEGRAL OPERATOR OF ELECTROMAGNETIC SCATTERING*

NEIL V. BUDKO† AND ALEXANDER B. SAMOKHIN‡

Abstract. The spectrum of the volume integral operator of three-dimensional electromagnetic scattering is analyzed. The operator has both continuous essential spectrum, which dominates at lower frequencies, and discrete eigenvalues, which spread out at higher ones. The explicit expression of the operator's symbol provides an exact outline of the essential spectrum for any inhomogeneous anisotropic scatterer with Hölder continuous constitutive parameters. Geometrical bounds on the location of discrete eigenvalues are derived for various physical scenarios. Numerical experiments demonstrate good agreement between the predicted spectrum of the operator and the eigenvalues of its discretized version.

Key words. electromagnetics, singular integral operators, symbol, spectrum, eigenvalues

AMS subject classifications. 78A25, 78A45, 45E10, 47A10

DOI. 10.1137/050630660

1. Introduction.

Knowledge of the spectrum of an integral operator is important for the efficient numerical solution of integral equations. For example, it may help to choose an iterative algorithm, estimate its convergence rate, and construct a proper preconditioner. The spectra of electromagnetic integral operators may also find direct applications in physics and engineering such as plasma studies, effective medium theories, and target identification.

Numerical estimation of the operator's spectrum after discretization is not too practical as it requires even more computational resources than the complete numerical solution of a scattering problem—not to mention that such a spectrum will be that of a matrix, which only approximates the original operator. In these circumstances any useful analytical prediction is extremely valuable.

In the pioneering papers [1] and [2] by Kleinman et al. the one-, two-, and (acoustical) three-dimensional cases and the (boundary) integral equations were considered. Here we look at the integral equation arising in vectorial three-dimensional electromagnetic scattering on penetrable objects known as or integral equation.

The first results concerning the spectrum of the three-dimensional electromagnetic scattering problems were derived from the analytical Mie solution for a spherical scatterer. This case was analyzed in considerable detail in [3], [4], [5], and the so-called were identified as eigenvalues. In [6] Rahola compared the theory of the Mie resonances with the numerical spectrum of the volume integral operator. Numerical experiments indicated, though, that apart from the expected discrete Mie

*Received by the editors May 4, 2005; accepted for publication (in revised form) November 30, 2005; published electronically May 26, 2006. This research was financed by the Netherlands Organization for Scientific Research (NWO) and by the Russian Foundation for Basic Research (RFBR).
http://www.siam.org/journals/sisc/28-2/63066.html

†Laboratory of Electromagnetic Research, Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, Mekelweg 4, 2628 CD, Delft, The Netherlands (n.budko@ewi.tudelft.nl).

‡Department of Applied Mathematics, Moscow Institute of Radio Engineering, Electronics, and Automatics (MIREA), Verndasky av. 78, 117454, Moscow, Russia (absamokhin@yandex.ru).

resonances, the spectrum of the discretized volume integral operator contained eigenvalues clustered along distinct line segments in the complex plane. However, the ad hoc explanation of this phenomenon presented in [6] cannot be extended to the case of a general inhomogeneous scatterer. The latter requires a different and more rigorous approach.

Our goal is to establish the connection between the operator's spectrum and the physical properties of a scatterer. Using the symbol of the electromagnetic volume integral operator we show that in addition to discrete eigenvalues the spectrum has an essential continuous part as well. Fortunately, the symbol can be derived explicitly allowing for exact predictions of the essential spectrum for any Hölder continuous anisotropic scatterer. In the isotropic case the essential spectrum is just the set of values admitted by the constitutive parameters of the object in $\mathbb{R}^3$. The approximate location of the discrete eigenvalues is estimated in the spirit of Kleinman et al. and depends mostly on the extent and shape of the object.

The paper is organized as follows. In the next two sections we recall several more or less well-known facts from the theory of Fredholm operators with pointwise multipliers and simple singular integral operators. Driven by the electromagnetic problem at hand in section 4 we consider "combined" operators consisting of both pointwise multiplication operators and simple singular integral operators. The theory of such combined operators was developed by Mikhlin and Prössdorf [7], [8], and we hope that our brief introduction will help to popularize their approach. The main purpose of these three sections is to introduce the concept of a symbol of a combined operator, which helps to establish the essential part of the operator's spectrum. Having mastered this tool we move on to the electromagnetic case where we construct the symbol of the volume integral operator, analyze the essential spectrum, and provide geometrical bounds on the distribution of discrete eigenvalues. Finally, we present some numerical experiments with the matrix version of the operator and try to comprehend similarities and differences between the continuous and the discrete worlds.

2. Spectrum of Fredholm operators with pointwise multipliers. The volume scattering in one and two dimensions is described by integral equations with manifestly Fredholm operators, i.e., an identity operator plus a compact integral operator. This is also the case in the acoustic three-dimensional scattering and has to do with the fact that the singularity in the kernel of all these integral operators is weak, i.e., its order is smaller than three (dimension of the spatial manifold), whereas the support of the scatterer is finite. For integral equations with Fredholm operators the questions of existence and uniqueness are linked, so that it is sufficient to establish the uniqueness of the solution only.

With operators which are not of the form "identity plus compact" the Fredholm property is established by proving the existence of a regularizer—an operator which, when applied to the operator in question, reduces it to the form "identity plus compact." There may be a left and a right regularizer, naturally. First, we clarify one important thing: the spectra of the original Fredholm operator and its regularized version are, in general, different. Indeed, consider the Fredholm integral equation

$$a(\mathbf{x})u(\mathbf{x}) + \int_{\mathbf{x},\mathbf{x}'\in D} K(\mathbf{x},\mathbf{x}')u(\mathbf{x}')\,\mathrm{d}\mathbf{x}' = v(\mathbf{x}), \tag{2.1}$$

where instead of the identity operator we have a (pointwise) multiplication with some continuous function $a(\mathbf{x}) \neq 0$, $\mathbf{x} \in D$. The integral operator is presumed to be

compact. In operator notation we shall write

$$[\mathrm{d}(a) + K]\, u = v, \tag{2.2}$$

where $\mathrm{d}(a)$ indicates that we have a multiplication operator, analogous to a diagonal matrix. The regularized Fredholm equation is obtained from (2.1) by a trivial operation of (pointwise) multiplication by $a(\mathbf{x})^{-1}$, i.e.,

$$u(\mathbf{x}) + a(\mathbf{x})^{-1} \int_{\mathbf{x},\mathbf{x}' \in D} K(\mathbf{x}, \mathbf{x}') u(\mathbf{x}')\, \mathrm{d}\mathbf{x}' = a(\mathbf{x})^{-1} v(\mathbf{x}). \tag{2.3}$$

Hence, the regularizer in this case is simply $\mathrm{d}(a)^{-1}$, as it gives the "identity plus compact" form

$$\mathrm{d}(a)^{-1}\, [\mathrm{d}(a) + K] = I + \mathrm{d}(a)^{-1} K = I + K', \tag{2.4}$$

where K' is a compact operator. The spectra of the two (original and regularized) operators are quite different though. Indeed, consider operator

$$\mathrm{d}(a) - \lambda I + K, \tag{2.5}$$

and let $\lambda = a(\mathbf{x}_0)$ for some arbitrary $\mathbf{x}_0 \in D$. If λ does not belong to the spectrum, then there exists a bounded inverse of (2.5) for that particular λ. This means that if we consider equality

$$[\mathrm{d}(a) - \lambda I + K]\, u = v, \tag{2.6}$$

then for all possible perturbations Δv of the right-hand side and the corresponding perturbations Δu of the solution there exists a constant $C < \infty$ such that

$$\|\Delta u\| \leq C \|\Delta v\|. \tag{2.7}$$

Due to the continuity of $a(\mathbf{x})$ in D we can always choose a neighborhood of $\mathbf{x}_0$, say, $\delta(\mathbf{x}_0)$, such that $|a(\mathbf{x}) - a(\mathbf{x}_0)| < \epsilon$ for $\mathbf{x} \in \delta(\mathbf{x}_0)$. Consider perturbations $\Delta u(\mathbf{x})$ which are zero outside $\delta(\mathbf{x}_0)$ and have a fixed norm $\|\Delta u\| = R$, independent of $\delta(\mathbf{x}_0)$. This can be a function of the type

$$\Delta u(\mathbf{x}) \sim e^{i\mathbf{x}\cdot\mathbf{k}(\delta,\epsilon)}, \quad \mathbf{x} \in \delta(\mathbf{x}_0), \tag{2.8}$$

or a similar oscillating function with varying spatial frequency $\mathbf{k}(\delta, \epsilon)$. Moreover, since K is compact, we may assume that our perturbations give $\|K\Delta u\| \leq \kappa\epsilon$ for all sufficiently small $\delta(\mathbf{x}_0)$. Further, we have $\|[a(\mathbf{x}) - a(\mathbf{x}_0)]\Delta u\| \leq R\epsilon$. Hence, for the perturbation of the right-hand side we obtain

$$\|\Delta v\| \leq \|(a - \lambda)\Delta u\| + \|K\Delta u\| \leq (R + \kappa)\epsilon\,, \tag{2.9}$$

whereas $\|\Delta u\| = R$ for all Δu. For the condition (2.7) to hold we must be able to provide $C < \infty$ such that

$$R \leq C(R + \kappa)\epsilon \tag{2.10}$$

for all ϵ, and this is obviously not possible, since we can make ϵ as small as we like by reducing $\delta(\mathbf{x}_0)$, while keeping R constant. Hence, the inverse of operator (2.5) is unbounded and $\lambda = a(\mathbf{x}_0)$ belongs to the spectrum of operator $[\mathrm{d}(a) + K]$.

If continuous function $a(\mathbf{x})$ is not constant on D, then the set of its values is an extended dense subset of the complex plane. The spectrum generated by such a function is called continuous as opposed to the usual discrete eigenvalues. In addition to continuous spectrum, operator $[\mathrm{d}(a) + K]$ may have discrete eigenvalues as well. The regularized operator $[I + K']$, however, is known to have a discrete spectrum only. This follows from the fact that the spectrum of K' is purely discrete [9]. Thus, regularization destroys the continuous spectrum. Finally, we notice that operator (2.5) cannot be regularized at the points of the continuous spectrum.

3. Spectrum of simple singular integral operators. The operator of three-dimensional electromagnetic scattering has the form

$$Au = \mathrm{d}(a)u + A^{\mathrm{s}}\mathrm{d}(b)u + Ku\,, \tag{3.1}$$

where K is compact and

$$A^{\mathrm{s}}\mathrm{d}(b)u = \lim_{\epsilon\to 0}\int_{\mathbf{x}'\in\mathbb{R}^3\setminus\delta(\epsilon)} \frac{f(\Theta)}{r^3} b(\mathbf{x}')u(\mathbf{x}')\,\mathrm{d}\mathbf{x}', \tag{3.2}$$

with $r = |\mathbf{x}' - \mathbf{x}|$, and $\Theta = (\mathbf{x}' - \mathbf{x})/r$. The principal-value integral (*p.v.*-integral) above is understood as an integral over the entire $\mathbb{R}^3$ excluding a ball around $\mathbf{x}$ whose radius tends to zero in the limit. This integral exists under certain assumptions on $b(\mathbf{x})u(\mathbf{x})$ and $f(\Theta)$. The necessary condition is

$$\int_{\Theta\in\mathbb{S}} f(\Theta)\mathrm{d}S = 0, \tag{3.3}$$

where integration is performed over a unit sphere $\mathbb{S}$. The Hölder continuity of $b(\mathbf{x})u(\mathbf{x})$ in $\mathbb{R}^3$ is another (sufficient) condition. Operator A^{s} in (3.2) is called the [illegible]. Its kernel is not an integrable function, and the best way to comprehend the meaning of such a *p.v.*-integral is to take the three-dimensional Fourier transform of $A^{\mathrm{s}}u$, i.e.,

$$\begin{aligned}
\tilde{v}(\mathbf{k}) &= \lim_{\epsilon\to 0}\int_{\mathbf{x}\in\mathbb{R}^3}\int_{\mathbf{x}'\in\mathbb{R}^3\setminus\delta(\epsilon)} \frac{f(\Theta)}{r^3} u(\mathbf{x}')e^{-i\mathbf{k}\cdot\mathbf{x}}\mathrm{d}\mathbf{x}'\mathrm{d}\mathbf{x} \\
&= \lim_{\epsilon\to 0}\int_{\mathbf{x}\in\mathbb{R}^3}\int_{\mathbf{x}'\in\mathbb{R}^3\setminus\delta(\epsilon)} \frac{f(\Theta)}{r^3} u(\mathbf{x}')e^{-i\mathbf{k}\cdot(\mathbf{x}-\mathbf{x}')}e^{-i\mathbf{k}\cdot\mathbf{x}'}\mathrm{d}\mathbf{x}'\mathrm{d}\mathbf{x} \\
&= \lim_{\epsilon\to 0}\int_{\mathbf{y}\in\mathbb{R}^3\setminus\delta(\epsilon)} \frac{f(-\Theta)}{r^3} e^{-i\mathbf{k}\cdot\mathbf{y}}\mathrm{d}\mathbf{y}\int_{\mathbf{x}'\in\mathbb{R}^3} u(\mathbf{x}')e^{-i\mathbf{k}\cdot\mathbf{x}'}\mathrm{d}\mathbf{x}' \\
&= F(\mathbf{k})\tilde{u}(\mathbf{k})\,.
\end{aligned} \tag{3.4}$$

In other words, we have arrived at a familiar $\mathbf{k}$-domain expression for the Fourier transform of a convolution. The function $F(\mathbf{k})$ can only be found if one knows the explicit form of $f(\Theta)$. However, there is something to say even in the general case.

Consider this:

$$
\begin{aligned}
F(\mathbf{k}) &= \lim_{\epsilon\to 0}\int_{\mathbf{y}\in\mathbb{R}^3\setminus\delta(\epsilon)} \frac{f(-\Theta)}{r^3}e^{-i\mathbf{k}\cdot\mathbf{y}}\mathrm{d}\mathbf{y} \\
&= \lim_{\epsilon\to 0}\int_{\epsilon}^{\infty}\int_S \frac{f(-\Theta)}{r}e^{-ir\rho\cos\gamma}\mathrm{d}S\mathrm{d}r = \lim_{\epsilon\to 0}\int_{\rho\epsilon}^{\infty}\int_S f(-\Theta)\frac{e^{-it\cos\gamma}}{t}\mathrm{d}S\mathrm{d}t \\
&= \lim_{\epsilon\to 0}\int_{\rho\epsilon}^{1}\int_S f(-\Theta)\frac{e^{-it\cos\gamma}}{t}\mathrm{d}S\mathrm{d}t + \int_1^{\infty}\int_S f(-\Theta)\frac{e^{-it\cos\gamma}}{t}\mathrm{d}S\mathrm{d}t \\
&= \int_1^{\infty}\int_S f(-\Theta)\frac{e^{-it\cos\gamma}}{t}\mathrm{d}S\mathrm{d}t + \lim_{\epsilon\to 0}\int_{\rho\epsilon}^{1}\int_S f(-\Theta)\frac{1}{t}\left[e^{-it\cos\gamma}-1\right]\mathrm{d}S\mathrm{d}t \\
&\quad + \lim_{\epsilon\to 0}\int_{\rho\epsilon}^{1}\frac{1}{t}\mathrm{d}t\int_S f(-\Theta)\mathrm{d}S \\
&= \int_1^{\infty}\int_S f(-\Theta)\frac{e^{-it\cos\gamma}}{t}\mathrm{d}S\mathrm{d}t + \int_0^{1}\int_S f(-\Theta)\frac{1}{t}\left[e^{-it\cos\gamma}-1\right]\mathrm{d}S\mathrm{d}t.
\end{aligned}
\tag{3.5}
$$

Above we have introduced spherical coordinate systems for both $\mathbf{y}$ and $\mathbf{k}$, where $\rho = |\mathbf{k}|$, and used condition (3.3). As one can see, since $\epsilon \to 0$ tends to zero independently of ρ, $F(\mathbf{k})$ does not depend on ρ, and, hence, it is independent of $|\mathbf{k}|$. In other words, if $F(\mathbf{k})$ depends on $\mathbf{k}$ at all, then it is only the direction of $\mathbf{k}$, not its magnitude, i.e.,

$$
F(\mathbf{k}) = F\left(\frac{\mathbf{k}}{|\mathbf{k}|}\right) = F(\tilde{\Theta})\,. \tag{3.6}
$$

Note that this is only true if the order of singularity in (3.2) is exactly three. If we had a weakly singular integral, then $F(\mathbf{k})$ would depend on $|\mathbf{k}|$ explicitly.

Let us now construct a regularizer for a simple singular operator A^{s}. Denoting by $\mathcal{F}$ and $\mathcal{F}^{-1}$, correspondingly, the forward and the inverse Fourier transforms, and by $F(\tilde{\Theta})^{-1}$ the inverse of $F(\tilde{\Theta})$, we can do, for example, the following:

$$
\mathcal{F}^{-1}F(\tilde{\Theta})^{-1}\mathcal{F}A^{\mathrm{s}}u = \mathcal{F}^{-1}F(\tilde{\Theta})^{-1}F(\tilde{\Theta})\mathcal{F}u = \mathcal{F}^{-1}\mathcal{F}u = u\,. \tag{3.7}
$$

In this way the original operator A^{s} is reduced by the regularizer $\mathcal{F}^{-1}F(\tilde{\Theta})^{-1}\mathcal{F}$ to an identity operator; i.e., the regularizer is also an inverse of the operator.

To recover the spectrum of a simple singular operator we need to consider

$$
\left[A^{\mathrm{s}} - \lambda I\right]u = \lim_{\epsilon\to 0}\int_{\mathbf{x}'\in\mathbb{R}^3\setminus\delta(\epsilon)}\frac{f(\Theta)}{r^3}u(\mathbf{x}')\,\mathrm{d}\mathbf{x}' - \lambda u(\mathbf{x})\,. \tag{3.8}
$$

Repeating calculations (3.4) we arrive at the following equivalent version of the spectral problem:

$$
F(\tilde{\Theta})\tilde{u}(\mathbf{k}) = \lambda\tilde{u}(\mathbf{k}). \tag{3.9}
$$

Hence, generalizing this to matrix-valued $F(\tilde{\Theta})$, we conclude that the domain $s(\lambda)$ in the complex plane consisting of all eigenvalues λ of matrix $F(\tilde{\Theta})$ for all $\tilde{\Theta}$ on the unit sphere belongs to the spectrum of the simple singular integral operator A^{s}. Note that, although matrix eigenvalues are always discrete, they may vary continuously with $\tilde{\Theta}$, thus creating a continuous spectrum. Again, the operator of (3.8) cannot be regularized for $\lambda \in s(\lambda)$.

4. Essential spectrum of combined operators. In the previous two sections we saw that the spectrum and the regularization of operators are tightly linked. Namely, operators of the spectral problems (2.5) and (3.8) could not be regularized for $\lambda \in s(\lambda)$. This is also true if we consider the spectral problem for operator (3.1), and, perhaps, for any operator. However, a rigorous proof of this link for a general case is too abstract, and we shall not provide it here. Taking it for granted we shall instead concentrate on the construction of a general regularizer.

We have shown that the Fredholm operator with free term $\mathrm{d}(a)$ could be regularized by $\mathrm{d}(a)^{-1}$, whereas a simple singular integral operator is regularized by $\mathcal{F}^{-1}F(\tilde{\Theta})^{-1}\mathcal{F}$. However, neither of these approaches works with the general operator (3.1) arising in the electromagnetic scattering problem. Since this general operator is a combination of multiplication and simple singular operators (further—a [illegible] operator) we expect that a proper combination of the two approaches may work.

The technique presented below is a simplified version of the approach suggested by Mikhlin [7], [8]. The first thing to note is that, while searching for a regularizer, the actual form of the compact operator in the resulting "identity plus compact" expression is not important. Second, a regularizer cannot be a compact operator, as the product of a compact operator with any other bounded linear operator is a compact operator. Finally, the product of any two combined operators is also a combined operator (plus a compact one, in general). For example, the product of two operators of type (3.1) will look like

$$\begin{aligned} A_1A_2 &= [\mathrm{d}(a_1) + A_1^{\mathrm{s}}\mathrm{d}(b_1) + K_1]\,[\mathrm{d}(a_2) + A_2^{\mathrm{s}}\mathrm{d}(b_2) + K_2] \\ &= \mathrm{d}(a_1)\mathrm{d}(a_2) + \mathrm{d}(a_1)A_2^{\mathrm{s}}\mathrm{d}(b_2) + A_1^{\mathrm{s}}\mathrm{d}(b_1)\mathrm{d}(a_2) + A_1^{\mathrm{s}}\mathrm{d}(b_1)A_2^{\mathrm{s}}\mathrm{d}(b_2) + K, \end{aligned} \tag{4.1}$$

where all compact operators are hidden in the generic compact operator K.

To every combined operator there corresponds a function called a [illegible] constructed in accordance with the following set of rules.

DEFINITION 4.1.

1. [illegible]
2. [illegible]
3. [illegible]
4. [illegible]

For the last property to hold it is sufficient to ensure that the multipliers are Hölder continuous functions in $\mathbb{R}^3$, so that the products of simple singular integral operators and multiplication operators and the products of their symbols are well defined. According to Definition 4.1, the symbol of the combined operator (3.1) is just the function

$$\mathrm{Symb}\,[A] = \Phi(\mathbf{x}, \tilde{\Theta}) = a(\mathbf{x}) + F(\tilde{\Theta})b(\mathbf{x})\,, \tag{4.2}$$

and the symbol of the product (4.1) of two combined operators is

$$\begin{aligned} \mathrm{Symb}\,[A_1A_2] &= a_1(\mathbf{x})a_2(\mathbf{x}) + a_1(\mathbf{x})F_2(\tilde{\Theta})b_2(\mathbf{x}) + F_1(\tilde{\Theta})b_1(\mathbf{x})a_2(\mathbf{x}) \\ &\quad + F_1(\tilde{\Theta})b_1(\mathbf{x})F_2(\tilde{\Theta})b_2(\mathbf{x}) = \Phi_1(\mathbf{x}, \tilde{\Theta})\Phi_2(\mathbf{x}, \tilde{\Theta}) \\ &= \mathrm{Symb}\,[A_1]\,\mathrm{Symb}\,[A_2]\,. \end{aligned} \tag{4.3}$$

For our purposes the most important property of a symbol is that the correspondence between a combined operator and its symbol is one-to-one up to a compact operator. This means that to each combined operator there corresponds only one symbol, whereas to each symbol there may correspond more than one operator, but their difference would be a compact operator.

Suppose now that we want to find a regularizer A^{r} of a combined operator A, i.e.,

$$A^{\mathrm{r}}A = A^{\mathrm{r}}\left[\mathrm{d}(a) + A^{\mathrm{s}}\mathrm{d}(b) + K\right] = I + K'. \tag{4.4}$$

The symbol of this product can be constructed using Definition 4.1 and is given by

$$\mathrm{Symb}\left[A^{\mathrm{r}}A\right] = \mathrm{Symb}\left[A^{\mathrm{r}}\right]\left[a(\mathbf{x}) + F(\tilde{\Theta})b(\mathbf{x})\right] = 1. \tag{4.5}$$

To this symbol there corresponds an operator of the form "identity plus compact"; i.e., should we succeed in obtaining such a unit symbol, the regularization is accomplished. Therefore, the symbol of a regularizer, say, $\Phi_{\mathrm{r}}(\mathbf{x}, \tilde{\Theta})$, must be an inverse of the symbol of the combined operator in question. Since the latter symbol may be a matrix-valued function, one may have to find an inverse of a matrix.

In this simplified version of Mikhlin's approach summarized in Definition 4.1 we have limited ourselves to combined operators. Therefore, in order to be able to apply the multiplication property in (4.5) we have to be sure that the regularizer, if it exists, belongs to the class of combined operators as well. This would be the case if the inverse of the symbol of a combined operator, $[a(\mathbf{x}) + F(\tilde{\Theta})b(\mathbf{x})]^{-1}$, had the form of a symbol of a combined operator itself, i.e., contained only multipliers and Fourier transforms of simple singular integral operators. Fortunately, it follows from the Cayley–Hamilton theorem for matrices that the inverse of a nonsingular matrix can always be represented as a polynomial of this matrix. Obviously, any such polynomial contains only the products of the multipliers and the Fourier transforms mentioned above.

Conditions on the existence of a regularizer thus become conditions on the existence and boundedness of the inverse of the symbol of a combined operator for all $\mathbf{x} \in \mathbb{R}^3$ and all $\tilde{\Theta}$ on a unit sphere.

The symbol will help us to recover the essential spectrum of combined operator A given by (3.1). Since operator $A - \lambda I$ is also a combined operator, its symbol is given by

$$\Phi(\mathbf{x}, \tilde{\Theta}, \lambda) = a(\mathbf{x}) + F(\tilde{\Theta})b(\mathbf{x}) - \lambda\mathbb{I} = \Phi(\mathbf{x}, \tilde{\Theta}) - \lambda\mathbb{I}\,, \tag{4.6}$$

where $\mathbb{I}$ is an identity matrix of the corresponding dimensions. This symbol is a parametric function of λ as well as $\mathbf{x}$ and $\tilde{\Theta}$. Suppose that λ coincides with the eigenvalue of the symbol matrix $\Phi(\mathbf{x}, \tilde{\Theta})$ for some $\mathbf{x}$ and $\tilde{\Theta}$. Then, operator $A - \lambda I$ cannot be regularized and, hence, does not have a bounded inverse. In summary (generalizing to matrix-valued symbols), the domain $s(\lambda)$ in the complex plane consisting of all eigenvalues of symbol $\Phi(\mathbf{x}, \tilde{\Theta})$ for all $\mathbf{x} \in \mathbb{R}^3$ and all $\tilde{\Theta} \in \mathbb{S}$ belongs to the (essential) spectrum of the corresponding combined operator A. If the eigenvalues of $\Phi(\mathbf{x}, \tilde{\Theta})$ vary continuously in the complex plane with $\mathbf{x}$ and $\tilde{\Theta}$ varying correspondingly in $\mathbb{R}^3$ and on $\mathbb{S}$, then the essential spectrum is continuous.

5. The electromagnetic case. Now we turn to the electromagnetic three-dimensional case. In the frequency domain with time-factor $\exp(-i\omega t)$ the total

electric field $\mathbf{E}$ in the presence of an inhomogeneous anisotropic electric-type scatterer satisfies the singular integral equation (see, e.g., [12])

$$\begin{aligned}\mathbf{E}^{\mathrm{in}}(\mathbf{x},\omega) = &\left[\mathbb{I} + \frac{1}{3}\chi(\mathbf{x},\omega)\right]\mathbf{E}(\mathbf{x},\omega)\\ &- \lim_{\epsilon\to 0}\int_{\mathbf{x}'\in\mathbb{R}^3\setminus|\mathbf{x}-\mathbf{x}'|<\epsilon} \mathbb{G}_0(\mathbf{x}-\mathbf{x}')\chi(\mathbf{x}',\omega)\mathbf{E}(\mathbf{x}',\omega)\,\mathrm{d}\mathbf{x}'\\ &- \int_{\mathbf{x}'\in\mathbb{R}^3} \mathbb{G}_1(\mathbf{x}-\mathbf{x}',\omega)\chi(\mathbf{x}',\omega)\mathbf{E}(\mathbf{x}',\omega)\,\mathrm{d}\mathbf{x}',\end{aligned} \tag{5.1}$$

where $\mathbf{E}^{\mathrm{in}}$ is a given incident electric field, and χ is the tensor of relative electric contrast of the object with respect to a homogeneous isotropic background. Explicitly it is given by

$$\chi(\mathbf{x},\omega) = \frac{1}{\eta_{\mathrm{b}}(\omega)}\eta(\mathbf{x},\omega) - \mathbb{I}, \tag{5.2}$$

where $\mathbb{I}$ denotes the 3×3 identity matrix and the constitutive parameters of the scatterer are described by the tensor of the transverse medium admittance (per length), which in its turn contains the more familiar tensors of conductivity $\sigma(\mathbf{x},\omega)$ and dielectric permittivity $\varepsilon(\mathbf{x},\omega)$, namely,

$$\eta(\mathbf{x},\omega) = \sigma(\mathbf{x},\omega) - i\omega\varepsilon(\mathbf{x},\omega). \tag{5.3}$$

The background medium has parameters $\eta_{\mathrm{b}}(\omega) = \sigma_{\mathrm{b}}(\omega) - i\omega\varepsilon_{\mathrm{b}}(\omega)$ and $\zeta_{\mathrm{b}}(\omega) = -i\omega\mu_{\mathrm{b}}(\omega)$, where μ_{b} is the magnetic permeability. The so-called wavenumber $k_{\mathrm{b}}(\omega)$ of this medium can be found from $k_{\mathrm{b}}^2(\omega) = -\eta(\omega)\zeta(\omega)$ by taking a proper branch of a square root. Although the contrast function is different from zero only within a finite spatial domain D, we have extended the limits of integration in (5.1) to the whole of $\mathbb{R}^3$. This formal extension allows direct application of the results of the previous section.

The kernels of the intregal operators in (5.1) contain tensors $\mathbb{G}_0$ and $\mathbb{G}_1$, the first of which is given by

$$\mathbb{G}_0(\mathbf{x}) = \frac{1}{4\pi|\mathbf{x}|^3}\left[3\,\mathbb{Q} - \mathbb{I}\right], \tag{5.4}$$

where tensor $\mathbb{Q} = \Theta(\Theta\cdot\,)$ can also be written in matrix notation as $\mathbb{Q} = \Theta\Theta^T$. This tensor is an orthogonal projector, i.e., $\mathbb{Q}^2 = \Theta\Theta^T\Theta\Theta^T = \mathbb{Q}$. The second kernel contains tensor

$$\begin{aligned}\mathbb{G}_1(\mathbf{x},\omega) = &\left[-\alpha(\mathbf{x},\omega) - 3\beta(\mathbf{x},\omega) + 3\gamma(\mathbf{x},\omega)\right]\mathbb{Q}\\ &+ \left[\alpha(\mathbf{x},\omega) + \beta(\mathbf{x},\omega) - \gamma(\mathbf{x},\omega)\right]\mathbb{I},\end{aligned} \tag{5.5}$$

with

$$\begin{aligned}&\alpha(\mathbf{x},\omega) = k_{\mathrm{b}}^2(\omega)\frac{e^{ik_{\mathrm{b}}(\omega)|\mathbf{x}|}}{4\pi|\mathbf{x}|}, \quad \beta(\mathbf{x},\omega) = i\,k_{\mathrm{b}}(\omega)\frac{e^{ik_{\mathrm{b}}(\omega)|\mathbf{x}|}}{4\pi|\mathbf{x}|^2},\\ &\gamma(\mathbf{x},\omega) = \frac{e^{ik_{\mathrm{b}}(\omega)|\mathbf{x}|}-1}{4\pi|\mathbf{x}|^3}.\end{aligned} \tag{5.6}$$

In the operator notation of the previous section our equation is written as

$$\left[\mathrm{d}\left(\mathbb{I}+\frac{1}{3}\chi\right)-A^{\mathrm{s}}\mathrm{d}(\chi)+K\right]u=u^{\mathrm{in}}. \tag{5.7}$$

The operator of this equation acts in the Hilbert space $L^2_{(3)}(\mathbb{R}^3)$ of square-integrable vector-valued functions. This choice of the functional space is physically motivated by the energy considerations. All components of the matrix-valued function $\chi(\mathbf{x},\omega)$ are presumed to be Hölder continuous in $\mathbb{R}^3$, i.e., not only inside the object, but also across its outer boundary. As previously K denotes a compact operator, which in this case is the last integral operator in (5.1) with weakly singular tensors (5.5)–(5.6) in its kernel. Although, in general, weakly singular integral operators are not compact on $\mathbb{R}^3$, the kernel of the last operator in (5.1) contains the contrast function as well, which effectively restricts the spatial support of the domain of the integral operator to D. Therefore, the total operator is compact.

There exist other formulations of this problem. For example, for isotropic scatterers with differentiable constitutive parameters the regularization of the integral equation can be carried out explicitly; see [10], [11]. However, from the discussion of the previous sections we know that the spectra of the original and the regularized operator may be very different.

To find the spectrum of our operator we first need to derive its symbol. The most laborious task here is, obviously, the Fourier transform of the simple singular integral operator. The function $f(\Theta)$ required in (3.5) is in our case

$$f(\Theta)=\frac{1}{4\pi}\left[3\mathbb{Q}-\mathbb{I}\right]. \tag{5.8}$$

It is a matrix-valued function and therefore its Fourier image $F(\tilde{\Theta})$ is also a matrix-valued function. The complete symbol of the electromagnetic volume integral operator was first obtained in [12] using the technique suggested in [8], where $f(\Theta)$ is expanded in terms of spherical harmonics. Comparing the result of [12] with the present formulation we find that

$$F(\tilde{\Theta})=\frac{1}{3}\mathbb{I}-\tilde{\mathbb{Q}}, \tag{5.9}$$

where $\tilde{\mathbb{Q}}=\tilde{\Theta}(\tilde{\Theta}\cdot\,)$, or in matrix notation $\tilde{\mathbb{Q}}=\tilde{\Theta}\tilde{\Theta}^T$. Now using Definition 4.1 and expression (5.2) we obtain the symbol of the electromagnetic volume integral operator for an anisotropic scatterer as

$$\Phi(\mathbf{x},\tilde{\Theta})=\mathbb{I}+\tilde{\mathbb{Q}}\chi(\mathbf{x},\omega)=\mathbb{I}-\tilde{\mathbb{Q}}+\tilde{\mathbb{Q}}\,\eta_{\mathrm{r}}(\mathbf{x},\omega), \tag{5.10}$$

where both χ and $\eta_{\mathrm{r}}=\eta/\eta_{\mathrm{b}}$ are matrix-valued functions and in general do not commute with matrix $\tilde{\mathbb{Q}}$. The inverse of this symbol can be found explicitly as

$$\Phi^{-1}(\mathbf{x},\tilde{\Theta})=\mathbb{I}+\xi^{-1}\tilde{\mathbb{Q}}-\xi^{-1}\tilde{\mathbb{Q}}\,\eta_{\mathrm{r}}(\mathbf{x},\omega), \tag{5.11}$$

where

$$\xi(\mathbf{x},\tilde{\Theta})=\tilde{\Theta}^T\eta_{\mathrm{r}}(\mathbf{x},\omega)\tilde{\Theta}. \tag{5.12}$$

In the isotropic case, where $\eta(\mathbf{x},\omega)$ is a scalar, the inverse of the symbol looks even simpler:

$$\Phi^{-1}_{\mathrm{isotropic}}(\mathbf{x},\tilde{\Theta})=\mathbb{I}-\tilde{\mathbb{Q}}+\tilde{\mathbb{Q}}\,\eta_{\mathrm{r}}^{-1}(\mathbf{x},\omega)=\mathbb{I}+\tilde{\mathbb{Q}}\chi'(\mathbf{x},\omega), \tag{5.13}$$

with χ' given by

$$\chi'(\mathbf{x},\omega) = \frac{\eta_{\mathrm{b}}(\omega)}{\eta(\mathbf{x},\omega)} - 1. \tag{5.14}$$

Although it is not our primary concern, now we can find the explicit electromagnetic regularizer. Notice that expression (5.13) is identical in form to the symbol (5.10) of the original operator. This means that the operator corresponding to Φ^{-1} has the same form as the operator corresponding to Φ. Hence in the isotropic case the regularizer is

$$A^{\mathrm{r}}_{\mathrm{isotropic}} = \mathrm{d}\left(\mathbb{I} + \frac{1}{3}\chi'\right) - A^{\mathrm{s}}\mathrm{d}(\chi'), \tag{5.15}$$

with χ' given by (5.14).

The essential spectrum consists of the eigenvalues of the symbol matrix $\Phi(\mathbf{x},\tilde{\Theta})$ for all $\mathbf{x} \in \mathbb{R}^3$ and all $\tilde{\Theta}$ on the unit sphere $\mathbb{S}$. Direct computations show that in the anisotropic case these eigenvalues are $\lambda_1 = \lambda_2 = 1$ and $\lambda_3 = \tilde{\Theta}^T \eta_{\mathrm{r}}(\mathbf{x},\omega)\tilde{\Theta}$, and they depend on both $\mathbf{x}$ and $\tilde{\Theta}$, whereas in the isotropic case, where η_{r} is a scalar, the eigenvalues are $\lambda_1 = \lambda_2 = 1$ and $\lambda_3 = \eta_{\mathrm{r}}(\mathbf{x},\omega)$, and they depend on $\mathbf{x}$ only. Hence in general the essential spectrum is given by

$$\lambda_1^{\mathrm{ess}} = \lambda_2^{\mathrm{ess}} = 1, \qquad \lambda_3^{\mathrm{ess}} = \tilde{\Theta}^T \eta_{\mathrm{r}}(\mathbf{x},\omega)\tilde{\Theta}, \quad \mathbf{x} \in \mathbb{R}^3, \quad \tilde{\Theta} \in \mathbb{S}. \tag{5.16}$$

Notice that there is also a parametric dependence on the angular frequency ω in both isotropic and anisotropic cases, which matters if the scatterer is conductive and/or dispersive.

The object parameters are presumed to be Hölder continuous functions of $\mathbf{x}$ on $\mathbb{R}^3$. This means that even with a homogeneous scatterer, characterized by $\eta_{\mathrm{r}}(\mathbf{x},\omega) = \text{const}$, $\mathbf{x} \in D$, there exists a smooth transition between η_{r} and 1, the background medium parameter. Then, the essential spectrum must connect 1 and point η_{r} in the complex plane. Moreover, for any inhomogeneous scatterer the essential spectrum will emerge from point 1 on the real axis and continuously span all values admitted by $\tilde{\Theta}^T \eta_{\mathrm{r}}(\mathbf{x},\omega)\tilde{\Theta}$ for $\mathbf{x} \in D$ and $\tilde{\Theta} \in \mathbb{S}$.

Apart from the essential spectrum our operator may have discrete eigenvalues, which correspond to nontrivial solutions of the problem

$$\left[\mathrm{d}\left(\mathbb{I} + \frac{1}{3}\chi\right) - A^{\mathrm{s}}\mathrm{d}(\chi) + K\right] u - \lambda u = 0. \tag{5.17}$$

We shall consider here only the isotropic case. In this respect it is convenient to use the standard sufficient uniqueness condition for the original problem (5.7), which in terms of the admittance function $\eta(\mathbf{x},\omega)$ given by (5.3) can be written as

$$\mathrm{Re}\,[\eta(\mathbf{x},\omega)] > 0, \quad \mathbf{x} \in D. \tag{5.18}$$

Under this condition the null-space of the volume integral operator (operator in square brackets in (5.17)) is known to be trivial. Considering the case where $\lambda \neq 1$ we can rewrite (5.17) as

$$\left[\mathrm{d}\left(\mathbb{I} + \frac{1}{3}\chi'\right) - A^{\mathrm{s}}\mathrm{d}(\chi') + K'\right] u = 0, \tag{5.19}$$

where the only difference from the original operator is the new contrast function χ' given by

$$\chi'(\mathbf{x},\omega,\lambda)=\frac{\chi(\mathbf{x},\omega)}{1-\lambda}=\frac{\eta(\mathbf{x},\omega)-\lambda\eta_{\mathrm{b}}(\omega)}{\eta_{\mathrm{b}}(1-\lambda)}-1=\frac{\eta'(\mathbf{x},\omega,\lambda)}{\eta_{\mathrm{b}}(\omega)}-1. \tag{5.20}$$

And since the eigenvalue problem has been reduced to the problem about the non-trivial null-space of the original operator we can use an "inverse" of the condition (5.18), i.e.,

$$\mathrm{Re}\,[\eta'(\mathbf{x},\omega,\lambda)]\le 0,\quad \mathbf{x}\in D. \tag{5.21}$$

From here we derive the following bound on the eigenvalues:

$$\begin{aligned}\mathrm{Re}\,\eta(\mathbf{x},\omega)-[\mathrm{Re}\,\eta(\mathbf{x},\omega)+\mathrm{Re}\,\eta_{\mathrm{b}}(\omega)]\,\mathrm{Re}\lambda\\ -[\mathrm{Im}\,\eta(\mathbf{x},\omega)-\mathrm{Im}\,\eta_{\mathrm{b}}(\omega)]\,\mathrm{Im}\lambda+\mathrm{Re}\,\eta_{\mathrm{b}}(\omega)|\lambda|^2\le 0,\quad \mathbf{x}\in D.\end{aligned} \tag{5.22}$$

In addition we note that

$$|\lambda|\le\|A\|<\infty, \tag{5.23}$$

since A is bounded, and that $\lambda=0$ is not an eigenvalue if the scattering problem has a unique solution (which we presume from now on).

The only natural homogeneous background medium is a vacuum, where $\sigma_{\mathrm{b}}(\omega)=0$ and $\varepsilon_{\mathrm{b}}(\omega)=\varepsilon_0$. In particular cases, such as scattering from subsurface objects, other homogeneous backgrounds can be used as an approximation. We shall deal with those later. Let us first consider the natural case, and in addition presume that $\sigma(\mathbf{x},\omega)=\sigma(\mathbf{x})$ and $\varepsilon(\mathbf{x},\omega)=\varepsilon(\mathbf{x})$ and are real-valued functions. Further we also assume that the contrast in permittivity $\varepsilon(\mathbf{x})-\varepsilon_{\mathrm{b}}$ is positive for all $\mathbf{x}\in D$. These assumptions hold for a static model of a conducting dielectric, which is applicable at low frequencies where the dispersion of constitutive paremeters is insignificant. In this case the eigenvalue bound (5.22) reduces to

$$\sigma(\mathbf{x})-\sigma(\mathbf{x})\mathrm{Re}\lambda+\omega\,[\varepsilon(\mathbf{x})-\varepsilon_0]\,\mathrm{Im}\lambda\le 0,\quad \mathbf{x}\in D. \tag{5.24}$$

In Figures 5.1 and 5.2 (left) we present geometric estimates derived from this formula for the following three situations:

$$\sigma(\mathbf{x})=0,\qquad 1<\frac{\varepsilon(\mathbf{x})}{\varepsilon_0}\le M_0, \tag{5.25}$$

$$\sigma(\mathbf{x})=\mathrm{const}>0,\quad \varepsilon(\mathbf{x})=\mathrm{const}>\varepsilon_0, \tag{5.26}$$

$$\sigma(\mathbf{x})>0,\qquad \varepsilon(\mathbf{x})>\varepsilon_0, \tag{5.27}$$

where $\mathbf{x}\in D$. The other parameters in these figures are given by

$$M=\frac{\varepsilon}{\varepsilon_0}+i\frac{\sigma}{\omega\varepsilon_0},\quad N=-i\frac{\sigma}{\omega(\varepsilon-\varepsilon_0)}, \tag{5.28}$$

$$N_1=-i\max_{\mathbf{x}\in D}\frac{\sigma(\mathbf{x})}{\omega\,[\varepsilon(\mathbf{x})-\varepsilon_0]}, \tag{5.29}$$

$$N_2=-i\min_{\mathbf{x}\in D}\frac{\sigma(\mathbf{x})}{\omega\,[\varepsilon(\mathbf{x})-\varepsilon_0]}. \tag{5.30}$$

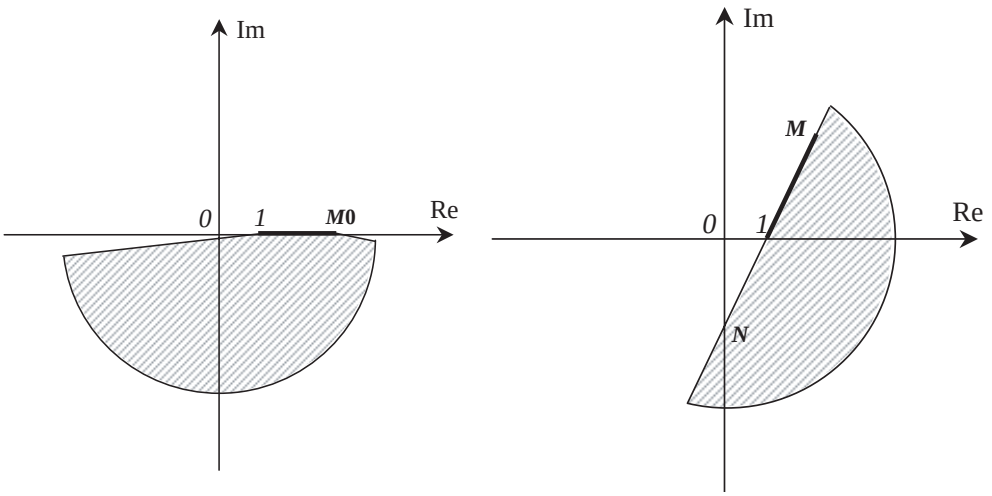


FIG. 5.1. *Location of the spectrum for a homogeneous object in a vacuum. Left: Nonconducting dielectric. Right: Conducting dielectric.*

Fat solid lines and curves schematically outline the essential part of the spectrum, which in the case of a homogeneous object we presume to be straight lines connecting the real 1 and the corresponding value of η/η_b. As one can see, in the natural (vacuum) background medium, the bounds are wedge shaped, as in [1]. In the case of an object with losses these bounds safely separate the domain of eigenvalues from the zero of the complex plane. This is important since the uniqueness condition can guarantee only that there are no eigenvalues [illegible] to zero but does not tell if there are any eigenvalues [illegible] to zero, and such eigenvalues may cause instabilities in the numerical solution and slow down the convergence of an iterative algorithm. From this point of view the worst case is the lossless object or an inhomogeneous object with lossless parts.

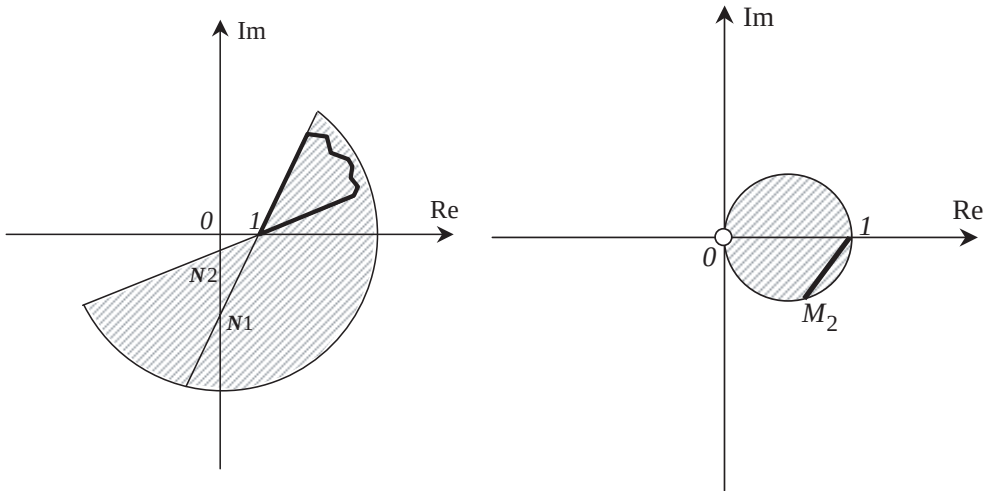


FIG. 5.2. *Location of the spectrum. Left: Inhomogeneous conducting dielectric in a vacuum. Right: Homogeneous nonconducting dielectric object in a conducting dielectric background (no contrast in permittivity).*

Now let us analyze the popular approximation, where the homogeneous background medium is considered to be conducting, i.e., $\sigma_\mathrm{b} > 0$, and/or has arbitrary constant permittivity $\varepsilon_\mathrm{b} > \varepsilon_0$. Such approximations are often used if a scatterer is located inside a large but finite and more or less homogeneous host medium, say, inside the Earth or a human body, and the reflections from the outer boundary of the host medium can be neglected. We again consider the static model of a dielectric and from (5.22) we derive the estimate

$$\left[\mathrm{Re}\lambda - \frac{A}{2}\right]^2 + \left[\mathrm{Im}\lambda + \frac{B}{2}\right]^2 < \left(\frac{D}{2}\right)^2, \tag{5.31}$$

which shows that all eigenvalues are situated inside a circle. Parameters of this circle's center are

$$A = \frac{\sigma(\mathbf{x})}{\sigma_\mathrm{b}} + 1, \quad B = \frac{\omega}{\sigma_\mathrm{b}}\left[\varepsilon(\mathbf{x}) - \varepsilon_\mathrm{b}\right], \tag{5.32}$$

and its diameter can be found from

$$D^2 = \left[\frac{\sigma(\mathbf{x})}{\sigma_\mathrm{b}} - 1\right]^2 + \omega^2\left[\frac{\varepsilon(\mathbf{x}) - \varepsilon_\mathrm{b}}{\sigma_\mathrm{b}}\right]^2. \tag{5.33}$$

Figure 5.2 (right) corresponds to a particular situation where a nonconducting object ($\sigma(\mathbf{x}) = 0$) with no contrast in permittivity ($\varepsilon(\mathbf{x}) = \varepsilon_\mathrm{b}$) is immersed in a conducting background. This can represent an air pocket or a low-contrast plastic landmine in wet soil. The constant M_2 is given by

$$M_2 = \frac{\eta}{\eta_\mathrm{b}} = \frac{\omega^2\varepsilon_\mathrm{b}^2}{\sigma_\mathrm{b}^2 + \omega^2\varepsilon_\mathrm{b}^2} - i\frac{\omega\varepsilon_\mathrm{b}\sigma_\mathrm{b}}{\sigma_\mathrm{b}^2 + \omega^2\varepsilon_\mathrm{b}^2}. \tag{5.34}$$

On one hand, the spectrum is now explicitly bounded, allowing, in particular, for a more precise convergence estimate when an iterative method is employed to solve the scattering problem. On the other hand, the circle extends toward the zero of the complex plane as in Figure 5.2 (right), meaning that, although there is no zero eigenvalue in the spectrum, some of the eigenvalues may get close to zero and render the scattering problem unstable.

The last point to note is about the relative weight of the compact and singular parts in the operator of (5.1) as a function of angular frequency ω. The coefficients α and β in (5.6) depend on the wavenumber $k_\mathrm{b}(\omega)$, which in its turn is proportional to ω. Hence, the kernel of the compact operator gets more "weight" as the frequency increases, and the norm of the operator increases as well. As the compact operator delivers only eigenvalues, we can expect that the eigenvalues will spread out at higher frequencies.

6. Numerical experiments. There is only a limited correspondence between the spectra of an integral operator and of its discretized (matrix) version. First, matrices do not have continuous spectra, but only discrete eigenvalues. Hence, we should not expect to see the lines and curves of the previous section. On the other hand, the continuous spectrum may serve as an accumulation area for the eigenvalues, as the latter should converge to the spectrum of an operator in the continuous limit. Let us first see if it is indeed so.

We discretize the operator of (5.1) using the standard collocation method (see, e.g., [6], [12]), which gives an order h^2 accuracy, where h is the size of an elementary

cubic cell. The scatterer is discretized on an N-point homogeneous grid, and the resulting matrix is $3N \times 3N$ and is completely filled with complex numbers. The matrix eigenvalues are computed using the standard MATLAB function `eig`.

According to our theoretical predictions the essential spectrum, continuous or not, dominates at lower frequencies (small objects). Therefore, we shall first consider an object whose extent is sufficiently small. The background medium is a vacuum and the frequency is set to $\omega/(2\pi) = 1$ GHz. The scatterer has the shape of a cube with side length $a \approx (\lambda_{\text{med}}/20)$, where λ_{med} is the medium wavelength; see Figure 6.1. According to the common ad hoc discretization rule such a small object can be modeled as a single cell (Figure 6.1, left). This, of course, would only give an order h^2 approximation of the field at the geometrical center of the scatterer. In an attempt to compute the field at other points inside the object one may wish to introduce a finer discretization. Then, if the object is homogeneous, i.e., $\eta(\mathbf{x}) = \text{const}$, the value of the constitutive parameter would be the same at all internal grid points (Figure 6.1, upper row). If, however, the object is inhomogeneous, then the matrix will include the new, refined, grid values of $\eta(\mathbf{x}_n)$, $n = 1, \ldots, N$ (Figure 6.1, middle and lower rows). Obviously, there is a link between the spatial discretization and the quantization of constitutive parameters, and we have observed an interesting phenomenon related to these two processes in our numerical experiments. Namely, if we refine the spatial discretization while keeping constitutive parameters constant, then the (low-frequency) spectrum converges to a set of very dense line segments connecting the real unit and the corresponding values of η/η_{b}. In this way we seem to model an object consisting of one or more homogeneous parts without really telling what the behavior of the constitutive parameter is across the interfaces. If, on the other hand, the quantization of the constitutive parameter results in new values at a finer level of discretization, then the line segments in the spectrum are less pronounced.

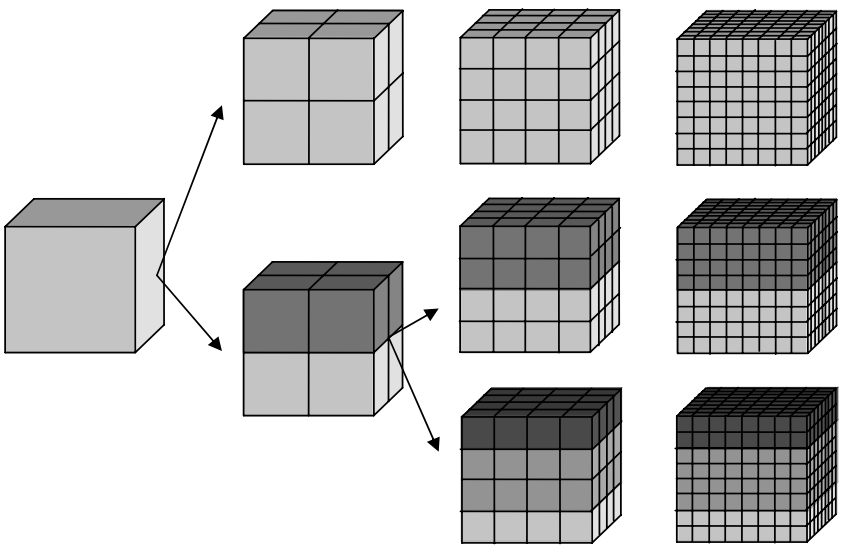


FIG. 6.1. *Different discretization (and quantization) levels for three cubes (homogeneous, two-layered, and three-layered). The side length of all cubes is* $a \approx \lambda_{\text{med}}/20$. *Inhomogeneity becomes "visible" only at an appropriate discretization level.*

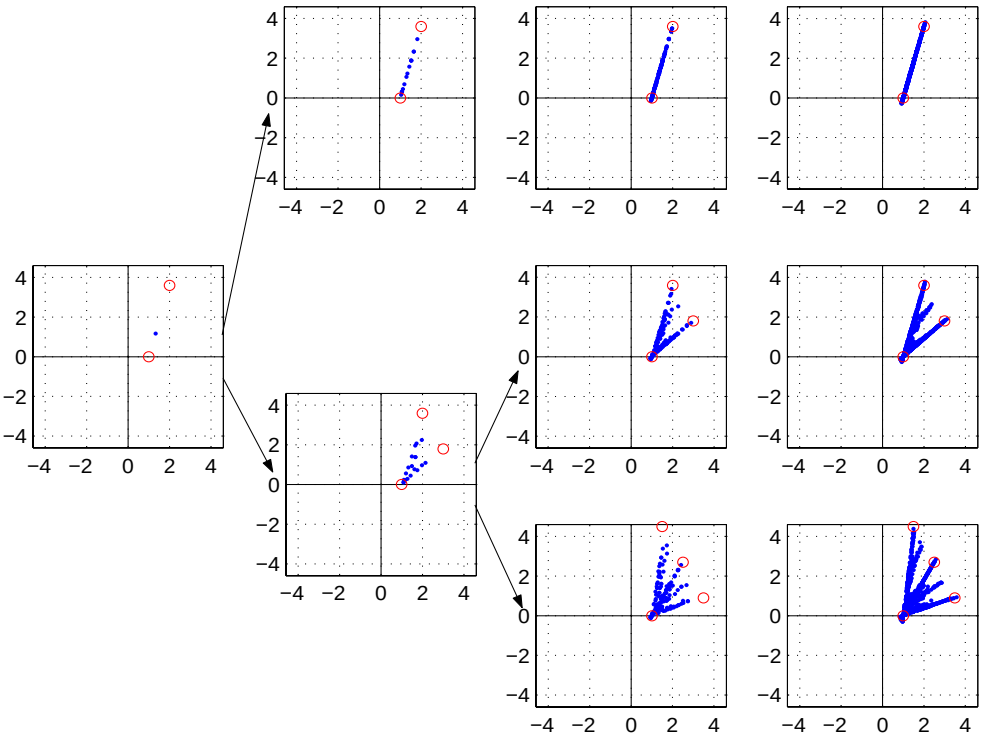


FIG. 6.2. *Effect of discretization (and quantization) refinement on the matrix eigenvalues in the low-frequency (small object) regime. Plots correspond to the objects shown in Figure* 6.1, *and constitutive parameters* η/η_{b} *are given as circles.*

Consider, for example, the case of three different cubes having the same size and the following sequence of grid steps: $h = a, a/2, a/4, a/8$. The matrix dimensions are then, correspondingly, 3×3, 24×24, 192×192, and 384×384. The objects and the eigenvalues are presented, respectively, in Figures 6.1 and 6.2. The upper row in both figures corresponds to the model of a completely homogeneous cube, the middle row models a cube consisting of two layers, and the bottom row corresponds to a three-layered cube. The layered structure becomes visible at $h = a/2$ (second column), and the difference between the two and the three layers is only seen at $h = a/4$ (third column). At $h = a/8$ (last column) we had no more changes in the constitutive parameters of all three objects. The constitutive parameters η/η_{b} are shown as circles in Figure 6.2. The dense lines appear as soon as we stop refining the quantization, but keep refining the spatial discretization.

Figure 6.3 illustrates the marginal influence of the object's geometry on the distribution of eigenvalues at low frequencies (the same frequency as before). Both objects (top and bottom row) consist of two homogeneous parts with constitutive parameters given by circles and have the same outer shape (long parallelepiped, roughly $\lambda/5 \times \lambda/42 \times \lambda/42$). The difference between the objects is in the geometry of the parts. The upper row in Figure 6.3 corresponds to the object consisting of two equal parts (roughly $\lambda/5 \times \lambda/42 \times \lambda/85$ each), which divide the parallelepiped [illegible] its longer dimension, whereas the lower row corresponds to the object consisting of two unequal parts (roughly $\lambda/21 \times \lambda/42 \times \lambda/42$ and $\lambda/7 \times \lambda/42 \times \lambda/42$) dividing the parallelepiped

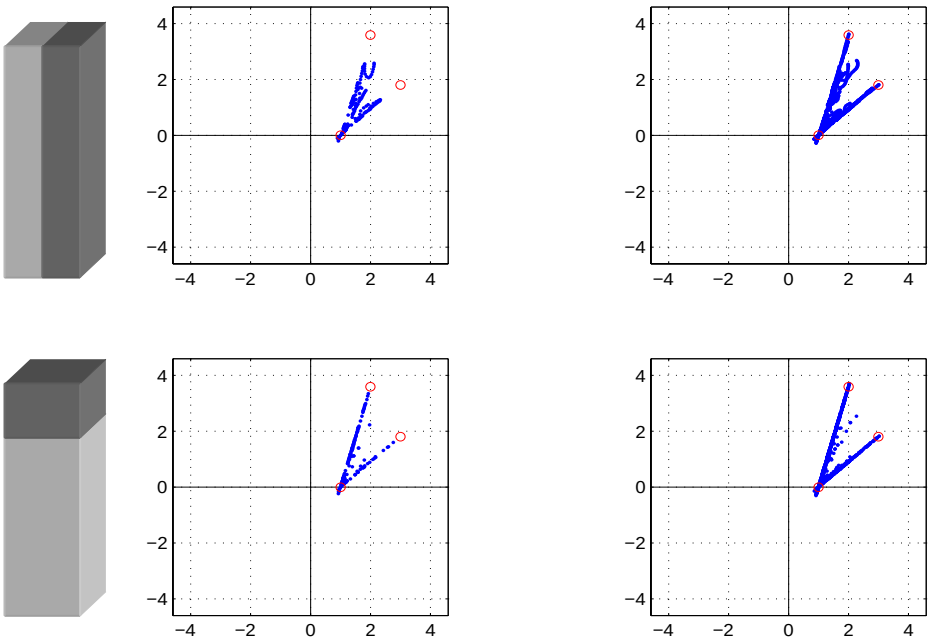


FIG. 6.3. *Effect of the object's shape on the matrix eigenvalues in the low-frequency (small object) regime: Same constitutive parameters* η/η_b *(circles), but different shapes. From left to right: Pictures of the objects, and eigenvalues for two discretization levels:* $h = \lambda_{\text{med}}/70$ *and* $\lambda_{\text{med}}/140$.

its longer dimension. From left to right two discretization levels are presented: $h = \lambda_{\text{med}}/70$ and $h = \lambda_{\text{med}}/140$. With both objects we notice the appearance of the line segments connecting the real unit and the values of η/η_b. The difference seems to be among the other eigenvalues situated between the line segments. These and many other low-frequency numerical experiments suggest that the line segments observed here and in [6] are the matrix analogue of the operator's essential spectrum and that they depend only on the grid values of $\eta(\mathbf{x})/\eta_b$.

Figure 6.4 gives the spectrum for a homogeneous cube at higher frequencies. The η/η_b of the object is shown as a circle. The length of the cube's side is chosen to coincide with the medium wavelength in this case, i.e., $a = \lambda_{\text{med}}$. One can see that now we have not only the dense line segment as in Figure 6.2 (top right), but also a few off-line eigenvalues, which appear within the predicted bounds of Figure 5.1 (right). Two discretizations are shown: $h = \lambda_{\text{med}}/5$ (Figure 6.4, left), and $h = \lambda_{\text{med}}/10$ (Figure 6.4, right). Normally, neither of these would be considered a "proper" level of discretization. However, here we observe an interesting phenomenon related to the essential spectrum, which partly justifies a much more relaxed attitude of practitioners to the discretization of the electromagnetic volume integral equation, as opposed to the tough requirements on the discretization of differential Maxwell's equations via the finite-difference approach. With $h = \lambda_{\text{med}}/5$ we have in total 375 eigenvalues, most of which are clustered along the line segment of the essential spectrum. With $h = \lambda_{\text{med}}/10$ we already have 3000 eigenvalues; however, all "new" 2625 eigenvalues keep filling the same line segment. Of course, the off-line eigenvalues do shift a little, but their number seems to remain the same for both discretization levels. This and similar experiments at higher (resonance) frequencies show that the refinement of

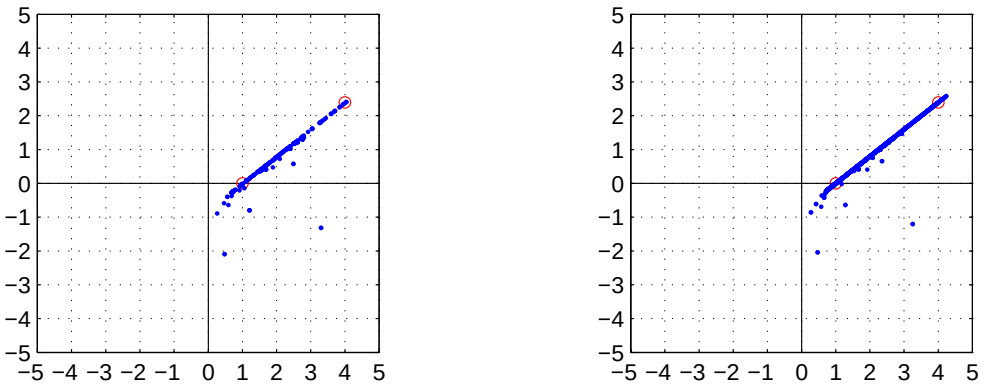


FIG. 6.4. *Effect of discretization on the matrix eigenvalues at higher frequencies. Object: Homogeneous cube. Left: Discretization level* $h = \lambda_{\text{med}}/5$. *Right: Discretization level* $h = \lambda_{\text{med}}/10$.

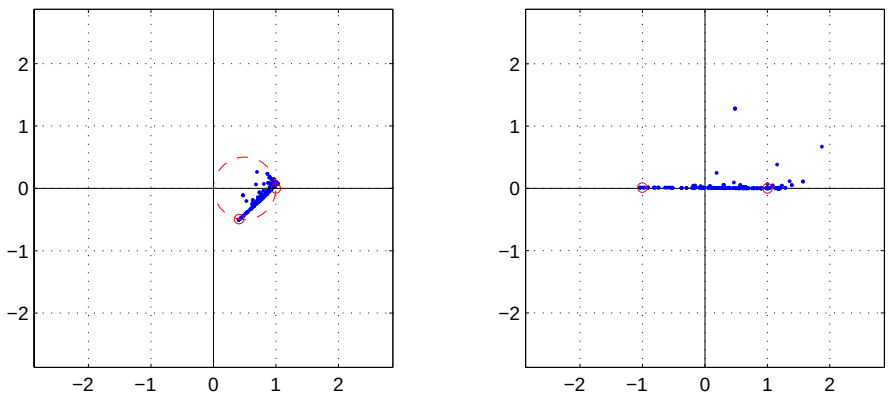


FIG. 6.5. *Spectrum in exotic cases. Left: Vacuum cube in a lossy background medium. Right: Cube with negative permittivity and small losses in vacuum background.*

discretization has no significant effect on the spectral radius of the matrix, which is what is expected from an integral equation formulation.

We conclude our numerical experiments with the result corresponding to the conductive background case described in Figure 5.2 (right). As one can see from Figure 6.5 (left) the matrix spectrum is, indeed, quite accurately described by the predicted circle.

7. Conclusions and applications. The spectrum of the volume integral operator of three-dimensional electromagnetic scattering contains both the essential continuous part and discrete eigenvalues. The apparent difference from the one- and two-dimensional electromagnetics, where the spectrum is purely discrete, stems from the presence of a strong singularity in the kernel of the integral operator in the three-dimensional case. We have shown that the essential spectrum is explicitly given by (5.16) for any anisotropic scatterer with Hölder continuous constitutive parameters.

Knowing the spectrum one can find a fixed iteration parameter which provides the optimal convergence for the Richardson's iteration scheme. In [13] we prove that the well-described essential spectrum dominates at lower frequencies and that all discrete eigenvalues are contained within the convex hull of the essential spectrum. We also

show in [13] how to employ this spectrum information to obtain the mentioned fixed optimal iteration parameter for low-frequency scattering problems.

At higher frequencies one should resort to more sophisticated iterative algorithms such as GMRES and BiCGStab. With large objects and objects with strong electromagnetic contrast GMRES may converge very slowly. In both cases this is probably caused by the increase in the spectral radius of the system matrix. However, now we understand that for the low-contrast large objects this is due to the discrete eigenvalues spreading out, whereas with small but strong-contrast objects the spectral radius will grow due to the stretching of the continuous spectrum. This suggests that two rather distinct preconditioning strategies are needed—an interesting topic for future research.

Information on the spatial spectrum can also be used in the analysis of electromagnetic resonances. Indeed, a resonance occurs at some angular frequency ω if the spatial spectrum either contains zero or is close to it at this frequency. In [14] we show that the spectral information obtained here and in [13] allows us to analyze electromagnetic resonances in arbitrarily inhomogeneous objects of finite extent. To date resonances are explicitly recovered only in infinite (unbounded) homogeneous or periodic media. Somewhat less explicit results can be derived for idealistic metallic and piecewise-homogeneous objects such as resonators, plane-layered media, and dielectric spheres. In [14] we suggest a novel approach to the problem of electromagnetic resonances based on the complex-plane dynamics of the spatial spectrum as a function of the angular frequency. We also show there that the modes associated with the essential spectrum are highly localized in space and are directly related to the so-called plasmons or polaritons.

Another related application area is the study of metamaterials. In particular, one can analyze materials with negative permittivity. Macroscopically these materials may be treated as highly dispersive media, such that for a certain range of ω's their dielectric permittivity and/or their magnetic permeability happen to have negative real parts. An immediate result which follows from our present study is that the spectrum for an object with a negative permittivity part will contain not only points in the left part of the complex plane but also lines or curves connecting these points with the real unit. For a low-loss material, which for obvious reasons is the ultimate goal of experimentalists, the lines of the essential spectrum may then proceed dangerously close to the zero of the complex plane as shown in Figure 6.5 (right), where $\epsilon/\epsilon_{\mathrm{b}} = -1$ and $\sigma = 0.001$ S/m. From the physical viewpoint this would mean that the electromagnetic field is unstable in such media. Without losses ($\sigma = 0$) the line would go right through the zero, meaning that the solution (electromagnetic field) does not exist at all. Perhaps that is the reason why we do not observe many left-handed substances in Nature. Yet another interesting application where the explicit knowledge of the essential spectrum may be of help is magnetically confined fusion plasma.

Acknowledgment. The authors wish to thank Dr. Rob Remis (Delft University of Technology) for his continuous support and encouragement and for pointing out the Cayley–Hamilton theorem.

REFERENCES

[1] R. E. Kleinman, G. F. Roach, and P. M. van den Berg, *Convergent born series for large refractive indices*, J. Opt. Soc. Amer. A, 7 (1990), pp. 890–897.

[2] G. C. Hsiao and R. E. Kleinman, *Mathematical foundations for error estimation in numerical solution of integral equations in electromagnetics*, IEEE Trans. Antennas and Propagation, 45 (1997), pp. 316–328.

[3] M. Gastine, L. Courtois, and J. L. Dormann, *Electromagnetic resonances of free dielectric spheres*, IEEE Trans. Microwave Theory and Techniques, 15 (1967), pp. 694–700.
[4] P. R. Conwell, P. W. Barber, and C. K. Rushforth, *Resonant spectra of dielectric spheres*, J. Opt. Soc. Amer. A, 1 (1984), pp. 62–67.
[5] B. A. Hunter, M. A. Box, and B. Maier, *Resonance structure in weakly absorbing spheres*, J. Opt. Soc. Amer. A, 5 (1988), pp. 1281–1286.
[6] J. Rahola, *On the eigenvalues of the volume integral operator of electromagnetic scattering*, SIAM J. Sci. Comput., 21 (2000), pp. 1740–1754.
[7] S. G. Mikhlin, *Multidimensional Singular Integrals and Integral Equations*, Pergamon Press, New York, 1965.
[8] S. G. Mikhlin and S. Prössdorf, *Singular Integral Operators*, Springer-Verlag, Berlin, 1986.
[9] E. Kreyszig, *Introductory Functional Analysis with Applications*, John Wiley & Sons, New York, 1989.
[10] C. Müller, *Foundations of the Mathematical Theory of Electromagnetic Waves*, Springer-Verlag, Berlin, 1969.
[11] D. Colton and R. Kress, *Inverse Acoustic and Electromagnetic Scattering Theory*, Springer-Verlag, Berlin, 1992.
[12] A. B. Samokhin, *Integral Equations and Iteration Methods in Electromagnetic Scattering*, VSP, Utrecht, The Netherlands, 2001.
[13] N. V. Budko, A. B. Samokhin, and A. A. Samokhin, *A generalized overrelaxation method for solving singular volume integral equations in low-frequency scattering problems*, Differential Equations, 41 (2005), pp. 1262–1266.
[14] N. V. Budko and A. B. Samokhin, *Classification of electromagnetic resonances in finite inhomogeneous three-dimensional structures*, Phys. Rev. Lett., 96 (2006), 023904; also available online at http://arxiv.org/abs/physics/0509259, 2005.

SIAM J. SCI. COMPUT.
Vol. 28, No. 2, pp. 701–727
© 2006 Society for Industrial and Applied Mathematics

SPACE-TIME APPROXIMATION WITH SPARSE GRIDS*

MICHAEL GRIEBEL†, DANIEL OELTZ†, AND PANAYOT VASSILEVSKI‡

Abstract. In this article we introduce approximation spaces, especially suited for the approximation of solutions of parabolic problems, which are based on the tensor product construction of a multiscale basis in space and a multiscale basis in time. Proper truncation then leads to so-called *space-time sparse grid spaces*. For a uniform discretization of the spatial space of dimension d with $O(N^d)$ degrees of freedom, these spaces involve for $d > 1$ also only $O(N^d)$ degrees of freedom for the discretization of the whole space-time problem. But they provide the same approximation rate as classical space-time finite element spaces which need $O(N^{d+1})$ degrees of freedoms. This makes these approximation spaces well suited for conventional parabolic and time-dependent optimization problems. We analyze the approximation properties and the dimension of these sparse grid space-time spaces for general stable multiscale bases. We then restrict ourselves to an interpolatory multiscale basis, i.e., a hierarchical basis. Here, to be able to handle also complicated spatial domains Ω, we construct the hierarchical basis from a given spatial finite element basis as follows: First we determine coarse grid points recursively over the levels by the coarsening step of the algebraic multigrid method. Then, we derive interpolatory prolongation operators between the respective coarse and fine grid points by a least squares approach. This way we obtain an algebraic hierarchical basis for the spatial domain which we then use in our space-time sparse grid approach. We give numerical results on the convergence rate of the interpolation error of these spaces for various space-time problems with two spatial dimensions. Implementational issues, data structures, and questions of adaptivity also are addressed to some extent.

Key words. parabolic differential equations, multilevel basis, sparse grids, space-time discretization

AMS subject classifications. 35K20, 65M99, 65Y20

DOI. 10.1137/050629252

1. Introduction.

The modeling of various phenomena in physics, chemistry, biology, and financial engineering leads to time-dependent partial differential equations (PDEs). In most cases there is no analytical solution available for these equations and hence they have to be solved approximately by an appropriate discretization scheme. If we employ, depending on the smoothness of the solution, a discretization scheme on a uniform grid in time and a uniform grid in d-dimensional space with an error of the order p in time and order q in space, we need $O(N^d)$ degrees of freedom for each time step and $O(N^{d+q/p})$ degrees of freedom for the whole approximate solution in space-time to obtain an $_{l'}$ $_{,,,}$ order of q. In particular, for a method which is second order in space and in time this would result in $O(N^{d+1})$ degrees of freedom. An approach which is only first order in time, e.g., the Euler methods, but second order in space needs $O(N^{d+2})$ degrees of freedom. The associated additional storage

*Received by the editors April 15, 2005; accepted for publication (in revised form) December 15, 2005; published electronically May 26, 2006. This work was performed under the auspices of the U. S. Department of Energy by University of California Lawrence Livermore National Laboratory under contract W-7405-Eng-48. Accordingly, the U.S. Government retains a nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or allow others to do so, for U.S. Government purposes. Copyright is owned by SIAM to the extent not limited by these rights.

http://www.siam.org/journals/sisc/28-2/62925.html

†Sonderforschungsbereich 611, Singuläre Phänomene und Skalierung in mathematischen Modellen, Institut für Numerische Simulation, Universität Bonn, Wegelerstr. 6, D-53115 Bonn, Germany (griebel@ins.uni-bonn.de, oeltz@ins.uni-bonn.de).

‡Center for Applied Scientific Computing, Lawrence Livermore National Laboratory, P.O. Box 808, Mail Stop L-560, Livermore, CA 94551 (panayot@llnl.gov).

requirements then fail to work for parabolic PDEs directly in space-time if $d = 3$ or larger, but enforce a time-slice approach, where only a few spatial grids at successive time points need to be stored.

In some applications it is necessary to solve a system of parabolic equations, where some of the equations are forward and the others are backward in time. Such systems typically arise in parabolic optimal control problems where the equations for the adjoint variables are backward in time but also involve the state variables for which forward equations have to be solved; see, e.g., [33]. Here, all values of the state variables in space $\cdot\, /$ time must be stored to be used for the solution of the adjoint. This is not computationally feasible for the case $d \geq 3$, however, due to the complexity of $O(N^{d+1})$ or $O(N^{d+2})$, even if N is moderately small.

One approach to overcoming this storage problem is the so-called reduced order method [34, 36]. Here, one tries to compute the most important spatial additional storage of the instationary solution from snapshots of the solution at different times. This results in (hopefully) only few spatial basis functions which approximate the given solution with sufficient accuracy over time. To this end, the instationary problem must, however, be approximately solved at the beginning to obtain the snapshots at certain time instances at all. Also, for any update of the reduced order model during an optimization it is necessary to compute an approximate solution to the actual full instationary problem again. Due to the above mentioned complexity orders this is still very expensive. Furthermore, the choice of the time points at which snapshots are taken strongly influences the reduced order model. To date, the question of how to choose the amount and the locations of the time points for the snapshots is not finally answered, and these topics are still a subject of current research.

Another approach is the so-called checkpointing [32]. Here, the solution of the state equation is stored only at some properly chosen time points, called checkpoints. Now, if the value of the state equation at a certain time is needed, the forward state equation is solved with the start time of the nearest checkpoint and the state variable at this point as initial value. Clearly, checkpointing leads to a further increase of computational time on the already expensive solution of the instationary problem.

In the present article we introduce new space-time sparse grid spaces which allow for complexity reduction of the space-time problem from $O(N^{d+q/p})$ to $O(N^d)$. If a slightly more involved regularity assumption is fulfilled, namely, if the mixed second derivative in space and time is bounded, the approximation rate for the space-time sparse grid space remains basically the same as for classical finite element spaces in space and time. Thus, the additional complexity stemming from the time part of a parabolic PDE is eliminated, and we obtain the same complexity order as for a stationary problem. Consequently, the memory and storage requirements are substantially reduced. These space-time sparse grid spaces are constructed from a d-dimensional multilevel basis in space and a one-dimensional multilevel basis in time via a tensor product approach. In this sense they are similar to classical sparse grid spaces; see, for example, [12], which are constructed via a tensor product approach from a one-dimensional multilevel basis for each coordinate direction. We, however, only use a d-dimensional isotropic multilevel basis in spatial space. For this purpose, any available stable multilevel basis in space can be employed. For ease of presentation, we restrict ourselves to an interpolatory multilevel basis, namely a hierarchical basis. To construct the necessary spatial multilevel basis from a given general finite element basis we proceed as follows: First we determine coarse grid points recursively over the levels by the coarsening step of an algebraic multigrid method (AMG). Then, to

assign hierarchical basis functions to these points, we use interpolatory prolongation operators between the respective coarse and fine grids which are derived by a least squares approach. We need higher accuracy of the prolongation operators than in the classical AMG methods in order to guarantee the desired approximation property of the hierarchical basis. In particular, by the least squares approach we employed, the linear functions are well represented on coarse levels. Note that the classical AMG methods have only a "weak approximation property" guaranteed; namely, only the constant function is well represented on coarse levels. Alternatives to the least squares approach could be the smoothed aggregation AMG [47] or the multiple vector preserving AMG interpolation proposed in [48]. These are not considered in the present paper but are worth pursuing in the future. The AMG approach in general allows us to employ any given finite element package on the finest level of the spatial discretization. Thus, we are able to deal also with complicated spatial domains Ω, where classical sparse grid spaces run into difficulties.

In the context of instationary control problems, these spaces can be used for an interpolant of the state variable which was solved with a classical discretization scheme. Then this cheap space-time sparse grid interpolant is employed in the adjoint equation. Alternatively, these spaces can be directly used for the discretization of both the state problem and the adjoint problem. Altogether, our approach overcomes the storage complexity in time-dependent two- and three-dimensional optimization problems and allows us to deal with quite complicated spatial geometries.

The remainder of this article is organized as follows.

In section 2 we describe the general construction of the proposed space-time sparse grid spaces from a multilevel basis in space and a multilevel basis in time. To this end, we simply use a tensor product and properly truncate the resulting multivariate series expansion. We then present results on the approximation rates for these spaces. Here, provided that a slightly stronger regularity assumption is fulfilled, it turns out that the approximation order of classical full grid spaces in space-time is preserved, while the overall complexity is significantly reduced.

In section 3 we discuss how a spatial hierarchical basis can be derived from a given nodal basis and a given sequence of prolongation operators. Here we use the classical coarsening scheme from AMG methods to obtain coarser grids and a least squares approach to compute the necessary interpolation operators.

In section 4 we address some implementational issues that arise for the proposed space-time sparse grid approach. We first describe the necessary operations which must be performed on the data to compute the space-time sparse grid interpolant for a given function. These insights are then used as guidelines for the development of efficient data structures for space-time sparse grids. Furthermore, we discuss additional requirements to take into account adaptivity in space and time.

In section 5 we consider numerical experiments with the proposed space-time sparse grid space. Here, we focus on the interpolation error. The results show that the approximation rates with respect to the L^2-norm deteriorate only slightly in comparison to those of classical finite element spaces in space and time. For the H^1-norm even the same rates are achieved. In addition, the results for an adaptive space-time sparse grid applied to a nonsmooth function indicate that singular functions also can be dealt with in an efficient manner.

Finally, in section 6 we draw some conclusions.

2. Space-time sparse grids. In this section we describe the general construction of space-time sparse grid spaces following the presentation in [29]. We discuss

their approximation properties as well as their cost complexity. It will turn out that these spaces provide the same approximation rate as classical finite element spaces in space-time under just slightly stronger regularity assumptions. The number of involved degrees of freedom, however, is significantly reduced, i.e., by a factor of $O(N^{q/p})$, where p denotes the order of the associated time discretization and q denotes the order of the space discretization, respectively.

For a d-dimensional spatial domain $\Omega \subset \mathbb{R}^d$ and $T > 0$ we define $\Omega_T := \Omega \times (0,T)$. In the following, let $H^m(\Omega)$, $m \in \mathbb{N}$, denote the usual Sobolev space on Ω with weak derivatives of order m. Now, for $m \in \mathbb{N}$, we define

$$H^{2m,m}(\Omega_T) := \big(H^{2m}(\Omega) \otimes L^2((0,T))\big) \cap \big(L^2(\Omega) \otimes H^m((0,T))\big) \tag{2.1}$$

with associated norm

$$\|u\|^2_{H^{2m,m}(\Omega_T)} := \sum_{2\cdot p + \|\mathbf{q}\|_1 \le 2m} \|\partial_t^p \partial_x^{\mathbf{q}} u\|^2_{L^2(\Omega_T)}. \tag{2.2}$$

Here and in the following we use boldface letters to denote multi-indices, i.e., $\mathbf{q} \in \mathbb{N}^d$. Function spaces with bounded mixed derivatives play an important role in the theoretical analysis of sparse grids; see [12, 44, 45]. For our space-time setting we define

$$H^{2m,m}_{mix}(\Omega_T) := H^{2m}(\Omega) \otimes H^m((0,T)) \tag{2.3}$$

with associated norm

$$\|u\|^2_{H^{2m,m}_{mix}(\Omega_T)} := \sum_{\substack{p \le m,\\ \|\mathbf{q}\|_1 \le 2m}} \|\partial_t^p \partial_x^{\mathbf{q}} u\|^2_{L^2(\Omega_T)}, \tag{2.4}$$

and

$$H^{m,m}_{mix}(\Omega_T) := H^m(\Omega) \otimes H^m((0,T)) \tag{2.5}$$

with associated norm

$$\|u\|^2_{H^{m,m}_{mix}(\Omega_T)} := \sum_{\substack{p \le m,\\ \|\mathbf{q}\|_1 \le m}} \|\partial_t^p \partial_x^{\mathbf{q}} u\|^2_{L^2(\Omega_T)}. \tag{2.6}$$

Note that the space $H^{2m,m}_{mix}(\Omega_T)$ has a structure similar to that of the space $H^{2m,m}$ which is usually employed in regularity estimates of parabolic problems; cf. [37]. Under regularity assumptions for the initial condition and right-hand side that are only slightly stronger than in classical regularity theory, the solution of the Cauchy problem can actually be shown to be in $H^{2m,m}_{mix}(\Omega_T)$; see [29] for details.

In the following we want to construct approximation spaces for the approximation of functions $u \in H^{2m,m}_{mix}(\Omega_T)$ and $u \in H^{m,m}_{mix}(\Omega_T)$, respectively. They will possess a substantially reduced number of overall degrees of freedom compared to a conventional finite element space on Ω_T without a change in the approximation order. Let us assume that there are finite-dimensional spaces V_j^Ω and V_j^T, $j \in \mathbb{N}$, such that

$$L^2(\Omega) = \sum_{j \ge 0} V_j^\Omega,$$
$$L^2((0,T)) = \sum_{j \ge 0} V_j^T,$$

and $V_j^\Omega \subset V_{j+1}^\Omega$, $V_j^T \subset V_{j+1}^T$. Furthermore, we define increment spaces W_j^Ω and W_j^T such that

$$V_j^\Omega = V_{j-1}^\Omega \oplus W_j^\Omega,\ j \geq 1,$$
$$V_j^T = V_{j-1}^T \oplus W_j^T,\ j \geq 1,$$

and $W_0^\Omega := V_0^\Omega$, $W_0^T := V_0^T$. Since $L^2(\Omega_T) = L^2(\Omega) \otimes L^2((0,T))$ (cf. [28]), we obtain

$$L^2(\Omega_T) = \bigoplus_{\mathbf{j}\in\mathbb{N}^2} V_{\mathbf{j}}, \quad V_{\mathbf{j}} = V_{j_1}^\Omega \otimes V_{j_2}^T.$$

In the following we identify the tensor product $\varphi^\Omega \otimes \varphi^T$ of two functions $\varphi^\Omega(x) \in L^2(\Omega)$ and $\varphi^T(t) \in L^2((0,T))$ with $\varphi^\Omega \cdot \varphi^T$. We define the hierarchical increment spaces $W_{\mathbf{j}}$, $\mathbf{j} \in \mathbb{N}^2$, as

$$W_{\mathbf{j}} = W_{(j_1,j_2)} := W_{j_1}^\Omega \otimes W_{j_2}^T. \tag{2.7}$$

To simplify notation we omit the domain of integration in the norms; e.g., we write $\|\cdot\|_{L^2}$ instead of $\|\cdot\|_{L^2(\Omega)}$. If not stated otherwise, we assume that the hierarchical increments W_j^T and W_j^Ω are L^2-stable; i.e.,[1]

$$\left\|\sum_j w_j\right\|_{L^2(\Omega)}^2 \simeq \sum_j \|w_j\|_{L^2(\Omega)}^2 \quad \forall \sum_j w_j \in L^2(\Omega),\ w_j \in W_j^\Omega,$$
$$\left\|\sum_j w_j\right\|_{L^2((0,T))}^2 \simeq \sum_j \|w_j\|_{L^2((0,T))}^2 \quad \forall \sum_j w_j \in L^2((0,T)),\ w_j \in W_j^T.$$

With this notation, we define the full grid spaces

$$V_\ell^\infty := \bigoplus_{\substack{2j_1\leq 2\ell,\\ j_2\leq 2\ell}} W_{(j_1,j_2)} \tag{2.8}$$

and

$$\tilde{V}_\ell^\infty := \bigoplus_{\|\mathbf{j}\|_\infty \leq \ell} W_{(j_1,j_2)} \tag{2.9}$$

as well as the sparse grid spaces

$$V_\ell^1 := \bigoplus_{2j_1+j_2\leq 2\ell} W_{(j_1,j_2)} \tag{2.10}$$

and

$$\tilde{V}_\ell^1 := \bigoplus_{\|\mathbf{j}\|_1 \leq \ell} W_{\mathbf{j}}. \tag{2.11}$$

Note that the space-time sparse grid spaces V_ℓ^1 and $\tilde{V}_\ell^1$ are similar to classical sparse grid spaces; see [10, 12, 28]. Now, however, the supports of the basis functions are

[1] Here, $A \simeq B \Leftrightarrow c_1 B \leq A \leq c_2 B$ with positive constants c_1, c_2.

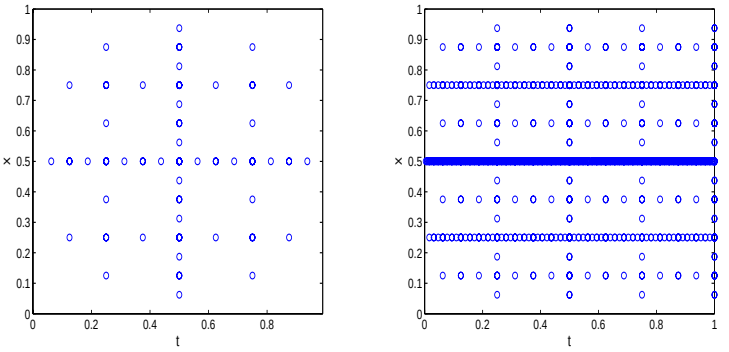


FIG. 2.1. *Grid points for the space* $\tilde{V}_4^1$ *(left) and* V_4^1 *(right) for the one-dimensional domain* $\Omega = (0,1)$ *and the time interval* $(0,1)$.

anisotropic with respect to space versus time but are isotropic in space. This results from our construction (2.7) which involves a tensor product between an (isotropic) multilevel basis in space and a one-dimensional multilevel basis in time, which is in contrast to the conventional sparse grid approach, where the basis in space is formed by a tensor product of a one-dimensional multiscale basis for each coordinate direction. Furthermore, the space V_ℓ^1 is twice as refined in the time direction as in the space direction. This can also be seen in Figure 2.1 (right) where sparse grids of level $\ell = 4$ on the domain $\Omega_T = (0,1)^2$ are shown. To this end, the one-dimensional piecewise linear hierarchical basis [22, 49, 50] was used for both space and time within the tensor product construction. The sparse grid points are the center points of the supports of the associated basis functions used for the respective spaces. Here we see that V_ℓ^1 involves more grid points in time than in the space direction, which results in a higher overall number of degrees of freedom than for $\tilde{V}_\ell^1$. As we will see later in Lemma 2.3, the sparse grid V_ℓ^1 is more suited to problems with lower regularity in time than in space, whereas $\tilde{V}_\ell^1$ is more efficient for problems with the same regularity in space and time.

To derive estimates of the approximation rates of the different spaces, we need the following lemma.

LEMMA 2.1. [illegible] $t \geq 0$ [illegible] $v^\Omega \in H^{2t}(\Omega)$ $v^\Omega = \sum_j w_j^\Omega$ $w_j^\Omega \in W_j^\Omega$

$$\|v^\Omega\|_{H^{2t}}^2 \simeq \sum_j 2^{2(2t)j} \|w_j^\Omega\|_{L^2}^2 \tag{2.12}$$

[illegible] $v^T \in H^t((0,T))$ $v^T = \sum_j w_j^T$ $w_j^T \in W_j^T$

$$\|v^T\|_{H^t}^2 \simeq \sum_j 2^{2tj} \|w_j^T\|_{L^2}^2. \tag{2.13}$$

[illegible] $u \in H^{2t,t}_{mix}(\Omega_T)$ $u = \sum_{\mathbf{j}} w_{\mathbf{j}}$ [illegible]

$$\|u\|_{H^{2t,t}_{mix}(\Omega_T)}^2 \simeq \sum_{\mathbf{j}} 2^{2t\|(2j_1,j_2)\|_1} \|w_{\mathbf{j}}\|_{L^2(\Omega_T)}^2. \tag{2.14}$$

(2.12)

$$\|v^{\Omega}\|_{H^t}^2 \simeq \sum_j 2^{2tj} \|w_j^{\Omega}\|_{L^2}^2, \tag{2.15}$$

(2.13) $u \in H^{t,t}_{mix}(\Omega_T)$ $u = \sum_{\mathbf{j}} w_{\mathbf{j}}$

$$\|u\|^2_{H^{t,t}_{mix}(\Omega_T)} \simeq \sum_{\mathbf{j}} 2^{2t\|(j_1,j_2)\|_1} \|w_{\mathbf{j}}\|^2_{L^2(\Omega_T)}. \tag{2.16}$$

The detailed proof of these norm equivalences can be found in [29]. Here, we restrict ourselves to (2.14) to show that the estimates follow directly from a proposition about additive subspace splittings taken from [30]. To this end we use the notation $\{V; a\}$ to denote a Hilbert space V which is equipped with the inner product $a(\cdot,\cdot)$. We now consider two Hilbert spaces H_1 and H_2 and a collection of closed subspaces $V_{1i} \subset H_1$ and $V_{2i} \subset H_2$, $i \in \mathbb{N}$, such that topologically

$$H_1 = \sum_i V_{1i} \quad \text{and} \quad H_2 = \sum_i V_{2i}.$$

For a sequence $b_{ki}(\cdot,\cdot)$ of bilinear forms on V_{ki}, $k \in \{1,2\}$, we call $\{H_k; a_k\} = \sum_i \{V_{ki}; b_k\}$ an additive subspace splitting which is stable if

$$a_k(u,u) \simeq |||u|||^2 \equiv \inf_{\substack{u_i \in V_{ki},\\ u=\sum_i u_i}} \sum_i b_{ki}(u_i,u_i) \tag{2.17}$$

holds true. From [30] we have the following proposition.

PROPOSITION 2.2. $\{H_k; a_k\} = \sum_i \{V_{ki}; b_{ki}\}$ $k = 1, 2$

$$\{H_1 \otimes H_2; a_1 \otimes a_2\} = \sum_{i_1} \sum_{i_2} \{V_{1i_1} \otimes V_{2i_2}; b_{1i_1} \otimes b_{2i_2}\}$$

Now we can start the proof of Lemma 2.1.

Here, we consider only (2.14); the other norm equivalences can be derived analogously. For $u \in H^{2t,t}_{mix}(\Omega_T)$, we obtain with Proposition 2.2 the stability of

$$\{H^{2t}(\Omega) \otimes H^t((0,T)); \|\cdot\|_{H^{2t}} \otimes \|\cdot\|_{H^t}\} = \sum_{\mathbf{j}} \{W_{\mathbf{j}}; 2^{2t\|(2j_1,j_2)\|_1} \|\cdot\|_{L^2} \otimes \|\cdot\|_{L^2}\}. \qquad \square$$

Now, we are in a position to state the approximation properties of the spaces $V_\ell^\infty, V_\ell^1, \tilde{V}_\ell^\infty$, and $\tilde{V}_\ell^1$.

LEMMA 2.3. (2.12) (2.13) s, t $t > s$ $u \in H^{2t,t}(\Omega_T)$

$$\inf_{v \in V_\ell^\infty} \|u - v\|^2_{H^{2s,s}(\Omega_T)} \le c \cdot 2^{4(s-t)\ell} \|u\|^2_{H^{2t,t}(\Omega_T)}. \tag{2.18}$$

$u \in H^{2t,t}_{mix}(\Omega_T)$

$$\inf_{v \in V_\ell^1} \|u - v\|^2_{H^{2s,s}(\Omega_T)} \le c \cdot 2^{4(s-t)\ell} \|u\|^2_{H^{2t,t}_{mix}(\Omega_T)} \tag{2.19}$$

... $u \in H^{t,t}(\Omega_T)$ ... (2.15) ... (2.13) ... s, t $t > s$ ...

$$\inf_{v \in \tilde{V}_\ell^\infty} \|u - v\|^2_{H^{s,s}(\Omega_T)} \leq c \cdot 2^{2(s-t)\ell} \|u\|^2_{H^{t,t}(\Omega_T)}, \tag{2.20}$$

... $u \in H^{t,t}_{mix}(\Omega_T)$ ...

$$\inf_{v \in \tilde{V}_\ell^1} \|u - v\|^2_{H^{s,s}(\Omega_T)} \leq c \cdot 2^{2(s-t)\ell} \|u\|^2_{H^{t,t}_{mix}(\Omega_T)}. \tag{2.21}$$

... We consider only the case (2.19). The other estimates follow in an analogous way. For $u \in H^{2t,t}_{mix}(\Omega_T)$, $u = \sum_{\mathbf{j}} w_{\mathbf{j}}$, $w_{\mathbf{j}} \in W_{\mathbf{j}}$, using the norm equivalences (2.12) and (2.13) we obtain

$$\begin{aligned}
\inf_{v \in V_\ell^1} \|u - v\|^2_{H^{2s,s}(\Omega_T)} &\leq \left\| u - \sum_{2j_1 + j_2 \leq 2\ell} w_{\mathbf{j}} \right\|^2_{H^{2s,s}(\Omega_T)} \\
&\leq c \cdot \sum_{2j_1 + j_2 > 2\ell} 2^{2s\|(2j_1, j_2)\|_\infty} \|w_{\mathbf{j}}\|^2_{L^2(\Omega_T)} \\
&= c \cdot \sum_{2j_1 + j_2 > 2\ell} 2^{2s\|(2j_1, j_2)\|_\infty - 2t\|(2j_1, j_2)\|_1} 2^{2t\|(2j_1, j_2)\|_1} \|w_{\mathbf{j}}\|^2_{L^2(\Omega_T)} \\
&\leq c \cdot \max_{2j_1 + j_2 > 2\ell} 2^{2s\|(2j_1, j_2)\|_\infty - 2t\|(2j_1, j_2)\|_1} \|u\|^2_{H^{2t,t}_{mix}(\Omega_T)} \\
&\leq c \cdot 2^{4(s-t)\ell} \|u\|^2_{H^{2t,t}_{mix}(\Omega_T)}. \qquad \square
\end{aligned}$$

The above lemma shows that the sparse grid spaces constructed with any multilevel basis which fulfills the norm equivalency (2.14) provide the same approximation rates for functions from $H^{2t,t}_{mix}(\Omega_T)$ as the full grid spaces. For a simple domain Ω which is (e.g., after a suitable differentiable mapping) a d-cube, there exist a variety of such bases like orthogonal wavelets, spline-wavelets, prewavelets, biorthogonal wavelets, lifting wavelets, and similar constructions which are constructed from a mother function by translation and dilation; see [13, 15, 19, 21, 43] and the references cited therein. Also a multilevel generating system, i.e., the collection of finite element bases for $V_j^\Omega, j \geq 0$, results in an estimate similar to (2.12) or (2.15).[2] The same holds for a domain Ω which is composed of simple nonoverlapping subdomains. Here, a wavelet-type basis or a multilevel basis is employed within each subdomain.[3] However, for a more complicated, general spatial domain Ω, the construction of a

[2] Here, the nonuniqueness of the representation $v^\Omega = \sum_j v_j^\Omega, v_j^\Omega \in V_j^\Omega$, is compensated by taking into account the prewavelets of all possible multilevel representations; i.e., instead of (2.15) we would have

$$\|v^\Omega\|^2_{H^t} \simeq \inf_{\substack{v^\Omega = \sum_j v_j^\Omega, \\ v_j^\Omega \in V_j^\Omega}} \sum_j 2^{2tj} \|w_j^\Omega\|^2_{L^2}. \tag{2.22}$$

[3] In these cases sparse grids can also be applied for the spatial discretization. This results in further substantial savings in cost provided that an additional smoothness prerequisite in space such as the boundedness of the second mixed derivatives is fulfilled. For details see the survey article [12] and the references cited therein.

multilevel basis which fulfills norm equivalencies like (2.12) or (2.15) can be quite difficult and demanding or even impossible. For wavelet-like approaches on polygonal domains, see [16, 20, 42, 41]. In view of the approximation properties, however, it is not really necessary that such norm equivalences are fulfilled. As the next lemma shows, we will need only upper bounds for the hierarchic increments $w_{\mathbf{j}} \in W_{\mathbf{j}}$ of a function $u = \sum_{\mathbf{j}} w_{\mathbf{j}}$ to obtain nearly the same rates of approximation.

LEMMA 2.4. [illegible] $t > s \geq 0$ [illegible] $c > 0$ [illegible]

$$\|w_{\mathbf{j}}\|_{H^{s,s}(\Omega_T)} \leq c 2^{s\|\mathbf{j}\|_\infty - t\|\mathbf{j}\|_1} \|u\|_{H^{t,t}_{mix}(\Omega_T)} \quad \forall \mathbf{j} \in \mathbb{N} \tag{2.23}$$

[illegible] $u \in H^{t,t}_{mix}(\Omega_T)$ $u = \sum_{\mathbf{j}} w_{\mathbf{j}}$ [illegible]

$$\inf_{v \in \tilde{V}_\ell^1} \|u - v\|^2_{H^{s,s}(\Omega_T)} \leq \begin{cases} c \cdot 2^{2(s-t)\ell} \cdot \ell^2 \|u\|^2_{H^{t,t}_{mix}(\Omega_T)} & \text{[illegible]} \ s = 0, t > 0, \\ c \cdot 2^{2(s-t)\ell} \|u\|^2_{H^{t,t}_{mix}(\Omega_T)} & \text{[illegible]} \ s > 0, t > 0. \end{cases} \tag{2.24}$$

This lemma is a simple generalization of Theorem 4 in [28].

The above lemma shows that it is possible to replace the norm equivalences (2.12) or (2.15) by simple upper bounds for the hierarchic increments of the type (2.23) while maintaining nearly the same approximation rates. As we will see in section 3 such upper bounds can easily be derived by tensor product arguments.

We are now interested in the dimension of the spaces $V_\ell^\infty, V_\ell^1, \tilde{V}_\ell^\infty$, and $\tilde{V}_\ell^1$.

LEMMA 2.5. [illegible] $\dim(W_j^\Omega) = O(2^{d \cdot j})$ [illegible] $\dim(W_j^T) = O(2^j)$ [illegible]

$$\dim(V_\ell^\infty) = O(2^{(d+2)\cdot \ell}), \tag{2.25}$$

$$\dim(V_\ell^1) = \begin{cases} O(2^{2d\ell}) & \text{[illegible]} \ d = 1, \\ O(2^{d\ell}\ell) & \text{[illegible]} \ d = 2, \\ O(2^{d\cdot\ell}) & \text{[illegible]} \ d > 2, \end{cases} \tag{2.26}$$

$$\dim(\tilde{V}_\ell^\infty) = O(2^{(d+1)\cdot \ell}), \tag{2.27}$$

$$\dim(\tilde{V}_\ell^1) = \begin{cases} O(2^\ell \ell) & \text{[illegible]} \ d = 1, \\ O(2^{d\cdot\ell}) & \text{[illegible]} \ d > 1. \end{cases} \tag{2.28}$$

[illegible] Since the estimates (2.25) and (2.27) for the full grid spaces are well known and the derivation of (2.26) can be found in [29], we will discuss only (2.28). We have

$$\begin{aligned} \tilde{V}_\ell^1 &= \bigoplus_{\|\mathbf{j}\|_1 \leq \ell} W_{\mathbf{j}} = \bigoplus_{j_1 \leq \ell} \bigoplus_{j_2 \leq \ell - j_1} W_{\mathbf{j}} \\ &= \bigoplus_{j_1 \leq \ell} W_{j_1}^T \otimes \left(\bigoplus_{j_2 \leq \ell - j_1} W_{j_2}^\Omega \right) \\ &= \bigoplus_{j_1 \leq \ell} W_{j_1}^T \otimes V_{\ell - j_1}^\Omega. \end{aligned}$$

Therefore, we get

$$\begin{aligned}\dim(\tilde{V}_\ell^1) &= \sum_{j_1=0}^{\ell} \dim(W_{j_1}^T) \cdot \dim(V_{\ell-j_1}^\Omega) \\ &\leq c \sum_{j_1=0}^{\ell} 2^{j_1} \cdot 2^{d\cdot(\ell-j_1)} \\ &= c \cdot 2^{d\cdot\ell} \sum_{j_1=0}^{\ell} 2^{(1-d)j_1},\end{aligned}$$

and (2.28) follows immediately. □

This lemma shows that the sparse grid spaces V_ℓ^1 and $\tilde{V}_\ell^1$ have a lower dimension than the full grid spaces V_ℓ^∞ and $\tilde{V}_\ell^\infty$. The overall complexity is reduced by one and two orders of magnitudes, respectively, where for the cases $d = 1$ and $d \leq 2$ the additional logarithmic factors ℓ and ℓ^{d-1} appear. The numerical results in section 5, furthermore, indicate that $\tilde{V}_\ell^1$ has just about four times as many degrees of freedom as the spatial multilevel basis alone; i.e.,

$$\dim(\tilde{V}_\ell^1) \approx 4 \cdot \dim(V_\ell^\Omega).$$

Taking the approximation results from Lemma 2.3 into account, we see that under some mild additional regularity assumptions we can approximate space-time functions by means of sparse grid spaces with basically the same complexity as functions stemming from stationary problems. The additional order of complexity due to time is no longer present, and therefore the associated problems discussed in the introduction are circumvented. Thus our sparse grid space-time spaces provide a powerful tool for a variety of instationary PDEs and parabolic optimal control problems.

3. A spatial multilevel basis. In this section we discuss the construction of a multilevel splitting and a simple hierarchical basis for the spatial discretization. In view of Lemma 2.4 it is sufficient to use a multilevel basis which just fulfills upper bounds of the type (2.23) within the sparse grid concept to obtain approximation properties similar to those of full grid spaces. To this end, we focus first on the hierarchical basis [49] constructed from a nested sequence of spatial finite element spaces in subsection 3.1. It turns out that, under suitable conditions on the sequence of the spatial finite element spaces, the use of the isotropic linear hierarchical basis in the spatial domain and the linear hierarchical basis in time results in sparse grid spaces which fulfill upper bounds of the type (2.23) for the L^2- and the energy-norm. Therefore, Lemma 2.4 can be applied, and thus estimates for the approximation rates of the resulting space-time sparse grids based on the spatial hierarchical basis are obtained.

But even for the simplest multiscale basis, i.e., the hierarchical basis, there is still the problem of the resolution of the boundary on all scales. The coarsest triangulation determines the domain from which the finer scale triangulations are derived by refinement. To overcome this problem we construct in subsection 3.2 a special spatial hierarchical basis from a finite element space given on the finest scale. Here, we determine so-called coarse grids by purely algebraic coarsening, i.e., following the approach of AMG [26, 39, 46] we use the AMG-coarsening scheme to create sets of coarse grid unknowns. Then, to assign hierarchical basis functions to these points,

we use prolongation operators between the respective coarse and fine grids which are derived from a least squares approach.

3.1. Multilevel subspace splitting, interpolation, and hierarchical basis. Let us assume that we have a sequence of nested triangulations/grids $\mathcal{T}_j$ on the domain Ω, $j \in \mathbb{N}$, $\mathcal{T}_j \subset \mathcal{T}_{j+1}$; i.e., the set of the nodes $\mathcal{N}_j$ of $\mathcal{T}_j$ is a subset of the set of nodes $\mathcal{N}_{j+1}$ of $\mathcal{T}_{j+1}$,

$$\mathcal{N}_j \subset \mathcal{N}_{j+1}.$$

As usual, we denote the diameter of $K \in \mathcal{T}_j$ with $h(K)$ and define

$$\rho(K) := \sup\{\operatorname{diam}(B) \mid B \subset K,\ B \text{ is a ball}\}.$$

Throughout this section we assume that $\mathcal{T}_j$ is a sequence of regular triangulations, i.e., that there is a constant σ such that

$$\frac{h(K)}{\rho(K)} \leq \sigma \ \forall j \text{ and } K \in \mathcal{T}_j,$$

and that there is a constant $c > 0$ such that

$$h_j := \max_{K \in \mathcal{T}_j} h(K) \leq c 2^{-j}.$$

We denote by V_j^Ω the associated finite element space of piecewise linear or d-linear functions, and define a mapping $P_j : C^1(\Omega) \to V_j^\Omega$. Then each $u_\ell \in V_\ell^\Omega$ has the representation

$$u_\ell = P_0 u_\ell + \sum_{j=0}^{\ell-1} (P_{j+1} u_\ell - P_j u_\ell), \tag{3.1}$$

with the increment spaces

$$W_{j+1}^\Omega := \operatorname{range}(P_{j+1} - P_j) \tag{3.2}$$

and $W_0^\Omega := V_0^\Omega$. For example, if we choose $P_j u$ as the interpolant of $u \in C^1(\Omega)$ in V_j^Ω, i.e.,

$$P_j u(x) = u(x) \ \forall x \in \mathcal{N}_j, \tag{3.3}$$

we obtain a splitting into increment spaces which are spanned by the hierarchical basis as described in [49]. For the choice $P_j u$ as the L^2-projection of u onto V_j we would obtain an L^2-orthogonal wavelet basis. Other choices of projection operators are discrete L^2-projections, approximate L^2-projections, quasi-interpolation or Clément-projections, which all lead to a stable multilevel basis.

In the following we focus on the choice of P_j as the interpolation operator (3.3) onto V_j. Since we are dealing with a regular triangulation, the classical interpolation theory of finite element spaces, (cf. [3, 14]) shows that

$$\|u - P_j u\|_{L^2(\Omega)} \leq c \cdot 2^{-2j} \|u\|_{H^2(\Omega)} \tag{3.4}$$

for all $u \in H^2(\Omega)$ with constant $c > 0$ independent of j. Therefore, using the hierarchical decomposition (3.1) we obtain for the parts w_j of the splitting $u = P_0 u + \sum_{j\in\mathbb{N}}(P_{j+1}u_\ell - P_j u_\ell) = \sum_{j\in\mathbb{N}} w_j$, $u \in H^2(\Omega)$, the estimate

$$\|w_j\|_{L^2(\Omega)} \le c \cdot 2^{-2j} \|u\|_{H^2(\Omega)} \tag{3.5}$$

with constant $c > 0$ independent of j. An analogous estimate follows directly for all $u \in H^2((0,T))$ with the linear interpolation operator and thus the linear hierarchical basis in time. Then, arguments on the tensor product of operators [31] lead, for the hierarchical increment spaces $W_{\mathbf{j}} = W_{j_1}^{\Omega} \otimes W_{j_2}^{T}$ with the splitting $u = \sum_{\mathbf{j}\in\mathbb{N}^2} w_{\mathbf{j}}$, to the estimate

$$\|w_{\mathbf{j}}\|_{L^2(\Omega_T)} \le c \cdot 2^{-2\cdot|\mathbf{j}|_1} \|u\|_{H^{2,2}_{mix}(\Omega)} \ \forall \mathbf{j} \in \mathbb{N}^2, \tag{3.6}$$

with c independent of $\mathbf{j}$ and independent of $u \in H^{2,2}_{mix}(\Omega_T)$. Here, $W_{j_2}^T$ are the one-dimensional linear increment spaces in time which result analogously to (3.2) from the differences of piecewise linear interpolation operators in time.

Now we can apply Lemma 2.4 and obtain the following result on the approximation rate of the space-time sparse grid space using a linear hierarchical basis in time and the isotropic d-dimensional hierarchical basis in space.

LEMMA 3.1. [illegible] V_j^{Ω} [illegible] (3.4) [illegible] c [illegible] j [illegible] $u \in H^2(\Omega)$ [illegible] (3.2) [illegible] $\tilde{V}_\ell^1$ [illegible]

$$\inf_{v\in\tilde{V}_\ell^1} \|u - v\|_{L^2(\Omega_T)} \le c \cdot 2^{-2\ell} \cdot \ell \cdot \|u\|_{H^{2,2}_{mix}(\Omega_T)} \tag{3.7}$$

[illegible] $u \in H^{2,2}_{mix}(\Omega_T)$

We define the energy norm $\|\cdot\|_E$ for $u \in H^{1,1}(\Omega_T)$ as usual by

$$\|u\|_E^2 = \int_0^T \int_\Omega (\partial_t u)^2 + \sum_{i=1}^{d} (\partial_{x_i} u)^2 \, dx \, dt.$$

Furthermore, analogous to the definition of $H^{2,2}_{mix}(\Omega_T)$, we now define

$$H^{2,2,\infty}_{mix}(\Omega_T) := H^{2,\infty}(\Omega) \otimes H^{2,\infty}((0,T)),$$

where $H^{2,\infty}(\Omega)$ and $H^{2,\infty}((0,T))$ denote the classical Sobolev spaces of functions $u \in L^\infty(\Omega)$ and $u \in L^\infty((0,T))$ with weak derivatives up to the order 2 which are contained in $L^\infty(\Omega)$, respectively. We then can state the following lemma.

LEMMA 3.2. [illegible] V_k [illegible] $\tilde{V}_\ell^1$ [illegible] (3.2) [illegible]

$$\inf_{v\in\tilde{V}_\ell^1} \|u - v\|_E \le c 2^{-\ell} \|u\|_{H^{2,2,\infty}_{mix}(\Omega_T)} \tag{3.8}$$

[illegible] $u \in H^{2,2,\infty}_{mix}(\Omega_T)$ [illegible] $c > 0$ [illegible] ℓ

For the proof of this lemma, one has to verify condition (2.23). Then the application of Lemma 2.4 leads to the estimate (3.8). The verification of (2.23) is quite technical; we therefore postpone it until the appendix.

The above discussion shows that it is sufficient to have a sequence of grids and linear or d-linear finite element spaces which induce the hierarchical increment spaces in the space-time sparse grid construction to maintain the approximation order of a full grid space in the energy norm. For the L^2-norm only an additional logarithmic factor comes into play. Now, the question is how such a sequence of grids and prolongation operators can be obtained. Quite a few finite element packages, meanwhile, use geometric multigrid methods for the solution of the linear systems which result from the discretization of a PDE. These codes then also provide in one way or another access to interpolation operators P_j which we can exploit.

However, for complicated geometries and unstructured grids it is difficult or even impossible to construct a sequence of nested subspaces and a hierarchy of grids which are needed for geometric multigrid methods. Here, a main problem is the resolution of the domain on coarser levels, which is one of the reasons AMG methods [4, 5, 6, 39] were developed. Here, given a matrix A stemming from a finite element discretization, AMG computes (among other things) a sequence of coarse grids, i.e., a set of unknowns, and a sequence of prolongation operators. Now, taking a closer look at the construction (3.2) of the hierarchical increment spaces W_j^Ω, we find that geometric information about the grids $\mathcal{T}_j$ is not necessary but that only the prolongation operators P_j are needed. Therefore, AMG offers a solution to the problem of the resolution of the geometry. We discuss in the next subsection how we use AMG ingredients to construct a sequence of prolongations which we will use in our space-time sparse grid construction.

3.2. Construction of algebraic prolongation operators. In this subsection we will focus on the construction of a sequence of prolongation operators P_j needed for the construction (3.2) of the hierarchic increment spaces W_j. Here, prolongation operators as well as sets C_j of coarse grid points are algebraically constructed using only the entries of the system matrix A^ℓ on the finest level of discretization.[4]

For most AMG methods, the set-up phase, i.e., the phase where the prolongation operators and the coarse grid operators are computed, can be divided into two parts. In the first step, the coarse grid, i.e., a subset $C_j \subset N_j$ of the set of all unknowns N_j on level j, is chosen. Then, given the set of coarse grid points which form the new set of unknowns of the next coarser level, the prolongation $P_j = (p_{kl}^j)$ is computed. Here, one usually requires that each unknown associated to a fine grid point which is also contained in the coarser grid is interpolated by the value of the respective coarse grid unknown; i.e., after reordering the fine and coarse grid unknowns, the prolongation operator P_j has the block-matrix representation

$$P_j = \begin{pmatrix} I_j \\ \tilde{P}_j \end{pmatrix}, \tag{3.9}$$

$n_j := |N_j| = |C_{j-1}|$, $I_j \in \mathbb{R}^{n_{j-1} \times n_{j-1}}$ is the identity, and $\tilde{P}_j \in \mathbb{R}^{(n_j - n_{j-1}) \times n_{j-1}}$.

For the coarsening process, i.e., the computation of C_j, we use the coarsening algorithm due to Ruge and Stüben as described in [39]. To derive $\tilde{P}_j$ we used in a first

[4] AMG needs to be given a fine grid stiffness matrix A^l to set up the interpolation operators and coarser grid matrices A^j via the Galerkin identity. But in our context we have no unique naturally given stiffness matrix A^l at our disposal. Since we are merely interested in the construction of a multilevel basis for the space part of a space-time sparse grid discretization by means of AMG, we used the stiffness matrix which stems from the discretization of the Laplacian in the given finite element space in the numerical experiments presented in section 5. For this choice AMG methods provide nearly the same hierarchy as geometric multigrid methods on structured uniform grids.

attempt some classical interpolation formulas, the so-called standard interpolation and the so-called direct interpolation; see [46]. These formulas involve the entries a_{kl}^j of the matrix A^j within the interpolation formulas. Numerical experiments showed, however, that, especially on unstructured grids, the resulting prolongations did not provide satisfying approximation rates when used in the sparse grid context. The reason is that the prolongation formulas in AMG are developed to interpolate algebraic smooth errors in as exact a manner as possible, i.e., errors which cannot efficiently be reduced by a given stationary linear iteration scheme. This requirement, however, does not have to lead to good approximation results when used in space-time sparse grids.

Therefore we use a least squares approach to compute interpolation formulas which are able to reproduce (global) linear or d-linear functions in as exact a manner as possible. We denote by $\bar{u}^j \in \mathbb{R}^{n_j}$ the vector of coefficients of the basis representation of $u_j \in V_j^\Omega$. We choose a basis $\{\zeta_i\}$ of the space of linear or, alternatively, d-linear functions, i.e., $\{1, x_i, i = 1, \ldots, d\}$ or $\{1, x_{i_1}, x_{i_1}x_{i_2}, \ldots, x_{i_1} \cdot \ldots \cdot x_{i_d}, i_j = 1, \ldots, d, i_1 < i_2, i_1 < i_2 < i_3, \ldots, i_1 < i_2 < \cdots < i_d\}$, and a norm $\|\cdot\|_\star$ on $\mathbb{R}^{n_j}$, and define the functional $J : \mathbb{R}^{(n_j - n_{j-1}) \times n_{j-1}} \mapsto \mathbb{R}$ as

$$J(\tilde{P}_j) := \sum_{\zeta_i} \|\bar{\zeta}_i^j - (\tilde{P}_j \bar{\zeta}_i^{j-1})\|_\star^2. \tag{3.10}$$

Now we compute the block $\tilde{P}_j$ in (3.9) as the solution of the constrained minimization problem

$$\min_{\tilde{P}_j \in \mathbb{R}^{(n_j - n_{j-1}) \times n_{j-1}}} J(\tilde{P}_j). \tag{3.11}$$

In general, this would lead to densely populated operators $\tilde{P}_j = (\tilde{p}_{kl}^j)$. Therefore, we impose a second constraint on the above minimization problem, namely

$$a_{kl}^j = 0 \Rightarrow \tilde{p}_{kl}^j = 0. \tag{3.12}$$

Loosely speaking, this constraint enforces that only neighboring grid points are used in the prolongation process to interpolate fine grid points. Note that due to the constraint (3.12) on the sparsity pattern of the prolongation operator, the overall constraint minimization (3.11) can be split into local small least squares problems which can be efficiently solved by simple LU-factorization.

For the choice of $\|\cdot\|_\star$ in (3.10) there are several possibilities. The norm induced by the coarse grid operators A^j which are set up from the AMG approach anyway, i.e.,

$$\|x\|_{A^j} := (x^T A^j x)^{1/2},$$

would lead to cost functionals similar to those used for the construction of prolongation operators in the context of AMG methods based on element interpolation (AMGe) [7]. In the following, we will use the Euclidean norm instead. It is easy to compute, and provided good results in our numerical experiments. Nevertheless, future research is needed to study in more detail the influence of the choice of the norm $\|\cdot\|_\star$ on the space-time sparse grid approach.

4. Data structures and adaptivity. In this section we will discuss the underlying data structure used to store the space-time sparse grid coefficients. The aim is

to obtain a storage format for the coefficients which allows for an efficient implementation of a sparse grid interpolant for given nodal values and vice versa; see [2, 40] for more details.

For classical sparse grids, there are several approaches [2, 11, 35, 40] to this topic which use hash maps or binary trees to manage the coefficients of the sparse grid basis. Note, however, that we are not dealing with classical sparse grids which are constructed by a tensor product from a purely one-dimensional multilevel basis. There, each basis function has a fixed number of hierarchical sons such that tree data structures can be efficiently applied. In our application the space part of the multilevel basis is allowed to stem from general unstructured grids instead. Therefore, the simple concept of recursively used binary trees is no longer efficiently applicable. Instead we use a slightly different approach, i.e., binary trees of hash maps, which is described in the following. The associated data structures allow for an efficient implementation of the operations that have to be performed on a space-time sparse grid, such as, e.g., the computation of the function values of the interpolant [40]. They are also well suited for adaptive refinement procedures on space-time sparse grids.

4.1. Algorithmic approach and data structures. Let $\mathcal{B}(V_\ell^T) = \{\psi_{j,i}^T\}$ be the piecewise linear hierarchical basis in time for V_ℓ^T, i.e., $\psi_{j,i}^T \in W_j^T$, where the index j indicates the level, $||\psi_{j,i}^T||_\infty = 1$, and

$$\operatorname{supp}(\psi_{j,i}^T) \cap \operatorname{supp}(\psi_{j,k}^T) = \emptyset \ \forall i \neq k. \tag{4.1}$$

Furthermore, let $\mathcal{B}(V_\ell^\Omega) = \{\psi_{j,i}^\Omega\}$ be the spatial algebraic hierarchical basis constructed from a finite element space as described in section 3.2. Then, we obtain a basis of the space-time sparse grid space $\tilde{V}_\ell^1$ by

$$\mathcal{B}(\tilde{V}_\ell^1) := \{\psi_{\mathbf{j},\mathbf{i}} := \psi_{j_1,i_1}^\Omega \cdot \psi_{j_2,i_2}^T \mid j_1 + j_2 \leq \ell\}. \tag{4.2}$$

For ease of presentation, we will use the set of multilevel indices $\mathcal{I}_{\tilde{V}_\ell^1}$ contained in the sparse grid $\tilde{V}_\ell^1$,

$$\mathcal{I}_{\tilde{V}_\ell^1} := \{(\mathbf{j},\mathbf{i}) \mid \ j_1 + j_2 \leq \ell,\ 0 \leq i_1 \leq \dim(W_{j_1}^\Omega),\ 0 \leq i_2 \leq \dim(W_{j_2}^T)\}.$$

Furthermore, to each multilevel index $(\mathbf{j},\mathbf{i})$ we associate a grid point $(x_{j_1,i_1}, t_{j_2,i_2})$ such that

$$\psi_{(\mathbf{j},\mathbf{i})}(x_{j_1,i_1}, t_{j_2,i_2}) = 1.$$

We define the usual hierarchical relation $>$ between two hierarchical indices (j_1,i_1) and $(\bar{j}_1,\bar{i}_1)$ of the multilevel basis in space as

$$(j_1,i_1) > (\bar{j}_1,\bar{i}_1) :\Leftrightarrow \operatorname{supp}\psi_{\bar{j}_1,\bar{i}_1}^\Omega \subset \operatorname{supp}\psi_{j_1,i_1}^\Omega \tag{4.3}$$

and we use the analogous definition for the hierarchical indices of the spatial multilevel basis. Now, given the nodal values $u(x_{j_1,i_1}, t_{j_2,i_2})$ of a continuous function u at the sparse grid points $(x_{j_1,i_1}, t_{j_2,i_2})$, we want to compute the coefficients $u_{\mathbf{j},\mathbf{i}}$ of the sparse grid interpolant u_ℓ^{sp},

$$u_\ell^{sp} = \sum_{(\mathbf{j},\mathbf{i}) \in \mathcal{I}_{\tilde{V}_\ell^1}} u_{\mathbf{j},\mathbf{i}} \psi_{\mathbf{j},\mathbf{i}}(x,t),$$

i.e.,

$$u_\ell^{sp}(x_{\tilde{j}_1,\tilde{i}_1}, t_{\tilde{j}_2,\tilde{i}_2}) = \sum_{(\mathbf{j},\mathbf{i})\in\mathcal{I}_{\tilde{V}_\ell^1}} u_{\mathbf{j},\mathbf{i}}\psi_{\mathbf{j},\mathbf{i}}(x_{\tilde{j}_1,\tilde{i}_1}, t_{\tilde{j}_2,\tilde{i}_2}),$$

for all sparse grid points $(x_{\tilde{j}_1,\tilde{i}_1}, t_{\tilde{j}_2,\tilde{i}_2})$. Using (4.1) we obtain

$$\begin{aligned} u(x_{\tilde{j}_1,\tilde{i}_1}, t_{\tilde{j}_2,\tilde{i}_2}) &= \sum_{(\mathbf{j},\mathbf{i})\in\mathcal{I}_{\tilde{V}_\ell^1}} u_{\mathbf{j},\mathbf{i}}\psi_{\mathbf{j},\mathbf{i}}(x_{\tilde{j}_1,\tilde{i}_1}, t_{\tilde{j}_2,\tilde{i}_2}) \\ &= \sum_{\substack{(j_2,i_2)\geq(\tilde{j}_2,\tilde{i}_2)\\(\mathbf{j},\mathbf{i})\in\mathcal{I}_{\tilde{V}_\ell^1}}} \psi_{j_2,i_2}^T(t_{\tilde{j}_2,\tilde{i}_2}) \left(\sum_{\substack{j_1,i_1\\(\mathbf{j},\mathbf{i})\in\mathcal{I}_{\tilde{V}_\ell^1}}} u_{\mathbf{j},\mathbf{i}}\psi_{j_1,i_1}^\Omega(x_{\tilde{j}_1,\tilde{i}_1}) \right). \end{aligned}$$

Therefore, to obtain the hierarchical coefficients we proceed in two steps. First, according to the second sum, we compute spatial multilevel coefficients $\hat{u}_{\mathbf{j},\mathbf{i}}$ such that for each time step $t_{\tilde{j}_2,\tilde{i}_2}$

$$u(x_{\tilde{j}_1,\tilde{i}_1}, t_{\tilde{j}_2,\tilde{i}_2}) = \sum_{\substack{j_1,i_1\\((\tilde{j}_1,j_2),(\tilde{i}_1,i_2))\in\mathcal{I}_{\tilde{V}_\ell^1}}} \hat{u}_{\mathbf{j},\mathbf{i}}\psi_{j_1,i_1}^\Omega(x_{\tilde{j}_1,\tilde{i}_1}) \quad \forall x_{\tilde{j}_1,\tilde{i}_1},\ (\tilde{\mathbf{j}},\tilde{\mathbf{i}})\in\mathcal{I}_{\tilde{V}_\ell^1}. \tag{4.4}$$

Then we apply a basis transformation from the nodal basis to the multilevel basis in time to transform $\hat{u}_{\mathbf{j},\mathbf{i}}$ to the sparse grid coefficients $u_{\mathbf{j},\mathbf{i}}$. Here, according to the first sum which involves all hierarchical ancestors, this can be done by a simple top-down tree traversal if the coefficients $\hat{u}_{\mathbf{j},\mathbf{i}}$ are stored in a binary tree.

Thus, this algorithm suggests the use of a binary tree for the multilevel basis in time, where each node contains an array for the coefficients which belong to the respective time step. However, we also want to be able to adaptively refine our space-time sparse grids, e.g., by starting with a regular sparse grid $\tilde{V}_\ell^1$ and locally inserting new grid points and associated basis functions according to certain error estimators or indicators. Therefore we use hash maps in each node instead of simple arrays. For a detailed discussion on the use of hash maps in the context of sparse grids and hierarchical bases, see [40, 27]. For a general introduction we refer to [38]. Here, a hash table M is used to store the data, and each data entry has to be uniquely identified by a key, for which we simply use the spatial hierarchical index (j_1, i_1). Now, we need a hash function h to compute the position $h((j,i))$ where the associated coefficient is stored in the hash table M. In each node of the binary tree which represents the time level j_2 we use a hash table with $\dim(V_{\ell-j_2}^\Omega)$ entries and define the hash map h by

$$h((j,i)) = (\dim(V_{j-1}^\Omega) + i) \mod \dim(V_{\ell-j_2}^\Omega),$$

where $V_0^\Omega := \emptyset$. Obviously, for a regular sparse grid $\tilde{V}_\ell^1$, there are exactly $\dim(V_{\ell-j_2}^\Omega)$ data entries in each node which belong to the time level j_2, and the above hash function h provides a mapping without collision. However, for adaptively refined sparse grids, different coefficients can result in the same hash values. Therefore, we use so-called [illegible] [25] where all coefficients with the same hash value are stored in a separate list. Then, to obtain the coefficient value for a given spatial index, we have to search the list which belongs to the hash value of the key. Numerical experiments showed that these lists are usually very small, so that the computational cost of this search is bounded by a small constant.

4.2. Adaptivity. Sparse grids can easily be refined adaptively; cf. [2, 8, 35]. This is necessary if the function to be approximated does not fulfill the smoothness requirements which are a prerequisite for regular sparse grids. This way singularities or strong variations in the function can be taken care of. It can be shown that the favorable approximation rates and complexities of regular sparse grids also can be obtained for nonsmooth functions provided that the right adaptivity scheme with a proper error estimator is used. Here one can derive classical error estimators involving residual- or duality-based error indicators in a way similar to classical finite elements; see [9, 12]. An alternative is the use of the size of the coefficients in a wavelet-like representation as indicators for local refinement along the lines of [17, 18, 19]. It can be shown that, for any stable multiscale basis, such an approach provides true error estimators, i.e., error indicators which are reliable and efficient. Note that there is an easy way to switch from our hierarchical basis representation to such a wavelet-type representation by means of the lifting scheme [35, 43].

In the following we simply use the weighted size of the coefficients in the hierarchical basis representation as an error indicator for local refinement. This approach was successfully used in several different application areas, e.g., in visualization [23], in numerical integration [24], and for the solution of PDEs [2, 8, 12, 35]. It results in a reliable but in general not perfectly efficient error estimator.[5] Nevertheless, numerical results show that the proposed adaptivity criterion provides quite good results.

We proceed as follows: For a basis function $\psi_{\mathbf{j},\mathbf{i}} = \psi^T_{j_1,i_i} \cdot \psi^\Omega_{j_2,i_2}$ we define the set $\mathcal{H}(\psi_{\mathbf{j},\mathbf{i}})$ of hierarchical sons, i.e.,

$$\mathcal{H}(\psi_{\mathbf{j},\mathbf{i}}) := \{\psi_{\bar{\mathbf{j}},\bar{\mathbf{i}}} \mid (\bar{j}_1,\bar{j}_2) = (j_1+1,j_2) \text{ or } (\bar{j}_1,\bar{j}_2) = (j_1,j_2+1), \tag{4.5}$$

$$\psi^T_{j_1,i_i} > \psi^T_{\bar{j}_1,\bar{i}_1} \text{ and } \psi^\Omega_{j_2,i_2} > \psi^\Omega_{\bar{j}_2,\bar{i}_2}\}, \tag{4.6}$$

using the $>$-relation from (4.3). Now, given the sparse grid approximation $u^{sp} = \sum_{(\mathbf{j},\mathbf{i})\in\Lambda} u_{\mathbf{j},\mathbf{i}} \cdot \psi_{\mathbf{j},\mathbf{i}}$, where Λ denotes the set of involved indices, we add $\mathcal{H}(\psi_{\mathbf{j},\mathbf{i}})$ to the actual sparse grid space[6] and thus the indices $(\tilde{\mathbf{j}},\tilde{\mathbf{i}})$ of the hierarchical sons to the actual set Λ whenever

$$|u_{\mathbf{j},\mathbf{i}}| > 2^{\frac{1}{2}(d\cdot j_1 + j_2)} \cdot \varepsilon \tag{4.7}$$

for a given threshold ε. Here, the factor $\frac{1}{2}$ in the exponent resembles a measurement of the error in the L^2-norm; for other norms, appropriate values have to be substituted. The term $d \cdot j_1 + j_2$ corresponds to an approximation scheme with the same order in space and time. Such a refinement strategy leads asymptotically to an equilibration of the error indicators in the sense of [1].

5. Numerical results. In this section we present some numerical results concerning the convergence rates of the space-time sparse grid interpolant in $\tilde{V}^1_\ell$ for the regular case and the adaptive refinement case. To this end, we use a bilinear finite element space V^Ω_ℓ, where $h := 2^{-\ell} > 0$ indicates the mesh width of the underlying grid, on the spatial domain $\Omega \subset \mathbb{R}^2$ to construct the spatial multilevel basis as described in section 3. Here we use the stiffness matrix of the Laplacian in the AMG coarsening algorithm.

[5] Note that the upper bound (which implies reliability) in the norm equivalency for the hierarchical basis is (after proper weighting) independent of the number of levels, whereas the lower bound (which relates to efficiency) depends slightly on the number of levels involved.

[6] Of course, this is done only for the indices which are "leaves" of the structure of Λ, i.e., for the indices $(\mathbf{j},\mathbf{i}) \in \Lambda$ with $\mathcal{H}(\psi_{\mathbf{j},\mathbf{i}}) \notin \Lambda$.

For $u \in C^1(\Omega_T)$ given, we set $h := 2^{-\ell}$ and denote by u_h the full grid interpolant of u and by u_ℓ^{sp} the sparse grid interpolant. We define the interpolation errors $e_h := u - u_h$ and $e_\ell^{sp} := u - u_\ell^{sp}$. Then obviously

$$\|e_\ell^{sp}\| \leq \|e_h\| + \|u_h - u_\ell^{sp}\| \tag{5.1}$$

for any given norm $\|\cdot\|$. In the following, we will use the norms

$$\begin{aligned}
\|u\|_{L^2\times L^\infty} &:= \operatorname{ess\,sup}_{t\in(0,T)} \|u(\cdot,t)\|_{L^2(\Omega)}, \\
\|u\|_{L^2\times L^2} &:= \left(\int_{0<t<T} \|u(\cdot,t)\|^2_{L^2(\Omega)}\, dt\right)^{1/2}, \\
\|u\|_{H^1\times L^\infty} &:= \operatorname{ess\,sup}_{t\in(0,T)} |u(\cdot,t)|_{H^1(\Omega)}, \\
\|u\|_{H^1\times L^2} &:= \left(\int_{0<t<T} |u(\cdot,t)|^2_{H^1(\Omega)}\, dt\right)^{1/2}.
\end{aligned}$$

For these norms, classical interpolation theory shows that the error e_h decreases with at least the same order as the error of the sparse grid interpolant e_ℓ^{sp} for decreasing mesh width. We are interested if the orders predicted by our theory in section 2 for the geometrically constructed hierarchical basis can also be achieved for the algebraically constructed increment spaces described in section 3.2. With (5.1) we see that it is sufficient to use $\tilde{e}_h := \|u_h - u_\ell^{sp}\|$, which is easier to compute. Therefore, we will present results in terms of $\tilde{e}_h$ only.

In all the following experiments we consider $T = 1.0$.

1. In our first example we examine the interpolation error for the space-time sparse grid space $\tilde{V}_\ell^1$ for the function

$$u(x,t) = t^2 \cdot (x_1^2 + x_2^2)$$

on the unit square $\Omega = [0,1]^2$. This function is just a product of a quadratic polynomial in time and in space. Here it serves as the simplest model problem for discussing the basic properties of our space-time sparse grid approach. We use a uniform rectangular grid with mesh width $h = 2^{-\ell}$ for the underlying finite element discretization from which the spatial multilevel basis is derived along the lines of subsection 3.2. In this case, the AMG coarsening algorithm of Ruge and Stüben leads to a sequence of coarse grids which is nearly identical to conventional geometric coarsening. Therefore, the constructed spatial multilevel basis is basically the usual isotropic hierarchical basis [49]. Note that the function is zero at the time point $t = 0$ and therefore all degrees of freedom at this time point can be eliminated.

Table 5.1 shows the error as well as the dimension of the space-time sparse grid space. First of all note that the number of degrees of freedom of the space-time sparse grid space $\tilde{V}_\ell^1$ increases with the same order, i.e., $O(h^{-2})$, as the degrees of freedom of the spatial grid as expected from (2.28) of Lemma 2.5. Moreover, the additional constant amount for the space-time sparse grid in comparison to the spatial grid is just a factor of three where we eliminated the degrees of freedom for $t = 0$ (it would be four including those degrees of freedom). Figure 5.1 shows the resulting space-time sparse grid of level $\ell = 5$. Here, one can clearly see the finest grids with $h = 1/32$ at the time points $t = 0.5$ and $t = 1.0$ which dominate the overall number of degrees of freedom.

Furthermore, from Lemma 3.1 in section 3 we expect a behavior of $O(2^{-2\ell}\ell)$ ($O(h^2 \log h)$) for the error in the $L^2 \times L^2$-norm on level ℓ ($h = 2^{-\ell}$). In fact, looking

TABLE 5.1
Degrees of freedom of the space-time sparse grid $\tilde{V}_\ell^1$ and the interpolation error $\tilde{e}_h := u_\ell^{sp} - u_h$ in different norms for the continuous function $u(x,t) = t^2 \cdot (x_1^2 + x_2^2)$.

Level	Spatial DOF	$\tilde{V}_\ell^1$ DOF	$\|\tilde{e}_h\|_{L^2\times L^\infty}$	$\|\tilde{e}_h\|_{L^2\times L^2}$	$\|\tilde{e}_h\|_{H^1\times L^\infty}$	$\|\tilde{e}_h\|_{H^1\times L^2}$
5	1089	3424	2.21e-4	1.64e-4	2.87e-3	2.32e-3
6	4225	13120	6.90e-05	5.10e-05	1.44e-3	1.16e-3
7	16641	51072	2.06e-05	1.53e-05	7.21e-4	5.82e-4
8	66049	200960	6.01e-06	4.44e-06	3.61e-4	2.91e-4
9	263169	796160	1.71e-06	1.27e-06	1.80e-4	1.46e-4

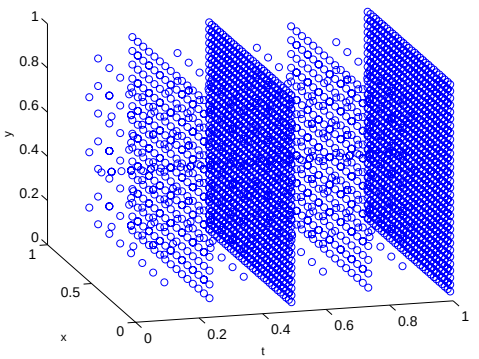


FIG. 5.1. *Space-time sparse grid associated to $\tilde{V}_\ell^1$ with two spatial dimensions for level $\ell = 5$ ($h = 1/32$).*

at the results in Table 5.1 we see that the quotient of the errors $\tilde{e}_h$ of two successive levels of discretization measured in $\|\cdot\|_{L^2\times L^2}$ and $\|\cdot\|_{L^\infty\times L^2}$ slowly approaches 0.25 from above for decreasing mesh width. This value would be the rate achieved with a full grid discretization. In more detail, from level $\ell = 5$ to level $\ell = 6$ the error is reduced by a factor of about 0.31 and decreases for growing ℓ. We finally obtain a factor of about 0.28 for the reduction from $\ell = 8$ to $\ell = 9$. This behavior accounts for the additional $\log h$-term in the convergence rate, which becomes more and more unimportant with growing number of levels in comparison to the exponential term. For the quotient of the error estimate of two consecutive levels we obtain $0.25\cdot(\ell+1)/\ell$, which is also an upper bound for the reduction factor. Moreover, using this formula, we get a theoretical estimate of about 0.28 for the reduction of the error from level $\ell = 8$ to $\ell = 9$, which is exactly the rate we observe in our experiment.

From Lemma 3.2 we would expect the same error reduction rate for the sparse grid interpolant as for the full grid interpolant if the H^1-norm were involved. Indeed, looking at the results in the $H^1 \times L^2$- and $H^1 \times L^\infty$-norms we observe a factor of 0.5 independent of the number of levels, which is just the same as that of a full grid interpolant.

2. We now present results for the more complicated function

$$u(x,t) = \sin(4\pi t x_1) + \sin(4\pi t x_2)$$

on the unit square, $\Omega = (0,1)^2$. Here, $u(x,t)$ is zero for $t = 0$ but develops more and more oscillations for $t \to 1.0$. A plot of the function at different time points is shown in Figure 5.2

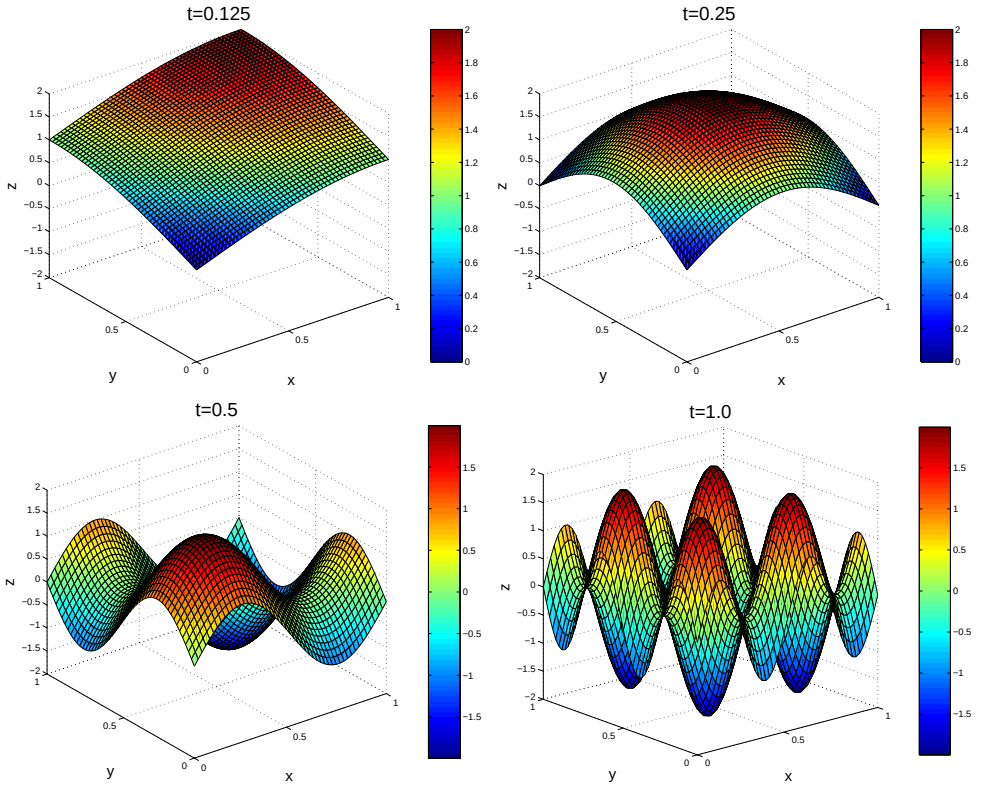


FIG. 5.2. *Plots of the function* $u(x,t) = \sin(4\pi t x_1) + \sin(4\pi t x_2)$ *of Example* 2 *at time points* $t = 0.125, 0.25, 0.5, 1.0$.

Although predicted by the theoretical results given in Lemmas 3.1 and 3.2, it is not directly obvious that the sparse grid approach still works well in practice for this case due to the oscillations in the function $u(x,t)$. Nevertheless, looking at the results in Table 5.2, we see that the behavior of the error measured in the $\|\cdot\|_{L^2\times L^2}$- and $\|\cdot\|_{L^2\times L^\infty}$-norms for increasing level number is the same as in the previous example. Here, the initial reduction rate from level $\ell = 5$ to $\ell = 6$ for the $\|\cdot\|_{L^2\times L^2}$-norm amounts to about 0.35, which is a little bit larger than in the previous example. On level $\ell = 9$ we obtain a reduction rate of about 0.30.

For the two norms involving the H^1-seminorm we expect from Lemma 3.2 an error reduction rate of about 0.5. In fact, the results of Table 5.2 clearly show that we achieve this predicted approximation rate.

3. In this example we show results for the proposed space-time sparse grid method for a slightly more complex geometry in space. We again consider the function

$$u(x,t) = t^2 \cdot (x_1^2 + x_2^2) \tag{5.2}$$

but choose now the unit circle

$$\Omega = \{x \,|\, \|x\| \le 1.0\}$$

as spatial domain. Note that for such a domain a sequence of geometric discretizations on different scales with piecewise linear finite elements no longer results in nested

TABLE 5.2
Degrees of freedom of the space-time sparse grid $\tilde{V}_\ell^1$ and the interpolation error $\tilde{e}_h := u_\ell^{sp} - u_h$ in different norms for the continuous function $u(x,t) = \sin(4 \cdot \pi \cdot t \cdot x_1) + \sin(4 \cdot \pi \cdot t \cdot x_2)$.

Level	Spatial DOF	$\tilde{V}_\ell^1$ DOF	$\|\tilde{e}_h\|_{L^2 \times L^\infty}$	$\|\tilde{e}_h\|_{L^2 \times L^2}$	$\|\tilde{e}_h\|_{H^1 \times L^\infty}$	$\|\tilde{e}_h\|_{H^1 \times L^2}$
5	1089	3424	2.67e-2	1.34e-2	4.05e-1	2.25e-1
6	4225	13120	9.83e-3	4.66e-3	2.09e-1	1.16e-1
7	16641	51072	3.31e-3	1.51e-3	1.04e-1	5.87e-2
8	66049	200960	1.03e-3	4.67e-4	5.24e-2	2.95e-2
9	263169	796160	3.10e-4	1.39e-4	2.63e-3	1.48e-2

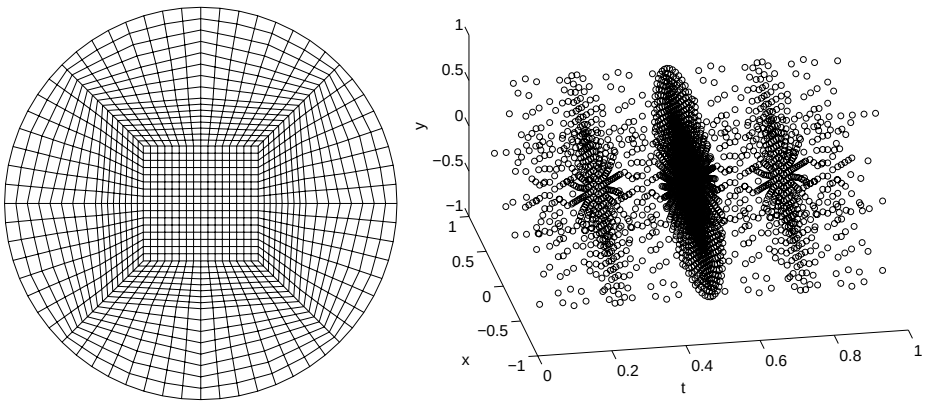


FIG. 5.3. *A spatial grid for the unit circle (left) and the space-time sparse grid (right).*

TABLE 5.3
Degrees of freedom of the spatial grid, the space-time sparse grid, and the interpolation error $\tilde{e}_h := u_{sp} - u_h$ in different norms for $u(x,t) = t^2 \cdot (x_1^2 + x_2^2)$ on the domain $\Omega = \{x | \|x\| \leq 1.0\}$.

Spatial DOF	Overall DOF	$\|\tilde{e}_h\|_{L^2 \times L^\infty}$	$\|\tilde{e}_h\|_{L^2 \times L^2}$	$\|\tilde{e}_h\|_{H^1 \times L^\infty}$	$\|\tilde{e}_h\|_{H^1 \times L^2}$
5185	10203	5.91e-4	4.67e-4	1.14e-2	9.22e-4
20609	40747	1.61e-4	1.23e-4	5.61e-3	4.53e-4
82177	162907	6.19e-05	4.81e-05	2.85e-3	2.32e-4

finite element spaces. This is due to the circular boundary, which is only linearly approximated on each level. Here, the algebraic coarsening by AMG sets in. Together with the prolongation operator constructed by (3.11) and the constraint (3.12), we obtain an algebraic hierarchical basis with good approximation properties also near the boundary. An example of a grid which is used for the finite element discretization of the Laplacian to obtain the initial system matrix for the AMG approach is given in Figure 5.3 (left); an associated space-time sparse grid is depicted in Figure 5.3 (right).

The results for three different space-time sparse grids are shown in Table 5.3. For the $H^1 \times L^2$- and the $H^1 \times L^\infty$-norms we obtain nearly the same approximation rate of about 0.5 as in the previous examples. For the $L^2 \times L^2$- and the $L^2 \times L^\infty$-norms the error is reduced by the factors 0.27 and 0.38, respectively. Here, the grids used for the construction of the spatial finite element spaces vary in their quality depending on their mesh width; i.e., the maximal and minimal interior angles of the rectangles vary between the different grids. This might influence the approximation property of the space-time sparse grid and, consequently, the error reduction rate to some extent.

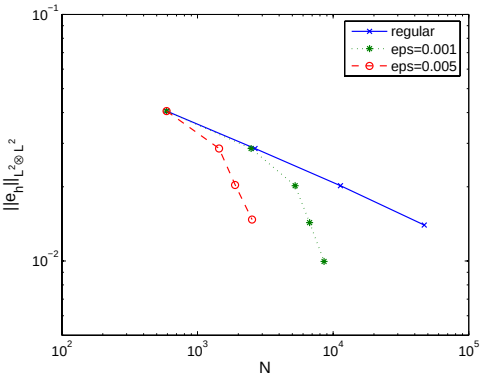


FIG. 5.4. *Interpolation error in the* $\|\cdot\|_{L^2\times L^2}$*-norm for the function* $\sqrt{(t\cdot x_1)}\cdot x_2$*, regular sparse grids for different levels, and adaptive refinement starting from level* $l = 4$ *with thresholds* $\varepsilon = 0.001$ *and* $\varepsilon = 0.005$ *versus overall degrees of freedom (N).*

4. In this example we present numerical results for the function

$$u(x,t) = \sqrt{(t\cdot x_1)}\cdot x_2,$$

which has a singularity on the edge $x_1 = 0$ for all times $t > 0$ and a singularity at $t = 0$. Note that this function is no longer in $H^{2,2}_{mix}(\Omega_T)$ as the functions of the previous examples are, and we therefore expect a decrease of the approximation rates. Furthermore, note that we would also obtain worse approximation results for a full grid approximation in space and time, since u is no longer in $H^{2,2}(\Omega_T)$ either. Therefore, using the error $\tilde{e}_h$ instead of e^{sp}_ℓ no longer provides reliable bounds for the space-time sparse grid approximation rates. Instead we use a very fine full grid ($\ell = 11$) as a reference solution, interpolate the adaptive sparse grid solution to this grid, and measure the error between the full grid and the sparse grid solution there.

To cope with the singularities in $u(x,t)$ we adaptively refine our sparse grids in space and time as described in section 4.2. We use the error indicator (4.7) with the thresholds $\varepsilon = 0.001$ and $\varepsilon = 0.005$. Figure 5.4 shows the error history in the $L^2\times L^2$-norm of the adaptive cycle when we apply the adaptive refinement starting on level $\ell = 4$ with the two different thresholds. For comparison the convergence result for regular sparse grids is given as well. We clearly see that the uniform refinement provides only a poor cost-benefit ratio. Here, starting with an error of about 0.02 for level $\ell = 4$ (i.e., 592 degrees of freedom), the space-time sparse grid results in an error of 0.0024 on level $\ell = 8$ (i.e., 46976 degrees of freedom). This leads to an error reduction rate with respect to the number of degrees of freedom N of 0.48, i.e., the error behaves as $O(N^{-0.48})$ which can be seen from the slope of the respective curve in Figure 5.4. Remember that for a smooth solution[7] we would obtain a behavior of nearly $O(N^{-1})$. Now, looking at the results for the adaptive cycles we see that we obtain a much higher accuracy with substantially fewer degrees of freedom. In the case of $\varepsilon = 0.001$ we obtain a final error of 0.0015 with only 8578 degrees of freedom, and the use of $\varepsilon = 0.005$ results in a final error of 0.0038 with 3286 degrees of freedom.

[7]In the previous examples the degrees of freedom grew roughly with a factor of 4 from level to level, and we gave the quotient of the error on two successive levels as reduction rate. Now we directly give the exponent of the error as a function of degrees of freedom, which is more appropriate in the adaptive case.

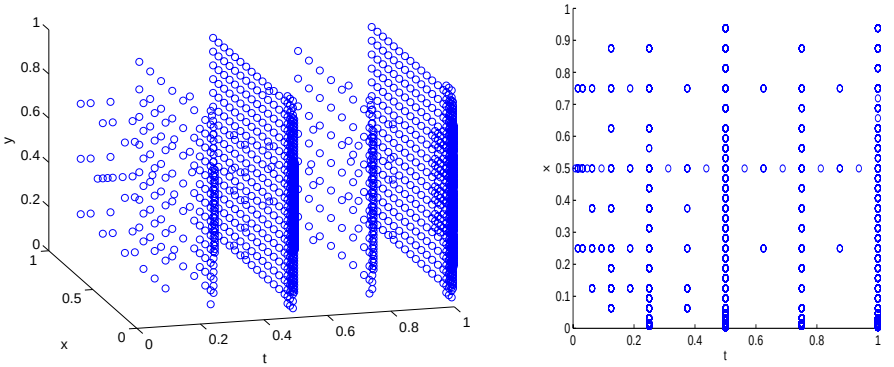


FIG. 5.5. *Adaptively refined space-time grid ($\varepsilon = 0.001$) after three refinement steps for the function $u(x,t) = \sqrt{(t \cdot x_1)} \cdot x_2$.*

Now, if we look at the cost-benefit ratio, i.e., if we compare the error and the degrees of freedom of the initial grid on level $\ell = 4$ and the grids obtained at the end of the adaptive cycles, we see that we get a rate of about 1.0 for both thresholds which is exactly the rate we would expect for a regular sparse grid and a smooth function. In Figure 5.5, we show the adaptive grid which results from the refinement process for $\varepsilon = 0.001$. Here we see that the nonsmoothness in space and time direction is detected by the error indicator and gets resolved by the insertion of additional points in [illegible]. This indicates that the proposed error indicator works well.

6. Conclusion. We presented a space-time sparse grid method for the approximation of functions which depend on space and time. It involves the tensor product of a multilevel basis in space and a multilevel basis in time. A proper truncation of the resulting series expansion then gives a space-time sparse grid. This approach reduces the number of degrees of freedom significantly in comparison to a conventional space-time finite element method.

Furthermore, we proposed a method to algebraically compute the spatial multilevel basis from a given finite element space. It involves AMG and special prolongation operators which are determined by a local minimization procedure. This allows us to also handle complex spatial geometries which pose a problem for classical sparse grid spaces. In the present paper we utilized a simple least squares approach to guarantee the approximation property of the coarse spaces. Other AMG interpolation procedures (i.e., [47] and [48]) are also feasible and worth pursuing in the future.

We showed that these space-time sparse grid spaces possess nearly the same approximation rates as classical finite element spaces if just slightly stronger regularity assumptions are fulfilled. Furthermore, it turned out that their number of degrees of freedom is of the same order as that of the spatial finite element space alone; i.e., the dimension of the space-time sparse grid spaces is one order of magnitude smaller than the dimension of conventional space-time finite element spaces. In addition numerical experiments showed that the approximation rates predicted by the theory really can be achieved in practice, even for complicated spatial geometries which do not allow for a sequence of nested finite element spaces. We also demonstrated for a problem with singularities in space and time that space-time adaptivity can easily be implemented and used. This results in adaptively refined space-time sparse grids which

resolve singularities and thus allow us to regain the superior convergence rates and complexities of space-time sparse grids also in the case of nonsmooth functions.

In this article we restricted ourselves to the hierarchical basis in space and in time as a special case of a multiscale basis. Note, however, that our approach is by no means confined to this type of basis. It works in the same (or an even better) way for any stable multiscale basis, such as wavelets or frames. However, to get rid of the relatively tight restriction on the spatial domain imposed by wavelet-like schemes due to translation and dilation, we opted for an AMG construction with associated algebraic hierarchical basis instead.

The proposed space-time sparse grid spaces can be efficiently used for the discretization of parabolic problems and related time-dependent control and optimization problems. To this end it is necessary to discretize and to apply the respective differential operator, e.g., in weak formulation, in space and time. The associated computational work should be proportional to the number of degrees of freedom. This is easily possible whenever the coefficient functions of the operator which belong to the spatial derivatives are constant in time. Then an algorithm which is based on the so-called unidirectional principle [8, 12] can be applied. Also the efficient solution of the arising linear system in optimal complexity is necessary by, e.g., a multilevel method which must be able to tackle the space-time discretized sparse grid problem directly. This, however, is future work.

7. Appendix. In this section, we give the proof of Lemma 3.2. Since the application of Lemma 2.4 shows the estimate, we just have to verify that inequality (2.23) holds true.

In the following, we denote a nodal basis of the spatial finite element space V_j^Ω by $\varphi_{j,i}^\Omega$, $1 \le i \le \dim(V_j^\Omega)$. Analogously, the hierarchical increment space W_j^Ω is spanned by a subset of the nodal basis of V_j^Ω. For ease of presentation we denote the basis elements of W_j^Ω by $\psi_{j,i}^\Omega$, $1 \le i \le \dim(W_j^\Omega)$. Furthermore (after reordering of the nodal set $\mathcal{N}_j$), we assume that the index i of the function $\psi_{j,i}^\Omega$ is associated to the nodal point where the basis function is equal to one; i.e., for every $n_j \in \mathcal{N}_j$ we have

$$\psi_{j,i}^\Omega(n_k) = \begin{cases} 1 & \text{if } k = i, \\ 0 & \text{otherwise.} \end{cases}$$

Using the definition (3.2) of W_j^Ω and the interpolation property of the prolongation operators P_j, we obtain the coefficients $w_{j,i}^\Omega$ of the basis representation $w_j^\Omega = \sum_{i=1}^{\dim(W_j)} w_{j,i}^\Omega \psi_{j,i}^\Omega$ of an element w_j^Ω, $w_j^\Omega \in W_j^\Omega$ by the formula

$$w_{j,i}^\Omega = (P_j u - P_{j-1} u)(n_i), \tag{7.1}$$

where $n_i \in \mathcal{N}_j$ is the nodal point with $\psi_{j,i}^\Omega(n_i) = 1.0$. Classical interpolation properties of finite element spaces (cf. [14]) show that

$$\|P_j u - P_{j-1} u\|_{L^\infty(\Omega)} \le \|P_j u - u\|_{L^\infty(\Omega)} + \|u - P_{j-1} u\|_{L^\infty(\Omega)} \le c \cdot 2^{-2j} \|u\|_{H^{2,\infty}},$$

and relation (7.1) leads to

$$|w_{j,i}^\Omega| \le \|P_j u - P_{j-1} u\|_{L^\infty(\Omega)} \le c \cdot 2^{-2j} \|u\|_{H^{2,\infty}(\Omega)}, \tag{7.2}$$

with $c > 0$ independent of u, j, and i. We denote by $\psi^T_{j,i}$ an element of the one-dimensional hierarchical basis on $(0,T)$ and derive the analogous estimate

$$|w^T_{j,i}| \le c \cdot 2^{-2j} \|u\|_{H^{2,\infty}((0,T))} \tag{7.3}$$

for $u \in H^{2,\infty}((0,T))$, with $u = \sum_j \sum_i w^T_{j,i} \psi^T_{j,i}$ and $c > 0$ independent of u, j, and i.

A basis of the increment space $W_{\mathbf{j}}$, $\mathbf{j} \in \mathbb{N}^2$, is given by $\{\psi_{\mathbf{j},\mathbf{i}}\}$, where $\psi_{\mathbf{j},\mathbf{i}} := \psi^\Omega_{j_1,i_1} \cdot \psi^T_{j_2,i_2}$, and we obtain the basis representation

$$w_{\mathbf{j}} = \sum_{\mathbf{i}} w_{\mathbf{j},\mathbf{i}} \psi_{\mathbf{j},\mathbf{i}} \tag{7.4}$$

for a function $w_{\mathbf{j}} \in W_{\mathbf{j}}$. Using (7.1) and (7.3) and a tensor product argument (cf. [31]), we obtain for $u \in H^{2,2,\infty}_{mix}(\Omega_T)$, $u = \sum_{\mathbf{j},\mathbf{i}} w_{\mathbf{j},\mathbf{i}} \psi_{\mathbf{j},\mathbf{i}}$, the estimate

$$|w_{\mathbf{j},\mathbf{i}}| \le c \cdot 2^{-2\|\mathbf{j}\|_1} |u|_{H^{2,2,\infty}_{mix}(\Omega)}$$

with $c > 0$ independent of u, $\mathbf{j}$ and $\mathbf{i}$.

Now we want to derive an estimate for the term $\|\psi_{\mathbf{j},\mathbf{i}}\|_E$. The tensor product structure of $\psi_{\mathbf{j},\mathbf{i}}$ leads to

$$\|\psi_{\mathbf{j},\mathbf{i}}\|_E = \|\psi^\Omega_{j_1,i_1}\|_{L^2(\Omega)} \cdot \|\psi^T_{j_2,i_2}\|_{E,(0,T)} + \|\psi^\Omega_{j_1,i_1}\|_{E,\Omega} \cdot \|\psi^T_{j_2,i_2}\|_{L^2((0,T))}. \tag{7.5}$$

Since we assume that the finite element spaces V^Ω_j are affine equivalent and we are dealing with a regular triangulation, classical finite element theory (cf. [14]) leads to

$$\|\psi^\Omega_{j_1,i_1}\|_{E,\Omega} \le c 2^{-(d-2)\cdot(j_1/2)}, \tag{7.6}$$

$$\|\psi^\Omega_{j_1,i_1}\|_{L^2(\Omega)} \le c 2^{-dj_1/2}. \tag{7.7}$$

For the one-dimensional hierarchical basis a straightforward calculation (see [12]) gives

$$\|\psi^T_{j_2,i_2}\|_{E,(0,T)} \le c 2^{j_2/2}, \tag{7.8}$$

$$\|\psi^T_{j_2,i_2}\|_{L^2((0,T))} \le c 2^{-j_2/2}. \tag{7.9}$$

This, together with (7.5), leads to

$$\|\psi_{\mathbf{j},\mathbf{i}}\|_E \le c \left(2^{-(d-2)\cdot(j_1/2)} \cdot 2^{-j_2/2} + 2^{-dj_1/2} \cdot 2^{j_2/2} \right) \tag{7.10}$$

$$\le c 2^{-dj_1/2} 2^{-j_2/2} \left(2^{j_1} + 2^{j_2} \right) \tag{7.11}$$

$$\le c 2^{-dj_1/2} 2^{-j_2/2} 2^{\|\mathbf{j}\|_\infty}. \tag{7.12}$$

Now, for $u \in H^{2,2,\infty}_{mix}(\Omega_T)$, $u = \sum_{\mathbf{j}} w_{\mathbf{j}}$, we obtain

$$\begin{aligned}
\|w_{\mathbf{j}}\|^2_{E,\Omega_T} &\le \sum_{\mathbf{i}} w^2_{\mathbf{j},\mathbf{i}} \|\psi_{\mathbf{j},\mathbf{i}}\|^2_{E,\Omega_T} \\
&\le \sum_{\mathbf{i}} 2^{-4\|\mathbf{j}\|_1} \cdot \|u\|^2_{H^{2,2,\infty}_{mix}(\Omega_T)} \cdot \|\psi_{\mathbf{j},\mathbf{i}}\|^2_{E,\Omega_T} \\
&\le c 2^{-4\|\mathbf{j}\|_1 + 2\|\mathbf{j}\|_\infty} \cdot 2^{-dj_1 - j_2} \sum_{\mathbf{i}} \|u\|^2_{H^{2,2,\infty}_{mix}(\Omega_T)} \\
&\le c 2^{-4\|\mathbf{j}\|_1 + 2\|\mathbf{j}\|_\infty} \|u\|^2_{H^{2,2,\infty}_{mix}(\Omega_T)}.
\end{aligned}$$

Therefore, we can apply Lemma 2.4, which then shows Lemma 3.2. ∎

REFERENCES

[1] I. Babuška and W. C. Rheinboldt, *Error estimates for adaptive finite element computations*, SIAM J. Numer. Anal., 15 (1978), pp. 736–754.

[2] R. Balder, *Adaptive Verfahren für elliptische und parabolische Differentialgleichungen*, Ph.D. thesis, Technische Universität München, Munich, Germany, 1994.

[3] D. Braess, *Finite Elements. Theory, Fast Solvers, and Applications in Solid Mechanics*, Cambridge University Press, Cambridge, UK, 2001.

[4] A. Brandt, *Algebraic multigrid theory: The symmetric case*, Appl. Math. Comput., 19 (1986), pp. 23–56.

[5] A. Brandt, S. F. McCormick, and J. W. Ruge, *Algebraic Multigrid for Automatic Multigrid Solutions with Application to Geodetic Computations*, Technical report, Institute for Computational Studies, Fort Collins, CO, 1982.

[6] A. Brandt, S. F. McCormick, and J. W. Ruge, *Algebraic multigrid for sparse matrix equations*, in Sparsity and Its Applications, D. J. Evans, ed., Cambridge University Press, Cambridge, UK, 1984.

[7] M. Brezina, A. J. Cleary, R. D. Falgout, V. E. Henson, J. E. Jones, T. A. Manteuffel, S. F. McCormick, and J. W. Ruge, *Algebraic multigrid based on element interpolation (AMGe)*, SIAM J. Sci. Comput., 22 (2000), pp. 1570–1592.

[8] H.-J. Bungartz, *Dünne Gitter und deren Anwendung bei der adaptiven Lösung der dreidimensionalen Poisson-Gleichung*, Ph.D. thesis, Technische Universität München, Munich, Germany, 1992.

[9] H.-J. Bungartz, *Finite Elements of Higher Order on Sparse Grids*, Shaker Verlag, Aachen, Germany, 1998.

[10] H.-J. Bungartz, T. Dornseifer, and C. Zenger, *Tensor product approximation spaces for the efficient numerical solution of partial differential equations*, in Proceedings of the International Workshop on Scientific Computations, Konya, Turkey, 1996, Nova Science Publishers, Hauppauge, NY, 1997.

[11] H.-J. Bungartz and M. Griebel, *A note on the complexity of solving Poisson's equation for spaces of bounded mixed derivatives*, J. Complexity, 15 (1999), pp. 167–199.

[12] H.-J. Bungartz and M. Griebel, *Sparse grids*, Acta Numer., 13 (2004), pp. 147–269.

[13] C. Chui and Y. Wang, *A general framework for compactly supported splines and wavelets*, J. Approx. Theory, 71 (1992), pp. 263–304.

[14] P. G. Ciarlet, *The Finite Element Method for Elliptic Problems*, Classics Appl. Math. 40, SIAM, Philadelphia, 2002.

[15] A. Cohen, *Numerical Analysis of Wavelet Methods*, Stud. Math. Appl. 32, North–Holland, Amsterdam, 2003.

[16] A. Cohen, L. M. Echeverry, and Q. Sun, *Finite element wavelets*, Technical report, Laboratoire d'Analyse Numérique, Université Pierre et Marie Curie, Paris, 2000.

[17] S. Dahlke, W. Dahmen, and R. DeVore, *Nonlinear approximation and adaptive techniques for solving elliptic operator equations*, in Multiscale Wavelet Methods for Partial Differential Equations, W. Dahmen, A. Kurdila, and P. Oswald, eds., Academic Press, San Diego, 1997, pp. 237–283.

[18] S. Dahlke, W. Dahmen, R. Hochmuth, and R. Schneider, *Stable multiscale bases and local error estimation for elliptic problems*, Appl. Numer. Math., 23 (1997), pp. 21–47.

[19] W. Dahmen, *Wavelet and multiscale methods for operator equations*, Acta Numer., 6 (1997), pp. 55–228.

[20] W. Dahmen and R. Stevenson, *Element-by-element construction of wavelets satisfying stability and moment conditions*, SIAM J. Numer. Anal., 37 (1999), pp. 319–352.

[21] I. Daubechies, *Orthogonal bases of compactly supported wavelets*, Comm. Pure Appl. Math., 41 (1988), pp. 909–996.

[22] G. Faber, *Über stetige Funktionen*, Math. Ann., 66 (1908), pp. 81–94.

[23] T. Gerstner, *Adaptive hierarchical methods for landscape representation and analysis*, in Process Modelling and Landform Evolution, Lecture Notes in Earth Sci. 78, Springer-Verlag, 1999.

[24] T. Gerstner and M. Griebel, *Numerical integration using sparse grids*, Numer. Algorithms, 18 (1998), pp. 209–232.

[25] G. H. Gonnet and R. Baeza-Yates, *Handbook of Algorithms and Data Structures*, 2nd ed., Addison–Wesley, Reading, MA, 1991.

[26] T. Grauschopf, M. Griebel, and H. Regler, *Additive multilevel-preconditioners based on bilinear interpolation, matrix dependent geometric coarsening and algebraic multigrid coarsening for second order elliptic PDEs*, Appl. Numer. Math., 23 (1997), pp. 63–96.

[27] M. Griebel, *Adaptive sparse grid multilevel methods for elliptic PDEs based on finite differences*, Computing, 61 (1998), pp. 151–179.
[28] M. Griebel and S. Knapek, *Optimized tensor-product approximation spaces*, Constr. Approx., 16 (2000), pp. 525–540.
[29] M. Griebel and D. Oeltz, *A sparse grid space-time discretization scheme for parabolic problems*, to be submitted.
[30] M. Griebel and P. Oswald, *Tensor product type subspace splittings and multilevel iterative methods for anisotropic problems*, Adv. Comput. Math., 4 (1995), pp. 171–206.
[31] M. Griebel, P. Oswald, and T. Schiekofer, *Sparse grids for boundary integral equations*, Numer. Math., 83 (1999), pp. 279–312.
[32] A. Griewank, *Achieving logarithmic growth of temporal and spatial complexity in reverse automatic differentiation*, Optim. Methods Softw., 1 (1992), pp. 35–54.
[33] M. D. Gunzburger, *Perspectives in Flow Control and Optimization*, Adv. Des. Control 5, SIAM, Philadelphia, 2002.
[34] M. Hinze and K. Kunisch, *Three control methods for time-dependent fluid flow*, J. Flow, Turbul. Combust., 65 (2000), pp. 273–298.
[35] F. Koster, *Multiskalen-basierte Finite-Differenzen-Verfahren auf adaptiven dünnen Gittern*, Ph.D. thesis, Rheinische Friedrich-Hilhelms-Universität, Bonn, Germany, 2002.
[36] K. Kunisch and S. Volkwein, *Galerkin proper orthogonal decomposition methods for parabolic systems*, Numer. Math., 90 (2001), pp. 117–148.
[37] O. A. Ladyzenskaja, V. A. Solonnikov, and N. N. Ural'ceva, *Linear and Quasilinear Equations of Parabolic Type*, Trans. Math. Monogr. 23, American Mathematical Society, Providence, RI, 1988.
[38] K. Mehlhorn, *Datenstrukturen und Effiziente Algorithmen, Band* 1: *Sortieren und Suchen*, Teubner Verlag, Stuttgart, 1988.
[39] J. W. Ruge and K. Stüben, *Efficient solution of finite difference and finite element equations*, in Multigrid Methods for Integral and Differential Equations, Inst. Math. Appl. Conf. Ser., New Ser. 3, D. J. Paddon and H. Holstein, eds., Oxford University Press, New York, 1985.
[40] T. Schiekofer, *Die Methode der Finiten Differenzen auf Dünnen Gittern*, Ph.D. thesis, Rheinische Friedrich-Hilhelms-Universität, Bonn, Germany, 1998.
[41] P. Schröder and W. Sweldens, *Spherical wavelets: Efficiently representing functions on the sphere*, in Computer Graphics Proceedings (SIGGRAPH 95), 1995, pp. 161–172.
[42] R. Stevenson, *Locally supported, piecewise polynomial biorthogonal wavelets on nonuniform meshes*, Constr. Approx., 19 (2003), pp. 477–508.
[43] W. Sweldens, *The lifting scheme: A construction of second generation wavelets*, SIAM J. Math. Anal., 29 (1998), pp. 511–546.
[44] V. N. Temlyakov, *Approximation of functions with bounded mixed derivative*, Proc. Steklov Inst. Math., 178 (1989).
[45] V. N. Temlyakov, *On approximate recovery of functions with bounded mixed derivatives*, J. Complexity, 9 (1993), pp. 41–59.
[46] U. Trottenberg, C. W. Osterlee, and A. Schüller, *Multigrid*, Academic Press, San Diego, 2001, Appendix A: An Introduction to Algebraic Multigrid, by K. Stüben, pp. 413–532.
[47] P. Vaněk, J. Mandel, and M. Brezina, *Algebraic multigrid based on smoothed aggregation for second and fourth order problems*, Computing, 56 (1996), pp. 179–196.
[48] P. S. Vassilevski and L. T. Zikatanov, *Multiple vector preserving interpolation mappings in algebraic multigrid*, SIAM J. Matrix Anal. Appl., 27 (2006), pp. 1040–1055.
[49] H. Yserentant, *On the multi-level splitting of finite element spaces*, Numer. Math., 58 (1986), pp. 379–412.
[50] H. Yserentant, *Hierarchical bases in the numerical solution of parabolic problems*, in Large Scale Scientific Computing, P. Deuflhard et al., eds., Birkhäuser Boston, Boston, 1987, pp. 22–36.

SIAM J. SCI. COMPUT.
Vol. 28, No. 2, pp. 728–750
© 2006 Society for Industrial and Applied Mathematics

EFFICIENT WEIGHTED LATTICE RULES WITH APPLICATIONS TO FINANCE*

XIAOQUN WANG† AND IAN H. SLOAN‡

Abstract. Good lattice rules are an important type of quasi-Monte Carlo algorithm. They are known to have good theoretical properties, in the sense that they can achieve an error bound (or optimal error bound) that is independent of the dimension for *weighted* function spaces with suitably decaying weights. To use the theory of weighted function spaces for practical applications, one has to determine *what weights should be used*. After explaining that lattice rules based on figures of merit with classical weights (classical lattice rules) may not give good results for high-dimensional problems, we propose a "matching strategy," which chooses the weights in such a way that the typical functions of a weighted Korobov space are similar (in the sense of similar relative sensitivity indices) to those of the given function. The matching strategy relates the weights of the spaces to the sensitivity indices of the given function. We apply the Korobov construction and the component-by-component construction of lattice rules with the suitably chosen weights to the valuation of high-dimensional financial derivative securities and find that such weighted lattice rules improve dramatically on the results for the classical lattice rules; moreover, they are competitive with other types of low discrepancy sequences. We also demonstrate that lattice rules combined with variance reduction techniques and dimension reduction techniques increase the efficiency significantly. We show that an acceptance-rejection approach to the construction of lattice rules can reduce the construction cost with almost no loss of accuracy.

Key words. quasi-Monte Carlo methods, lattice rules, multivariate integration, option pricing

AMS subject classifications. 65C05, 65D30, 65D32

DOI. 10.1137/S1064827502418197

1. Introduction.

High-dimensional integrals occur in various applications, such as physics, statistics, and financial mathematics. For example, from the theory of arbitrage-free pricing [13], the fair value of a European path-dependent option or multiasset option can be formulated as a multiple integral, with the dimension corresponding to the number of time steps in the time discretization or the number of assets. Monte Carlo (MC) and quasi-Monte Carlo (QMC) methods are now widely used for pricing complex financial derivatives (see [1, 2, 3, 21]). Consider the evaluation of a d-dimensional integral:

$$I_d(f) = \int_{[0,1]^d} f(\mathbf{x})\, d\mathbf{x}.$$

The MC estimate of the integral $I_d(f)$ takes the form

$$Q_{n,d}(f) = \frac{1}{n} \sum_{k=0}^{n-1} f(\mathbf{x}_k),$$

*Received by the editors November 20, 2002; accepted for publication (in revised form) December 16, 2005; published electronically May 26, 2006. This work was supported by the National Science Foundation of China and the Australian Research Council's Centre of Excellence Program.
http://www.siam.org/journals/sisc/28-2/41819.html

†Department of Mathematical Sciences, Tsinghua University, Beijing 100084, China, and School of Mathematics, University of New South Wales, Sydney 2052, Australia (xwang@math.tsinghua.edu.cn).

‡School of Mathematics, University of New South Wales, Sydney 2052, Australia (i.sloan@unsw.edu.au).

where the points $\mathbf{x}_0, \ldots, \mathbf{x}_{n-1}$ are independently and identically distributed (i.i.d.) random samples from the uniform distribution on $[0,1]^d$. It is well known that the MC method has a probabilistic convergence rate of order $O(n^{-1/2})$ independently of the dimension.

The QMC methods replace the random samples in MC by carefully chosen deterministic points, so as to yield better uniformity than the random samples. Let H be a Hilbert space of functions defined on $[0,1]^d$ with norm $||\cdot||_H$; then (see [9, 25])

$$|I_d(f) - Q_{n,d}(f)| \le e(Q_{n,d}; H)\, ||f||_H \qquad \forall f \in H,$$

where

$$e(Q_{n,d}; H) = \sup\{|I_d(f) - Q_{n,d}(f)| : f \in H,\ ||f||_H \le 1\} \tag{1.1}$$

is called the [illegible] of $Q_{n,d}(f)$ or the [illegible] of the point set $\{\mathbf{x}_k\}_{k=0}^{n-1}$.

To minimize the QMC error, one should construct a QMC algorithm such that its worst-case error in a suitable space H is as small as possible. There are two important classes of point sets that are well suited to multivariate integration: digital nets and good lattice rules [18, 22]. In general, the QMC methods have deterministic error bounds of the order $O(n^{-a}(\log n)^b)$ for functions satisfying some regularity conditions, where $a > 1/2$ and b is a constant in the order of d. This convergence is asymptotically better than that of MC. For weighted function spaces with fast decaying weights, it is possible to establish a QMC error bound that is independent of the dimension [11, 15, 23, 30].

Lattice rules (of rank 1) are QMC algorithms of the special form

$$Q_{n,d}(f) = \frac{1}{n} \sum_{k=0}^{n-1} f\left(\left\{\frac{k\mathbf{z}}{n}\right\}\right), \tag{1.2}$$

where $\mathbf{z} = (z_1, \ldots, z_d)$ is the generating vector of integers with no factor in common with n, and the braces around a vector indicate that each component of the vector is to be replaced by its fractional part. A central topic in the theory of lattice rules is to find a "good" generator $\mathbf{z}$. Several constructions are known, such as the Korobov construction [14] and the component-by-component (CBC) construction [24]. These constructions are considered in [illegible] function spaces, and some good theoretical properties of the resulting lattice rules have been revealed [15, 24, 35]. However, their usefulness for practical high-dimensional problems has not yet been established, the main difficulty being that these constructions depend on certain "weights." For practical applications, we need to know what weights should be used.

It has been observed that the functions in financial derivative pricing are often special, in that some variables are more important than others, and/or the lower-order "interactions" of the variables are more important than the higher-order ones (that is to say, the functions can often be well approximated by sums of low-dimensional functions); see [3, 34]. Many functions in other applications have similar properties. The classical lattice rules (i.e., the lattice rules with classical weights; see section 2) do not take advantage of such specific properties. While many high-dimensional financial problems have been successfully solved by digital nets (such as the Sobol points [28]; see [21]), the successful use of good lattice rules has been reported only in relatively low dimensions. In fact, in our experience (reported below) lattice rules based on figures of merit with classical weights may not give good results for high-dimensional

financial problems and indeed may give worse results than MC. By investigating the ANOVA decomposition, we shall see that the classical choice of weights not only treats all variables equally but also implicitly treats higher-order interactions between the variables as more important than the lower-order interactions. This unreasonable emphasis on the higher-dimensional interactions may lead to poor quality of low-dimensional projections of the lattice points, especially two-dimensional projections (see [31]), and is therefore inappropriate for the functions from finance. We believe that this is an important reason why classical lattice rules tend to fail for high-dimensional finance problems.

In this paper we make use of the so-called weighted spaces, introduced in [25] for Sobolev spaces and in [26] for (periodic) Korobov spaces. In weighted function spaces, the variables are associated with weights. These weights then enter the worst-case error, which is the criterion of goodness used to search for lattice rule generators. For this reason the determination of suitable weights is a central theme of this paper. Intuitively, for finance applications lattice rules need to concentrate on more important variables and on low-order interactions. Using different coordinate weights in the figure of merit was suggested in [7, 8].

The outline of this paper is as follows. In section 2, we introduce the weighted Korobov spaces and present the criterion of goodness for good lattice rules. In section 3, we propose a "matching strategy" to choose good weights, relating the weights to the sensitivity indices. In section 4, we present several constructions of lattice rules, including a variant of the constructive algorithm for finding the vector $\mathbf{z}$ in (1.2), that speeds the construction by introducing an acceptance-rejection technique. The effectiveness of various (weighted and unweighted) lattice rules for financial derivative pricing is investigated numerically in section 5. Section 6 presents the conclusions.

2. Weighted Korobov spaces and the quality measure for lattice rules. As in [26], we work in the framework of weighted Korobov spaces $H_{d,\alpha,\gamma}$ of functions defined in $[0,1)^d$. This is the Hilbert space of periodic integrable functions with absolutely convergent Fourier series and with the reproducing kernel $K_{d,\alpha,\gamma}$ given by

$$K_{d,\alpha,\gamma}(\mathbf{x},\mathbf{y}) = \prod_{j=1}^{d}\left(1+\gamma_j \sum_{h=-\infty}^{\infty}{}' \frac{e^{2\pi i h(x_j-y_j)}}{|h|^\alpha}\right), \quad i=\sqrt{-1}, \tag{2.1}$$

where $\{\gamma_j\}$ is a sequence of positive numbers (the weights). The prime on the sum indicates that the term $h=0$ is omitted. The smoothness parameter $\alpha > 1$ characterizes the rate of decay of the Fourier coefficients. If $\gamma_j = 1$, the space $H_{d,\alpha,\gamma}$ is the classical unweighted Korobov space. The space $H_{d,\alpha,\gamma}$ is equipped with the inner product

$$\langle f, g\rangle_{H_{d,\alpha,\gamma}} = \sum_{\mathbf{h}\in\mathbf{Z}^d}\prod_{j=1}^{d} r(\gamma_j,h_j)\ \hat{f}(\mathbf{h})\ \overline{\hat{g}(\mathbf{h})},$$

where $\mathbf{h}=(h_1,h_2,\ldots,h_d)$, $\mathbf{Z}=\{\ldots,-1,0,1,\ldots\}$,

$$r(\gamma,h) = \begin{cases} 1 & \text{if } h=0,\\ \gamma^{-1}|h|^\alpha & \text{if } h\neq 0,\end{cases}$$

and

$$\hat{f}(\mathbf{h}) = \int_{[0,1)^d} \exp(-2\pi i\mathbf{h}\cdot\mathbf{x}) f(\mathbf{x})\, d\mathbf{x}$$

is the Fourier coefficient of $f(\mathbf{x})$, with $\mathbf{h}\cdot\mathbf{x} = h_1x_1 + \cdots + h_dx_d$. Note that for a small weight γ_j, a function f with a norm at most 1 can depend only weakly on the jth variable. More general assumptions on the weights, and thus on the induced weighted spaces, have been studied; see [5, 6].

In the space $H_{d,\alpha,\gamma}$ the worst-case error defined by (1.1) is given by (see [26])

(2.2)
$$e_{n,d}(\mathbf{z}) := e(Q_{n,d}; H_{d,\alpha,\gamma}) = \left(-1 + \frac{1}{n}\sum_{k=0}^{n-1}\prod_{j=1}^{d}\left(1 + \gamma_j \sum_{h=-\infty}^{\infty}{}' \frac{e^{2\pi i h k z_j/n}}{|h|^\alpha}\right)\right)^{1/2}$$

for the case of the rank-1 lattice rule (1.2). The worst-case error $e_{n,d}(\mathbf{z})$ will be used as the criterion of goodness for lattice rules.

For the constructive algorithms to be discussed in section 4, it is important to be able to compute the worst-case error efficiently. For the case when α is an even integer, it is known that

$$\sum_{h=-\infty}^{\infty}{}' \frac{e^{2\pi i h x}}{|h|^\alpha} = \frac{(2\pi)^\alpha}{(-1)^{\frac{\alpha}{2}+1}\alpha!}\, B_\alpha(x), \quad x \in [0,1], \tag{2.3}$$

where $B_\alpha(x)$ is the Bernoulli polynomial of degree α. In particular, $B_2(x) = x^2 - x + 1/6$. Thus the infinite sum in the formula of the worst-case error (2.2) can be computed easily when $\alpha = 2$.

[illegible] 1. Note that if $\gamma_j = 1$ for all j, then $e^2_{n,d}(\mathbf{z})$ equals P_α, which is the classical criterion for lattice rules and is defined as (see [22])

$$P_\alpha = \sum_{\mathbf{h}\cdot\mathbf{z}\equiv 0 \bmod (n)}{}' \frac{1}{(\overline{h}_1 \cdots \overline{h}_d)^\alpha}, \tag{2.4}$$

where $\overline{h} := \max(1, |h|)$. In this paper, the term "classical lattice rules" is used to mean the lattice rules constructed by minimizing the classical quality measure P_α. Let $g(\mathbf{x}) = K_{d,\alpha,\gamma}(\mathbf{x}, \mathbf{0})$ with $\gamma_j = 1$ for all j; then P_α can be written as the quadrature error:

$$P_\alpha = Q_{n,d}(g) - I_d(g) = Q_{n,d}(g) - 1.$$

This observation is important for the analysis in section 3.

3. What weights should be used in practice? The construction of lattice rules in weighted Korobov spaces depends on the weights γ_j, since the weights enter the criterion of goodness. Such a situation is quite different from the constructions of digital sequences, such as the Sobol sequence [28] and the Niederreiter sequence [18]. However, the dependence of lattice rules on the weights is also good news: we have the possibility to control the weights based on information about the integrands, in order to make the constructed lattice rules more suited to our needs. An important question is, [illegible] Another is, [illegible]

The weights in a weighted Korobov space moderate the relative importance of successive variables and groups of variables. On the other hand, for a given function the relative importance of variables and groups of variables can be measured by Sobol's [illegible] [29], which are explained below. We shall relate the weights of the

space to the sensitivity indices of the given function, the basic idea being to choose the weights in such a way that "typical functions" (see section 3.2) of the weighted Korobov spaces have similar relative sensitivity indices to the given function. If this matching is successful, then we have good reason to expect that the resulting lattice rules will be suitable for the given function.

3.1. Sensitivity analysis for the given problems. The relative importance of each variable or group of variables can be measured by sensitivity indices, which are related to the ANOVA decomposition of functions. For any subset $u \subseteq \{1, \dots, d\}$, we use the notation $|u|$ for its cardinality and use $-u$ for its complementary set $\{1, \dots, d\} - u$. For $\mathbf{x} \in [0,1)^d$, let $\mathbf{x}_u$ be the $|u|$-dimensional vector containing the components of $\mathbf{x}$ whose indices are in u. Any square-integrable function $f(\mathbf{x})$ can be written as a sum of its ANOVA terms:

$$f(\mathbf{x}) = f_\emptyset + \sum_{j=1}^{d} f_{\{j\}}(x_j) + \sum_{1 \le i < j \le d} f_{\{i,j\}}(x_i, x_j) + \cdots + f_{\{1,\dots,d\}}(x_1, \dots, x_d), \tag{3.1}$$

where $f_\emptyset = I_d(f)$ and the terms $f_u(\mathbf{x})$ are defined recursively by

$$f_u(\mathbf{x}) = \int_{[0,1)^{d-|u|}} f(\mathbf{x}) d\mathbf{x}_{-u} - \sum_{v \subset u} f_v(\mathbf{x}),$$

where the sum in the last term is over strict subsets $v \neq u$. Each ANOVA term identifies the contribution of distinct group of variables to f. The ANOVA decomposition (3.1) is orthogonal, in that $\int_{[0,1]^d} f_u(\mathbf{x}) f_v(\mathbf{x}) d\mathbf{x} = 0$ whenever $u \neq v$. It follows that the variance of f can be expressed as

$$\sigma^2(f) = \sum_{\emptyset \neq u \subseteq \{1,\dots,d\}} \sigma_u^2(f),$$

where $\sigma_u^2(f) = \int_{[0,1]^d} [f_u(\mathbf{x})]^2 d\mathbf{x}$ for $|u| > 0$ is the variance of f_u, which indicates the contribution of the term f_u to the total variance $\sigma^2(f) = \int_{[0,1]^d} f^2(\mathbf{x}) d\mathbf{x} - f_\emptyset^2$.

[illegible] are defined as the ratios

$$S_u(f) = \frac{\sigma_u^2(f)}{\sigma^2(f)} \quad \text{for } \emptyset \neq u \subseteq \{1, \dots, d\}.$$

They are useful for measuring the relative importance of each ANOVA term. Clearly, $\sum_{\emptyset \neq u \subseteq \{1,\dots,d\}} S_u(f) = 1$. Letting u be a subset of $\{1, \dots, d\}$, the [illegible] corresponding to u is defined as

$$T_u(f) := \sum_{\emptyset \neq v \subseteq u} \sigma_v^2(f).$$

Writing $\mathbf{x} = (\mathbf{x}_u, \mathbf{x}_{-u})$ and $\mathbf{y} = (\mathbf{y}_u, \mathbf{y}_{-u})$, then it can be shown (see [29, 32]) that

$$T_u(f) = \int_{[0,1)^{2d-|u|}} f(\mathbf{x}) f(\mathbf{x}_u, \mathbf{y}_{-u}) d\mathbf{x} d\mathbf{y}_{-u} - [I_d(f)]^2. \tag{3.2}$$

This formula can be used to calculate numerically an approximation to the variance of each ANOVA term. Because our focus is on financial derivative pricing, where the corresponding functions are dominated by low-order ANOVA terms, we are mainly interested in the variances of low-order ANOVA terms. The variance of the first- and second-order ANOVA terms can be computed as follows:

- For $|u| = 1$, $\sigma_u^2(f) = T_u(f)$, so $\sigma_u^2(f)$ can be computed directly by (3.2).
- For $|u| = 2$, say, $u = \{i, j\}$, $\sigma_{\{i,j\}}^2(f)$ can be evaluated from

$$T_{\{i,j\}}(f) = \sigma_{\{i\}}^2(f) + \sigma_{\{j\}}^2(f) + \sigma_{\{i,j\}}^2(f),$$

where $T_{\{i,j\}}(f)$ can be computed by (3.2).

In this way, for a given function f one can calculate its sensitivity indices $S_u(f)$ for $|u| = 1, 2$ to obtain information about the relative importance of each variable and the pairwise interactions between variables. For our purpose we need only to compute several sensitivity indices.

3.2. ANOVA decomposition for weighted Korobov spaces. For a weighted reproducing kernel Hilbert space, it is also possible to obtain some information about the behavior of the functions of the space by analyzing typical functions in the space. Consider the weighted Korobov spaces with the smoothness parameter $\alpha = 2$. Based on (2.3), the reproducing kernel $K_{d,\alpha,\gamma}(\mathbf{x}, \mathbf{y})$ in (2.1) with $\alpha = 2$ can be written as

$$K_{d,2,\gamma}(\mathbf{x}, \mathbf{y}) = \prod_{j=1}^{d} \left[1 + 2\pi^2 \gamma_j B_2(\{x_j - y_j\})\right]. \tag{3.3}$$

For any fixed $\mathbf{y}^* = (y_1^*, \ldots, y_d^*) \in [0, 1)^d$, define a function of $\mathbf{x}$ by

$$f_{\mathbf{y}^*}(\mathbf{x}) := K_{d,2,\gamma}(\mathbf{x}, \mathbf{y}^*) = \prod_{j=1}^{d} \left[1 + 2\pi^2 \gamma_j B_2(\{x_j - y_j^*\})\right] =: \prod_{j=1}^{d} g_j(x_j). \tag{3.4}$$

We call $f_{\mathbf{y}^*}(\mathbf{x})$ a "typical function" of the weighted Korobov space $H_{d,2,\gamma}$ due to the following:

(1) Each $f_{\mathbf{y}^*}(\mathbf{x})$ belongs to the space $H_{d,2,\gamma}$ and has full dimension d.

(2) Any function of the space $H_{d,2,\gamma}$ is either a linear combination of the form $\sum a_k f_{\mathbf{y}_k^*}(\mathbf{x})$ for some choices of $\{a_k\}$ and $\{\mathbf{y}_k^*\}$ or the limit of a sequence of such combinations.

(3) The sensitivity indices for the function $f_{\mathbf{y}^*}(\mathbf{x})$ are independent of the choice of $\mathbf{y}^*$. (This fact, shown below, is a consequence of the translation invariance of the space.)

(4) The quadrature error of $f_{\mathbf{y}^*}(\mathbf{x})$ with the shifted lattice rule (see (5.3) below) with $\Delta = \mathbf{1} - \mathbf{y}^*$ is exactly the worst-case error in the space $H_{d,2,\gamma}$. This can be verified by using (2.2) and $I_d(f_{\mathbf{y}^*}) = 1$.

The function $f_{\mathbf{y}^*}(\mathbf{x})$ has a multiplicative structure. To exploit this, we note that, for any fixed $y_j^* \in [0, 1)$,

$$\int_0^1 B_2(\{x_j - y_j^*\}) dx_j = 0 \quad \text{and} \quad \int_0^1 B_2^2(\{x_j - y_j^*\}) dx_j = \frac{1}{180}.$$

Let

$$\mu_j := \int_0^1 g_j(x_j) dx_j = \int_0^1 \left[1 + 2\pi^2 \gamma_j B_2(\{x_j - y_j^*\})\right] dx_j = 1$$

and

$$\lambda_j^2 := \int_0^1 (g_j(x_j) - \mu_j)^2 dx_j = \int_0^1 4\pi^4 \gamma_j^2 B_2^2(\{x_j - y_j^*\}) dx_j = \frac{\pi^4}{45} \gamma_j^2.$$

Using the formulas in [32], the ANOVA terms of $f_{\mathbf{y}^*}(\mathbf{x})$ and the corresponding variances are

$$(f_{\mathbf{y}^*})_u = \prod_{j\in u}(g_j(x_j)-\mu_j)\prod_{j\notin u}\mu_j \quad \text{and} \quad \sigma_u^2(f_{\mathbf{y}^*}) = \left(\frac{\pi^4}{45}\right)^{|u|}\prod_{j\in u}\gamma_j^2. \tag{3.5}$$

The total variance of $f_{\mathbf{y}^*}(\mathbf{x})$ is

$$\sigma^2(f_{\mathbf{y}^*}) = \left[\prod_{j=1}^{d}\left(1+\frac{\pi^4\gamma_j^2}{45}\right)-1\right].$$

Therefore, the sensitivity indices of the typical function $f_{\mathbf{y}^*}$ can be computed easily. Note that all the variances $\sigma_u^2(f_{\mathbf{y}^*})$ and $\sigma^2(f_{\mathbf{y}^*})$ are independent of the point $\mathbf{y}^*$. Due to this fact, a typical function will also be denoted by $f^*(\mathbf{x})$. It is clear from (3.5) that the relative importance of the ANOVA terms strongly depends on the weights.

3.3. Why are classical lattice rules not suitable for high-dimensional financial problems? We explain why classical lattice rules may be [illegible] for high-dimensional financial problems. Consider the classical unweighted Korobov space with the weights $\gamma_j = 1$ for all j. For a typical function f^* in this space, from (3.5) we have

$$\sigma_u^2(f^*) = \left(\pi^4/45\right)^{|u|}.$$

Since $\pi^4/45 = 2.1646\cdots > 1$, the relative importance of a high-order ANOVA term of f^* is much more significant than that of a low-order term. Indeed, the sensitivity index of the ANOVA term f_u^*, namely

$$S_u(f^*) = \frac{1}{\sigma^2(f^*)}\left(\frac{\pi^4}{45}\right)^{|u|},$$

depends [illegible] on $|u|$ (with $\pi^4/45$ as the base). Moreover, the number of ANOVA terms with order $|u|$ is $\binom{d}{|u|}$, which is increasing for $|u| < d/2$. Thus a typical function f^* has large effective dimension in the superposition sense; see [32].

The classical quality measure of lattice rules P_α is the quadrature error of the lattice rule applied to a certain function; see (2.4). In particular, P_2 is the quadrature error of the function $K_{d,2,\gamma}(\mathbf{x},\mathbf{0}) = \prod_{j=1}^{d}\left[1+2\pi^2 B_2(x_j)\right]$ with all $\gamma_j = 1$ (see [22]), i.e.,

$$P_2 = Q_{n,d}(K_{d,2,\gamma}(\mathbf{x},\mathbf{0})) - I_d(K_{d,2,\gamma}(\mathbf{x},\mathbf{0})) = Q_{n,d}(K_{d,2,\gamma}(\mathbf{x},\mathbf{0})) - 1.$$

The function $K_{d,2,\gamma}(\mathbf{x},\mathbf{0})$ with all $\gamma_j = 1$ is a typical function in the space $H_{d,2,\gamma}$. As just shown, such a function has strong high-order ANOVA effects and relatively weak low-order ones (say the first- and second-order terms). Classical lattice rules are constructed by minimizing P_2. From the nature of the integrand $K_{d,2,\gamma}(\mathbf{x},\mathbf{0})$, by minimizing P_2 we are paying more attention to the high-order ANOVA terms and less attention to low-order terms. A lattice rule that minimizes P_2 may be the best one for the artificial space $H_{d,2,\gamma}$ (with all $\gamma_j = 1$), but the corresponding worst-case error is in general very large (see [22]). A recent investigation shows that the lattice rules found by minimizing P_2 have very bad two-dimensional projections [31]. (Note that all rank-1 lattices have the same one-dimensional projections.)

However, many practical problems have a quite different ANOVA structure from the function $K_{d,2,\gamma}(\mathbf{x},\mathbf{0})$. Quite commonly, the functions in finance are dominated by low-order ANOVA terms, often just the first- and second-order terms. While the third-order terms sometimes have a small influence on the function, higher-order terms are usually negligible. That is to say, an ANOVA expansion to the third-order

$$f(\mathbf{x}) \approx f_\emptyset + \sum_j f_{\{j\}}(x_j) + \sum_{i<j} f_{\{i,j\}}(x_i,x_j) + \sum_{i<j<k} f_{\{i,j,k\}}(x_i,x_j,x_k)$$

(or even to the second-order) often provides a quite satisfactory approximation of f (in the L_2-norm sense). It is therefore unreasonable to use the classical lattice rules for such financial problems. Indeed, it is quite possible that the classical lattice rules may lead to bad results, even worse than MC (see section 5 for examples). This is why it is important to choose more appropriate weights for practical applications in high dimensions.

3.4. Determining the weights by a matching strategy. We show how to determine empirical weights by matching the relative sensitivity indices of a typical function of a weighted Korobov space to the given function. Assuming that the variables $x_1, x_2, \ldots, x_d$ are labelled in the order of decreasing importance, for the given function we compute the approximate values of the variances of some ANOVA terms using the method in section 3.1:

$$v_{\{1\}}(f) \approx \sigma^2_{\{1\}}(f), \quad v_{\{2\}}(f) \approx \sigma^2_{\{2\}}(f), \quad v_{\{1,2\}}(f) \approx \sigma^2_{\{1,2\}}(f).$$

The relative sensitivities can then be obtained from

$$\frac{S_{\{2\}}(f)}{S_{\{1\}}(f)} = \frac{\sigma^2_{\{2\}}(f)}{\sigma^2_{\{1\}}(f)} \approx \frac{v_{\{2\}}(f)}{v_{\{1\}}(f)} \quad \text{and} \quad \frac{S_{\{1,2\}}(f)}{S_{\{1\}}(f)} = \frac{\sigma^2_{\{1,2\}}(f)}{\sigma^2_{\{1\}}(f)} \approx \frac{v_{\{1,2\}}(f)}{v_{\{1\}}(f)}. \tag{3.6}$$

The first ratio indicates the relative importance of x_1 and x_2, and the second ratio indicates the relative importance of the joint interaction of x_1 and x_2 to that of x_1 alone.

On the other hand, suppose that f^* is a "typical function" in the weighted Korobov space $H_{d,2,\gamma}$ and that the weights have the two-parameter form

$$\gamma_j = a\tau^j, \quad j = 1,2,\ldots,d, \tag{3.7}$$

where a and τ are parameters to be determined. Then from (3.5) the variance of the ANOVA term of f^* is

$$\sigma^2_u(f^*) = \left(\frac{\pi^4}{45}\right)^{|u|} \prod_{j\in u} \gamma_j^2 = \left(\frac{a^2\pi^4}{45}\right)^{|u|} \prod_{j\in u} \tau^{2j}. \tag{3.8}$$

From (3.8), we have

$$\frac{S_{\{2\}}(f^*)}{S_{\{1\}}(f^*)} = \frac{\sigma^2_{\{2\}}(f^*)}{\sigma^2_{\{1\}}(f^*)} = \tau^2 \quad \text{and} \quad \frac{S_{\{1,2\}}(f^*)}{S_{\{1\}}(f^*)} = \frac{\sigma^2_{\{1,2\}}(f^*)}{\sigma^2_{\{1\}}(f^*)} = \frac{\pi^4}{45}a^2\tau^4. \tag{3.9}$$

In the two-parameter [illegible] we choose the parameters a and τ, and hence the weights, by equating the respective ratios in (3.6) and (3.9). This yields

$$\tau = \sqrt{\frac{v_{\{2\}}(f)}{v_{\{1\}}(f)}}, \quad a = \frac{3}{\pi^2\tau^2}\sqrt{\frac{5v_{\{1,2\}}(f)}{v_{\{1\}}(f)}}. \tag{3.10}$$

The weights γ_j are then determined by $\gamma_j = a\tau^j$, $j = 1, \ldots, d$. For functions whose successive variables become less and less important and the interactions between variables are weaker than the main effects, one has $v_{\{2\}}(f) < v_{\{1\}}(f)$ and $v_{\{1,2\}}(f) < v_{\{1\}}(f)$, and thus one must have $\tau < 1$ and $a \le 3\sqrt{5}/\pi^2 \approx 0.68$ (implying the classical weights are much too large).

We can verify whether the weights determined in this way are reasonable for a given problem by comparing "predicted" sensitivity indices with the actual values. For example, for the particular problem, given $S_{\{1\}}(f)$ and the weights determined by $\gamma_j = a\tau^j$ we can predict the sensitivity indices of all ANOVA terms from the theory. Indeed, from (3.8) we have

$$S_{\{j\}}(f) = \sigma^2_{\{j\}}(f)/\sigma^2(f) = \tau^{2(j-1)} S_{\{1\}}(f), \quad j = 1, 2, \ldots, d, \tag{3.11}$$

and similarly

$$S_{\{i,j\}}(f) = \frac{\pi^4}{45} a^2 \tau^{2i+2j-2} S_{\{1\}}(f), \quad i = 1, \ldots, d-1; \quad j = i+1, \ldots, d. \tag{3.12}$$

On the other hand, we can compute their actual values for the given problem by the method given in section 3.1:

$$S_{\{j\}}(f) = \frac{\sigma^2_{\{j\}}(f)}{\sigma^2(f)} \approx \frac{v_{\{j\}}(f)}{\hat{\sigma}^2(f)}, \quad j = 1, \ldots, d,$$

and

$$S_{\{i,j\}}(f) = \frac{\sigma^2_{\{i,j\}}(f)}{\sigma^2(f)} \approx \frac{v_{\{i,j\}}(f)}{\hat{\sigma}^2(f)}, \quad i = 1, \ldots, d-1; \quad j = i+1, \ldots, d,$$

where $\hat{\sigma}^2(f) \approx \sigma^2(f)$. If the predicted values fit the actual values well, we have good reason to expect that the typical functions of the weighted Korobov space (with the weights so determined) match the given function well and that it is reasonable to use these weights for constructing lattice rules. We describe an experiment of this kind in section 5.

2. The use of the weights in the form (3.7) is motivated by the observation that in some financial problems [32, 34], the variances captured by the successive variables are naturally ordered and decrease approximately exponentially (we use τ to characterize this); moreover, the high-order interactions are weaker than the low-order interactions (here a characterizes the relative importance of ANOVA terms of different orders: small a means that only low-order terms are important). Such weights are easy to work with, since only two parameters, a and τ, need to be determined. Note that it would be more robust to base the estimation of the parameters on $v_u(f)$ for all u of cardinality one and two, but this would require much more computation.

The matching strategy can also be used to determine more general weights γ_j. Define the relative weights $\tau_j := \gamma_j/\gamma_1$. It is then convenient to write the weights as

$$\gamma_j = a\,\tau_j, \quad j = 1, \ldots, d, \quad \text{with} \quad \tau_1 = 1.$$

The parameters a and τ_j $(j = 2, \ldots, d)$ can also be determined by a matching strategy:

$$\tau_j = \sqrt{\frac{v_{\{j\}}(f)}{v_{\{1\}}(f)}}, \quad j = 2, \ldots, d; \quad a = \frac{3\sqrt{5}}{\pi^2} \sqrt{\frac{\sum_{i<j} v_{\{i,j\}}(f)}{\sum_j v_{\{j\}}(f)} \, \frac{\sum_j \tau_j^2}{\sum_{i<j} \tau_i^2 \tau_j^2}},$$

where $v_{\{j\}}(f)$ and $v_{\{i,j\}}(f)$ are the approximate values of $\sigma_j^2(f)$ and $\sigma_{\{i,j\}}^2(f)$, respectively.

A simplification of (3.7) is the one-parameter form of weights: $\gamma_j = a$ for $j = 1, \ldots, d$ (obtained by letting $\tau = 1$). This choice is suitable for functions with (nearly) equally important variables. At the other extreme, one could associate each "important" ANOVA term with its own weight and associate the unimportant terms with the weight 0. The weights are then of the nonproduct form studied in [6].

3. If we are completely successful with the matching, i.e., if the weights γ_j are such that all the sensitivity indices $S_u(f^*)$ for $u \subseteq \{1, \ldots, d\}$ of the typical functions f^* of the form (3.4) are the same as those of the given function f or, at the extreme, that the given f is actually one function of the form (3.4), for example, $f(\mathbf{x}) = \prod_{j=1}^{d}[1 + 2\pi^2\gamma_j B_2(x_j)]$, then the lattice rule constructed by minimizing the worst-case error of the corresponding weighted Korobov space with the weights γ_j is the best choice for the given f. This is because, as mentioned earlier, the worst-case error for that weighted Korobov space is exactly the quadrature error for the given function f.

4. Typically, accurate estimation of the sensitivity indices for a practical problem is no easier than the original integration problem, because the formula (3.2) involves high-dimensional integrals. For practical use, we can take a "simpler" problem which is similar to the original one but with the property that computation of the corresponding sensitivity indices is easier, or even analytically tractable. It then seems plausible that the weights and the lattice rules found for the simpler problem will be useful for the original problem (this is seen to be the case for the examples in section 5). Another situation is that we have many integrals of a similar type to compute, with these integrals often differing from each other by differing values of some parameters. If we can show that the choice of weights or the quality of the resulting lattice rules is not very sensitive to the parameters lying in some ranges, then the weights (and the lattice rules) found for one particular integral may be useful for other integrals in the class. In our experience, even a rough estimation of good weights can bring a large benefit. See Remark 5 below for more discussion.

3.5. Nonperiodic function spaces. The functions encountered in practice are generally not periodic, and hence do not lie in the weighted Korobov spaces. However, it is known that when dealing with randomly shifted lattice rules in nonperiodic reproducing kernel Hilbert spaces, it is the "shift-invariant" kernel that is relevant to the error analysis (see [23]). For a reproducing kernel $K(\mathbf{x}, \mathbf{y})$, the shift-invariant kernel K^{sh} associated with K is defined by (see [12])

$$K^{\mathrm{sh}}(\mathbf{x}, \mathbf{y}) := \int_{[0,1]^d} K(\{\mathbf{x} + \Delta\}, \{\mathbf{y} + \Delta\})\, d\Delta.$$

An important property of the shift-invariant kernel is

$$\int_{[0,1]^d} e^2(Q_{n,d}(f, \Delta); H(K))\, d\Delta = e^2(Q_{n,d}; H(K^{\mathrm{sh}})),$$

where $H(K)$ and $H(K^{\mathrm{sh}})$ are the reproducing kernel Hilbert spaces with kernels K and K^{sh}, respectively. Therefore, a good lattice rule for $H(K^{\mathrm{sh}})$ corresponds to a good randomly shifted lattice rule for the space $H(K)$.

In some important cases, the shift-invariant kernel is a weighted Korobov kernel. Consider the Sobolev space of nonperiodic functions with the reproducing kernel

(see [27])

$$K(\mathbf{x},\mathbf{y})=\prod_{j=1}^{d}\left[1+w_j\left(\frac{1}{2}B_2(\{x_j-y_j\})+\left(x_j-\frac{1}{2}\right)\left(y_j-\frac{1}{2}\right)\right)\right].$$

Its shift-invariant kernel can be obtained by direct computation

$$K^{\rm sh}(\mathbf{x},\mathbf{y})=\prod_{j=1}^{d}[1+2\pi^2\gamma_j B_2(\{x_j-y_j\})],\quad \gamma_j=\frac{w_j}{2\pi^2},$$

which is exactly the kernel $K_{d,2,\gamma}(\mathbf{x},\mathbf{y})$ in (3.3). Next, consider the Sobolev space used in [25]:

$$K(\mathbf{x},\mathbf{y})=\prod_{j=1}^{d}(1+w_j\min(1-x_j,1-y_j)). \tag{3.13}$$

In [26] it is shown that its shift-invariant kernel is

$$K^{\rm sh}(\mathbf{x},\mathbf{y})=\prod_{j=1}^{d}\left(1+\frac{w_j}{3}\right)\prod_{j=1}^{d}[1+2\pi^2\gamma_j B_2(\{x-y\})]$$

with

$$\gamma_j=\frac{w_j}{2\pi^2(1+w_j/3)}.$$

Once more, apart from a constant factor, the shift-invariant kernel is just $K_{d,2,\gamma}(\mathbf{x},\mathbf{y})$ in (3.3) with γ_j given as above. That constant factor is irrelevant for our present purpose. Therefore, by estimating γ_j (using the matching strategy) we may, for example, take the point of view that we are simultaneously estimating the weights

$$w_j=\frac{2\pi^2\gamma_j}{1-2\pi^2\gamma_j/3}$$

in the setting of [25], i.e., for the Sobolev space kernel (3.13).

4. Constructing the lattice rules. In dimension $d\ge 3$ one normally finds lattice rules by computer searches based on a certain criterion of goodness, say the worst-case error. In this section, we present several constructions, including the Korobov construction [14, 35], the CBC construction [15, 23, 24], and a cheaper modification of the CBC construction. We consider the case when n is prime. In this case, each component of the generating vector $\mathbf{z}$ of the lattice rule (1.2) can be chosen from the restricted set

$$\mathbf{Z}_n=\{1,2,\ldots,n-1\}.$$

ALGORITHM 1 (Korobov construction).

Suppose n is prime. For each fixed dimension d and given weights $\{\gamma_j\}$, the optimal Korobov generator is found by minimizing

$$e^2_{n,d}(\mathbf{z}(a))=-1+\frac{1}{n}\sum_{k=0}^{n-1}\prod_{j=1}^{d}\left(1+\gamma_j\sum_{h=-\infty}^{\infty}{}'\,\frac{\exp\left(2\pi ikha^{j-1}/n\right)}{|h|^{\alpha}}\right)$$

with respect to $a \in \mathbf{Z}_n$, where $\mathbf{z}(a)$ is a vector in Korobov form $\mathbf{z}(a) = (1, a, \ldots, a^{d-1}) \bmod (n)$.

ALGORITHM 2 (CBC full search algorithm).

Suppose n is prime and the weights are given. The generator $\mathbf{z}$ is found as follows:

1. Set the first component z_1 of the generator $\mathbf{z}$ to 1.
2. For $t = 2, 3, \ldots, d$, find $z_t \in \mathbf{Z}_n$ such that

$$e_{n,t}^2(z_1, \ldots, z_t) = -1 + \frac{1}{n} \sum_{k=0}^{n-1} \prod_{j=1}^{t} \left(1 + \gamma_j \sum_{h=-\infty}^{\infty}{}' \frac{e^{2\pi i h k z_j/n}}{|h|^\alpha}\right)$$

is minimized.

It is proved in [15] that the resulting lattice rule satisfies

$$e_{n,d}(\mathbf{z}) \le C(d,\delta)\, n^{-\alpha/2+\delta}, \quad 0 < \delta \le (\alpha-1)/2,$$

where $C(d,\delta)$ depends on d and δ. Moreover, if $\sum_{j=1}^{\infty} \gamma_j^{\frac{1}{\alpha-2\delta}} < \infty$, then $C(d,\delta)$ is uniformly bounded in d. The good theoretical properties indicate the potential superiority of such lattice rules. A disadvantage of Algorithm 2 is its full search character in finding each component. It is desirable to speed up the construction. Here we take an approach which allows an incomplete search at some steps.

4.1. Acceptance-rejection control algorithm. Given a d-dimensional generator $\mathbf{z} = (z_1, \ldots, z_d)$, Algorithm 2 indicates a way of finding the next component by full search. We give here a more flexible way. For any $z_{d+1} \in \mathbf{Z}_n$, we have

$$e_{n,d+1}^2(\mathbf{z}, z_{d+1}) = -1 + \frac{1}{n} \sum_{k=0}^{n-1} \prod_{j=1}^{d+1} \left(1 + \gamma_j \sum_{h=-\infty}^{\infty}{}' \frac{e^{2\pi i h k z_j/n}}{|h|^\alpha}\right).$$

Define the mean of $e_{n,d+1}^2(\mathbf{z}, z_{d+1})$ over $z_{d+1} \in \mathbf{Z}_n$ as

$$M_{n,d,\alpha}(\mathbf{z}) := \frac{1}{n-1} \sum_{z_{d+1} \in \mathbf{Z}_n} e_{n,d+1}^2(\mathbf{z}, z_{d+1}).$$

An explicit formula for $M_{n,d,\alpha}(\mathbf{z})$ can be derived by using a similar method to that in [15, 23, 24]. The proof is omitted.

LEMMA 4.1. [illegible] $\alpha > 1$ [illegible] $\mathbf{z} = (z_1, \ldots, z_d) \in \mathbf{Z}_n^d$ [illegible] n[illegible]

$$M_{n,d,\alpha}(\mathbf{z}) = e_{n,d}^2(\mathbf{z}) + \frac{2\gamma_{d+1}\zeta(\alpha)}{n} \prod_{j=1}^{d}(1 + 2\gamma_j\zeta(\alpha)) + \gamma_{d+1} T^* \left(e_{n,d}^2(\mathbf{z}) + 1 - \frac{1}{n}\prod_{j=1}^{d}(1 + 2\gamma_j\zeta(\alpha))\right),$$

[illegible] ζ[illegible] $\zeta(\alpha) = \sum_{h=1}^{\infty} h^{-\alpha}$ [illegible] $T^* = -\frac{2\zeta(\alpha)(1-n^{1-\alpha})}{n-1}$

The mean $M_{n,d,\alpha}(\mathbf{z})$ can be used as a benchmark in searching the next component (see Algorithm 3). In our experience, an optimal component may be expensive to find, but nearly optimal components are not rare; indeed, they are quite common. Thus at least for some components (for example, the components associated with small weights), the full search is unnecessary. Instead, we can perform a random search.

In order to preserve as far as possible the good theoretical properties, we use an acceptance-rejection control search.

ALGORITHM 3 (acceptance-rejection control search algorithm).

Suppose n is prime and the weights γ_j are given. Set two integer parameters $2 \le d^* \le d$ and $1 \le m^* \le n-1$. The generator $(z_1, \ldots, z_d)$ is found as follows:

1. Set the first component z_1 of the generator $\mathbf{z}$ to 1.
2. For $t = 2, \ldots, d^* - 1$, find $z_t \in \mathbf{Z}_n$, such that $e^2_{n,t}(z_1, \ldots, z_t)$ is minimized.
3. For $t = d^*, \ldots, d$, the component z_t is found in the following way:
   (a) Randomly generate $z_t \in \mathbf{Z}_n$. If $e^2_{n,t}(z_1, \ldots, z_t) \le M_{n,t-1,\alpha}(z_1, \ldots, z_{t-1})$, accept this z_t as one candidate; otherwise, reject it.
   (b) Repeat step (a) until the number of acceptances is m^* or all possible values of $z_t \in \mathbf{Z}_n$ are exhausted.
   (c) Choose the best one from the accepted z_t's in the sense of smallest $e_{n,t}(z_1, \ldots, z_t)$.

The search is controlled by two parameters d^* and m^*. The first $d^* - 1$ components are found in the same way as in Algorithm 2. The remaining components are found by using the acceptance-rejection search; m^* is the maximum number of acceptances. We have the flexibility to choose these parameters according to the computing resource or the problem at hand. By choosing $d^* = d + 1$ or m^* sufficiently large, Algorithm 3 becomes equivalent to Algorithm 2. The lattice rules generated by Algorithm 3 partially preserve the good theoretical properties of the rules generated by Algorithm 2. We have convergence order $O(n^{-1/2})$, the same as MC, but the practical performance can be much better than MC.

THEOREM 4.2. [illegible] 3

$$e^2_{n,d}(\mathbf{z}) \le \frac{1}{n} \prod_{j=1}^{d} (1 + 4\gamma_j \zeta(\alpha)). \tag{4.1}$$

[illegible] $\sum_{j=1}^{\infty} \gamma_j < \infty$ [illegible] $O(n^{-1/2})$ [illegible]

The result follows easily by proving recursively that at each step the squared worst-case error $e^2_{n,d}(z_1, \ldots, z_t)$ is no worse than the average $M_{n,t-1,\alpha}(z_1, \ldots, z_{t-1})$ and that

$$M_{n,t-1,\alpha}(z_1, \ldots, z_{t-1}) \le \frac{1}{n} \prod_{j=1}^{t} (1 + 4\gamma_j \zeta(\alpha)).$$

The error bound (4.1) then follows.

Among the three algorithms, Algorithm 1 (the Korobov construction) is the cheapest one, while Algorithm 2 (full CBC construction) is the most expensive. The cost of Algorithm 1 for a fixed dimension d is $O(n^2 d)$, while the costs of the obvious implementations of Algorithms 2 and 3 for all dimensions up to d are $O(n^2 d^2)$ and $O(n^2 d d^* + n m^* d^2)$, respectively. (Note added in proof: Since this paper was written, Nuyens and Cools [19] have shown that there is an algorithm equivalent to Algorithm 2 whose cost is reduced to $O(n \log n\, d^2)$.) Although the theoretical properties of the lattice rules generated by Algorithm 1 (see also [35]) and Algorithm 3 are not as good as those of the lattice rules from Algorithm 2, their practical performance is similar (see section 5).

5. Financial derivative pricing.

5.1. Path-dependent options. Consider the problem of pricing path-dependent options with terminal payoff $g(S_{t_1}, \ldots, S_{t_d})$, where $S_{t_1}, \ldots, S_{t_d}$ are the prices of the underlying asset at times $t_1, \ldots, t_d$. For simplicity, assume the prices are sampled at equally spaced times $t_0 = 0$, $t_j = t_{j-1} + \Delta t$, $j = 1, \ldots, d$, $\Delta t = T/d$, where T is the expiration date. Assume that under the risk-neutral measure the underlying asset follows geometric Brownian motion (Black–Scholes model):

$$dS_t = rS_t dt + \sigma S_t dB_t, \tag{5.1}$$

where r is the risk-free interest rate, σ is the volatility, and B_t is the standard Brownian motion. The analytical solution to (5.1) is

$$S_t = S_0 \exp((r - \sigma^2/2)t + \sigma B_t). \tag{5.2}$$

Thus simulating values of S_t reduces to simulating values of B_t. Based on the risk-neutral valuation principle [13], the value of the option at time $t = 0$ is

$$C_0 = E_Q[e^{-rT} g(S_{t_1}, \ldots, S_{t_d})],$$

where $E_Q[\cdot]$ is the expectation under the risk-neutral measure Q.

Note that the random vector $(B_{t_1}, \ldots, B_{t_d})^T =: (y_1, \ldots, y_d)^T$ is normally distributed with mean zero and covariance matrix $\mathbf{V} = (\min(t_i, t_j))_{i,j=1}^d$. Defining $\mu_j = \log S_0 + (r - \sigma^2/2)t_j$, $j = 1, \ldots, d$, from (5.2) we have $S_{t_j} = \exp(\mu_j + \sigma y_j)$. The payoff can then be written as

$$g(S_{t_1}, \ldots, S_{t_d}) = g(e^{\mu_1 + \sigma y_1}, \ldots, e^{\mu_d + \sigma y_d}) =: H(\mathbf{y}).$$

The value of the option at time 0 can be written as

$$\begin{aligned} E_Q[e^{-rT} g(S_{t_1}, \ldots, S_{t_d})] &= \frac{e^{-rT}}{(2\pi)^{d/2}\sqrt{\det V}} \int_{R^d} H(\mathbf{y}) e^{-\frac{1}{2}\mathbf{y}^T \mathbf{V}^{-1} \mathbf{y}} d\mathbf{y} \\ &= \frac{e^{-rT}}{(2\pi)^{d/2}} \int_{R^d} H(A\mathbf{z}) e^{-\frac{1}{2}\mathbf{z}^T \mathbf{z}} d\mathbf{z}, \end{aligned}$$

where in the second step a linear transformation $(y_1, \ldots, y_d)^T = A(z_1, \ldots, z_d)^T$ is performed using an arbitrary $d \times d$ matrix A satisfying $AA^T = \mathbf{V}$. The change of variables can also be interpreted as a covariance matrix decomposition or a way of generating the Brownian motion:

$$(B_{t_1}, \ldots, B_{t_d})^T = A(z_1, \ldots, z_d)^T,$$

where $z_1, \ldots, z_d$ are i.i.d. standard normal variables. The standard construction generates the Brownian motion sequentially in time: given $B_0 = 0$,

$$B_{t_j} = B_{t_{j-1}} + \sqrt{\Delta t}\, z_j, \quad j = 1, \ldots, d.$$

In this case, the corresponding matrix A is the Cholesky decomposition of $\mathbf{V}$.

Now consider Asian call options based on the arithmetic average of the underlying asset. The terminal payoff is

$$g_A(S_{t_1}, \ldots, S_{t_d}) = \max\left(0, \frac{1}{d}\sum_{j=1}^{d} S_{t_j} - K\right),$$

where K is the strike price at time T. To improve the efficiency, as an option in our calculations we take the payoff of the corresponding geometric average Asian call option as a control variate: $g_G(S_{t_1},\dots,S_{t_d}) = \max(0, \prod_{j=1}^{d} S_{t_j}^{1/d} - K)$. An analytical expression can be easily derived for $E_Q[g_G(S_{t_1},\dots,S_{t_d})]$. Writing

$$E_Q[g_A(S_{t_1},\dots,S_{t_d})] = E_Q[g_A(S_{t_1},\dots,S_{t_d}) - bg_G(S_{t_1},\dots,S_{t_d})] + bE_Q[g_G(S_{t_1},\dots,S_{t_d})]$$

for some multiplier b, the optimal b in the MC setting, in the sense of minimizing the variance of $g_A - bg_G$, is $b^* = \mathrm{Cov}(g_A, g_G)/\sigma^2(g_G)$, which can be estimated numerically. The same multiplier b^* will also be used in the QMC setting (see [10] for the possible dangers). We will also use optionally antithetic variates and their combinations with control variates.

To reduce the effective dimension, we use the Brownian bridge (BB) [3, 17] or principal component analysis (PCA); see [1]. The advantage of BB and PCA is that they will match the most significant parts of the Brownian motion with the coordinates of the QMC points that have the highest quality. See [32] on the determination of the effective dimension and the extent to which the effective dimension can be reduced by BB and PCA.

We divide our experiments into two parts: one is to determine suitable weights; the other is to construct the corresponding lattice rules and compare their performance to other methods. In our calculations, we use the following parameters:

$$S_0 = 100, \quad \sigma = 0.2, \quad r = 0.1, \quad T = 1 \text{ year}, \quad K = 100.$$

The exact estimation of the sensitivity indices for the arithmetic average Asian option problem is not easier than the original problem, due to the high dimensionality of the integrals involved (see (3.2)). Thus to estimate the sensitivity indices, we take the simpler problem of the geometric average Asian option. The sensitivity indices can be computed easily as shown in [34]. For the standard construction they are

$$S_{\{1\}} = 1.44 \times 10^{-1}, \ S_{\{2\}} = 1.26 \times 10^{-1}, \ S_{\{1,2\}} = 8.85 \times 10^{-3} \quad \text{for } d = 16,$$

$$S_{\{1\}} = 3.76 \times 10^{-2}, \ S_{\{2\}} = 3.64 \times 10^{-2}, \ S_{\{1,2\}} = 7.42 \times 10^{-4} \quad \text{for } d = 64.$$

The parameters a, τ are determined by (3.10), resulting in $\tau = 0.93$ and $a = 0.20$ for $d = 16$ and $\tau = 0.98$ and $a = 0.098$ for $d = 64$. The weights γ_j can be determined by (3.7). Tables 5.1 and 5.2 present the comparison of the predicted sensitivity indices (computed by (3.11) and (3.12)) and the actual values of the sensitivity indices of some first- and second-order ANOVA terms for $d = 64$. The predicted values fit the actual ones surprisingly well (both the predicted and the actual values for higher-order terms are very small; we do not list them).

We will use the obtained weights to construct lattice rules by Algorithms 1–3 (setting $d^* = 10$ and $m^* = 200$ in Algorithm 3) and use the resulting lattice rules for arithmetic average Asian option problems. We compare their performance with MC and QMC based on the Sobol sequence [28]. The performance of the unweighted lattice rules (i.e., with all $\gamma_j = 1$) will also be compared. In order to get an error estimate for each method, the standard deviations are computed. We use random shifts for lattice rules and the digit-scrambling method [20, 33] for the Sobol sequence. A randomly shifted lattice rule takes the form (see [4])

$$Q_{n,d}(f, \Delta) = \frac{1}{n} \sum_{k=0}^{n-1} f\left(\left\{ \frac{k\mathbf{z}}{n} + \Delta \right\} \right), \tag{5.3}$$

TABLE 5.1

The sensitivity indices $S_{\{j\}} \times 100$, $j = 1, \ldots, 32$, of some first-order ANOVA terms for geometric average Asian option pricing problem ($d = 64$): predicted values vs. actual values (the actual values are in parentheses). The predicted values are computed by (3.11) *and* (3.12)*, using the values of a and τ from* (3.10) *and the computed value of $S_{\{1\}}$.*

	The sensitivity indices of first-order ANOVA terms $S_{\{j\}} \times 100$							
$j = 1, \ldots, 8$	3.76	3.64	3.53	3.42	3.31	3.21	3.10	3.00
	(3.76)	(3.64)	(3.52)	(3.41)	(3.30)	(3.20)	(3.08)	(2.97)
$j = 9, \ldots, 16$	2.91	2.82	2.73	2.65	2.56	2.48	2.41	2.33
	(2.87)	(2.77)	(2.67)	(2.59)	(2.47)	(2.38)	(2.28)	(2.19)
$j = 17, \ldots, 24$	2.26	2.19	2.12	2.05	1.99	1.93	1.87	1.81
	(2.10)	(2.02)	(1.93)	(1.85)	(1.77)	(1.69)	(1.61)	(1.53)
$j = 25, \ldots, 32$	1.75	1.70	1.64	1.59	1.54	1.49	1.45	1.40
	(1.46)	(1.39)	(1.32)	(1.25)	(1.12)	(1.11)	(1.05)	(0.99)

TABLE 5.2

The same as Table 5.1 *but for sensitivity indices $S_{\{i,j\}} \times 10^4$ of some second-order ANOVA terms.*

	The sensitivity indices of second-order ANOVA terms $S_{\{i,j\}} \times 10^4$							
	$j = 2$	$j = 3$	$j = 4$	$j = 5$	$j = 6$	$j = 7$	$j = 8$	$j = 9$
$i = 1$	7.42	7.18	6.96	6.74	6.53	6.32	6.13	5.93
	(7.42)	(7.20)	(7.00)	(6.78)	(6.58)	(6.37)	(6.17)	(5.97)
$i = 2$		6.96	6.74	6.53	6.32	6.13	5.93	5.75
		(7.00)	(6.80)	(6.60)	(6.40)	(6.20)	(5.99)	(5.80)
$i = 3$			6.53	6.32	6.13	5.93	5.75	5.57
			(6.60)	(6.41)	(6.21)	(6.00)	(5.82)	(5.63)
$i = 4$				6.13	5.93	5.75	5.57	5.39
				(6.22)	(6.02)	(5.84)	(5.65)	(5.47)

where Δ is chosen from a uniform distribution on $[0,1)^d$. We take m replications of $Q_{n,d}(f,\Delta)$ and compute

$$\overline{Q}_{n,d}(f) = \frac{1}{m}\sum_{j=1}^{m} Q_{n,d}(f,\Delta_j),$$

where $\Delta_1, \ldots, \Delta_m$ are i.i.d. samples. The square root of the sample variance

$$\mathrm{Var}[\overline{Q}_{n,d}(f)] = \frac{1}{m(m-1)}\sum_{j=1}^{m}[Q_{n,d}(f,\Delta_j) - \overline{Q}_{n,d}(f)]^2$$

provides an estimation of the standard deviation of $\overline{Q}_{n,d}(f)$. The relative efficiency ratio of $\overline{Q}_{n,d}(f)$ with respect to crude MC is defined by $\mathrm{Eff}\,[\overline{Q}_{n,d}(f)] := \mathrm{Var}(Q_{\mathrm{MC}})/\mathrm{Var}[\overline{Q}_{n,d}(f)]$, where $\mathrm{Var}(Q_{\mathrm{MC}})$ is the sample variance of the crude MC estimate based on mn random points. The relative efficiency ratio is the factor by which the number of points in crude MC would need to be multiplied to achieve the same accuracy. Tables 5.3 and 5.4 show the standard deviations and relative efficiency ratios of various estimates with respect to crude MC. The results are discussed in section 5.3.

TABLE 5.3

Comparison of the standard deviations and relative efficiency ratios with respect to crude MC (relative efficiency ratios are in parentheses) for the arithmetic average Asian option with 30 *replications (*$d = 16$*,* $n = 4001$*; for the Sobol sequence* $n = 4096$*). For weighted lattice rules, the weights are* $\gamma_j = a\tau^j$ *with* $\tau = 0.93$ *and* $a = 0.20$*. The optimal multiplier* b^* *in the control variate is* $b^* \approx 1.034$*. The abbreviations are as follows: KBV—Korobov lattice rules; CBC—full search component-by-component construction; ARC—acceptance-rejection control search algorithm. The three blocks of rows correspond to different methods of path generation: Std—standard construction of Brownian motion; BB—Brownian bridge construction; PCA—principal component analysis. In the finer stratification by rows are the methods of variance reduction: Crude—crude estimate without variance reduction; AV—antithetic variates; CV—control variates; ACV—combination of AV and CV.*

				Unweighted lattice rules			Weighted lattice rules		
		MC	Sobol	KBV	CBC	ARC	KBV	CBC	ARC
Std	Crude	2.58e-2	1.74e-2	5.71e-2	3.54e-2	4.83e-2	5.05e-3	3.36e-3	4.73e-3
		(1.00)	(2.2)	(0.21)	(0.53)	(0.29)	(26)	(59)	(30)
	AV	1.09e-2	1.63e-2	5.83e-2	3.57e-2	4.86e-2	5.05e-3	3.58e-3	4.25e-3
		(5.6)	(2.5)	(0.20)	(0.52)	(0.28)	(26)	(52)	(37)
	CV	7.64e-4	9.89e-4	3.46e-3	6.51e-3	5.23e-3	3.55e-4	3.48e-4	3.26e-4
		(1143)	(682)	(56)	(16)	(24)	(5291)	(5524)	(6272)
	ACV	5.66e-4	6.54e-4	3.15e-3	6.00e-3	5.38e-3	2.37e-4	2.49e-4	2.82e-4
		(2086)	(1561)	(67)	(19)	(23)	(11886)	(10756)	(8417)
BB	Crude	2.59e-2	1.33e-3	3.36e-2	1.11e-2	1.14e-2	1.49e-3	1.43e-3	1.71e-3
		(0.99)	(376)	(0.59)	(5.4)	(5.1)	(301)	(322)	(228)
	AV	1.10e-2	7.63e-4	3.25e-2	1.11e-2	1.14e-2	1.41e-3	1.01e-3	1.25e-3
		(5.5)	(1146)	(0.63)	(5.4)	(5.1)	(335)	(650)	(425)
	CV	7.75e-4	2.23e-4	4.51e-3	1.38e-3	1.31e-3	1.83e-4	1.29e-4	1.28e-4
		(1111)	(13477)	(33)	(348)	(387)	(20039)	(39959)	(40931)
	ACV	5.68e-4	1.92e-4	4.17e-3	1.14e-3	1.16e-3	1.45e-4	1.00e-4	1.06e-4
		(2068)	(18117)	(38)	(511)	(492)	(31821)	(65759)	(59253)
PCA	Crude	2.60e-2	1.02e-3	9.96e-3	1.01e-3	9.89e-4	6.51e-4	6.95e-4	6.92e-4
		(0.99)	(643)	(6.7)	(654)	(682)	(1575)	(1381)	(1395)
	AV	1.10e-2	8.38e-5	7.82e-3	9.23e-4	8.69e-4	2.13e-4	1.58e-4	1.66e-4
		(5.5)	(95095)	(11)	(783)	(884)	(14666)	(26588)	(24192)
	CV	7.71e-4	7.02e-5	2.43e-3	2.72e-4	3.21e-4	9.36e-5	9.63e-5	9.48e-5
		(1122)	(135640)	(113)	(8995)	(6491)	(76117)	(71910)	(74204)
	ACV	5.68e-4	3.93e-5	2.03e-3	1.75e-4	1.74e-4	9.08e-5	7.94e-5	8.04e-5
		(2072)	(431810)	(162)	(21844)	(22015)	(81033)	(105980)	(103100)

5.2. Pricing multiasset options. Consider a European-style multiasset derivative whose terminal payoff is $\phi(S_T^1, \dots, S_T^d)$, where $S_T^1, \dots, S_T^d$ are the prices of d risky assets at time T. Assume that in the risk-neutral measure the prices $(S_t^1, \dots, S_t^d)$ of the assets satisfy

$$dS_t^j = rS_t^j dt + \sigma_j S_t^j dB_t^j, \quad j = 1, \dots, d,$$

where $\sigma_1, \dots, \sigma_d$ are volatility parameters and $B_t^1, \dots, B_t^d$ are correlated Brownian motions with correlations ρ_{ij} (each B_t^j has drift 0 and variance 1). The solutions to the stochastic differential equations are given by

$$S_t^j = S_0^j \exp\left(\left(r - \frac{1}{2}\sigma_j^2\right)t + \sigma_j B_t^j\right). \tag{5.4}$$

TABLE 5.4

The same as Table 5.3 but for the number of time steps $d = 64$. For weighted lattice rules, the weights are $\gamma_j = a\tau^j$ with $a = 0.098$ and $\tau = 0.98$. The optimal multiplier b^* in the control variate is $b^* \approx 1.037$.

				Unweighted lattice rules			Weighted lattice rules		
		MC	Sobol	KBV	CBC	ARC	KBV	CBC	ARC
Std	Crude	2.49e-2	1.17e-2	3.59e-2	2.21e-1	1.00e-1	4.68e-3	5.21e-3	6.24e-3
		(1.00)	(4.5)	(0.48)	(0.01)	(0.06)	(28)	(23)	(16)
	AV	1.05e-2	1.17e-2	3.59e-2	2.21e-1	1.00e-1	4.07e-3	5.32e-3	5.99e-3
		(5.6)	(4.5)	(0.48)	(0.01)	(0.06)	(37)	(22)	(17)
	CV	7.36e-4	7.36e-4	2.30e-3	1.29e-2	5.47e-3	3.76e-4	4.22e-4	4.28e-4
		(1141)	(1140)	(117)	(3.7)	(21)	(4369)	(3466)	(3364)
	ACV	5.42e-4	5.64e-4	2.34e-3	1.07e-2	4.94e-3	2.93e-4	3.59e-4	3.50e-4
		(2106)	(1941)	(113)	(5.4)	(25)	(7197)	(4788)	(5028)
BB	Crude	2.50e-2	1.41e-3	5.50e-2	1.36e-2	1.24e-2	2.09e-3	1.30e-3	1.27e-3
		(0.98)	(310)	(0.20)	(3.3)	(4.0)	(142)	(367)	(386)
	AV	1.06e-2	9.33e-4	5.44e-2	1.38e-2	1.28e-2	1.50e-3	9.53e-4	9.55e-4
		(5.5)	(709)	(0.21)	(3.2)	(3.7)	(275)	(679)	(677)
	CV	7.43e-4	2.46e-4	3.14e-3	1.75e-3	1.95e-3	3.00e-4	1.46e-4	1.68e-4
		(1117)	(10243)	(63)	(200)	(163)	(6847)	(29036)	(21780)
	ACV	5.41e-4	2.13e-4	3.10e-3	1.55e-3	1.81e-3	1.70e-4	9.79e-5	1.28e-4
		(2109)	(13638)	(64)	(256)	(189)	(21348)	(64470)	(37809)
PCA	Crude	2.51e-2	9.72e-4	1.40e-2	1.58e-3	1.22e-3	9.06e-4	7.93e-4	7.76e-4
		(0.98)	(653)	(3.1)	(249)	(417)	(752)	(983)	(1025)
	AV	1.06e-2	1.23e-4	1.34e-2	1.06e-3	9.69e-4	2.20e-4	1.30e-4	1.28e-4
		(5.5)	(40963)	(3.4)	(553)	(658)	(12737)	(36827)	(37602)
	CV	7.42e-4	6.96e-5	4.75e-3	3.34e-4	3.53e-4	1.33e-4	1.22e-4	1.19e-4
		(1120)	(127460)	(27)	(5524)	(4943)	(34986)	(41741)	(43480)
	ACV	5.41e-4	3.28e-5	2.93e-3	2.11e-4	2.20e-4	1.27e-4	1.05e-4	1.02e-4
		(2109)	(574710)	(72)	(13887)	(12743)	(38576)	(55668)	(59735)

Note that the random vector $(B_T^1, \ldots, B_T^d)^T =: (y_1, \ldots, y_d)^T$ is normally distributed with mean zero and covariance matrix $\Sigma = (\rho_{ij}T)_{i,j=1}^d$. Put $\nu_j := \log S_0^j + (r - \frac{1}{2}\sigma_j^2)T$. From (5.4), we have $S_T^j = e^{\nu_j + \sigma_j y_j}$. The payoff $\phi(S_T^1, \ldots, S_T^d)$ can be written as

$$\phi(S_T^1, \ldots, S_T^d) = \phi(e^{\nu_1+\sigma_1 y_1}, \ldots, e^{\nu_d+\sigma_d y_d}) =: G(\mathbf{y}).$$

Therefore, the current price of the derivative security can be written as

$$E_Q[e^{-rT}\phi(S_T^1, \ldots, S_T^d)] = \frac{e^{-rT}}{(2\pi)^{d/2}} \int_{R^d} G(A\mathbf{z}) e^{-\frac{1}{2}\mathbf{z}^T\mathbf{z}} d\mathbf{z},$$

where A is any real $d \times d$ matrix satisfying $AA^T = \Sigma$. The matrix A characterizes the method of generating the Brownian motions.

Consider a call option on the arithmetic average prices of the d assets. The payoff is

$$\phi_A(S_T^1, \ldots, S_T^d) = \max\left(0, \frac{1}{d}\sum_{j=1}^d S_T^j - K\right).$$

TABLE 5.5

Comparison of the standard deviations and relative efficiency ratios with respect to crude MC (in parentheses) for the arithmetic average multiasset option with 30 *replications for* $d = 16$ *(*$n = 4001$ *for MC and lattice rules and* $n = 4096$ *for the Sobol sequence). For weighted lattice rules, the weights are* $\gamma_j = a\tau^j$ *with* $a = 0.25$ *and* $\tau = 0.75$*. The optimal multiplier* b^* *in the control variate is* $b^* \approx 1.040$*.*

				Unweighted lattice rules			Weighted lattice rules		
		MC	Sobol	KBV	CBC	ARC	KBV	CBC	ARC
Std	Crude	2.91e-2	9.48e-3	4.47e-2	2.84e-2	2.59e-2	3.61e-3	3.13e-3	2.41e-3
		(1.00)	(9)	(0.42)	(1.1)	(1.3)	(65)	(86)	(146)
	AV	8.68e-3	8.25e-3	4.48e-2	2.55e-2	2.52e-2	2.58e-3	2.10e-3	2.30e-3
		(11)	(12)	(0.42)	(1.3)	(1.3)	(127)	(191)	(161)
	CV	1.75e-3	1.87e-3	6.74e-3	4.15e-3	4.63e-3	1.14e-3	7.73e-4	1.17e-3
		(277)	(243)	(19)	(49)	(40)	(648)	(1418)	(619)
	ACV	1.55e-3	1.24e-3	6.32e-3	3.77e-3	3.95e-3	9.50e-4	5.34e-4	7.90e-4
		(351)	(556)	(21)	(60)	(54)	(939)	(2965)	(1359)
PCA	Crude	2.93e-2	1.12e-3	1.91e-3	6.55e-3	5.18e-3	6.97e-4	6.95e-4	6.54e-4
		(0.99)	(679)	(234)	(20)	(32)	(1744)	(1754)	(1982)
	AV	8.79e-3	1.22e-4	1.17e-3	1.77e-3	2.48e-3	9.14e-5	7.43e-5	7.61e-5
		(11)	(57212)	(615)	(270)	(138)	(101420)	(153570)	(146360)
	CV	1.75e-3	2.96e-4	1.63e-3	6.85e-3	4.93e-3	8.35e-5	1.47e-4	1.57e-4
		(277)	(9665)	(320)	(18)	(35)	(121490)	(39342)	(34515)
	ACV	1.57e-3	9.81e-5	1.17e-3	1.78e-3	2.49e-4	5.76e-5	5.37e-5	6.59e-5
		(346)	(88110)	(615)	(267)	(136)	(255420)	(293410)	(195280)

The control variate is chosen to be the payoff of the corresponding call option on the geometric average price of the d assets: $\phi_G(S_T^1, \ldots, S_T^d) = \max(0, (\prod_{j=1}^d S_T^j)^{1/d} - K)$. An analytical expression can be easily derived for $E_Q[\phi_G(S_T^1, \ldots, S_T^d)]$. The following parameters will be used:

$$S_0^j = 100, \quad \sigma_j = 0.2, \quad \rho_{ij} = 0.3, \quad r = 0.1, \quad T = 1 \text{ year}, \quad K = 100.$$

We use the same strategy as above. We find suitable weights for the standard construction and the geometric average multiasset option and use them to construct lattice rules for the arithmetic average multiasset option. (For the model parameters given above, the parameters which determine the weights in the form (3.7) are $a = 0.25$ and $\tau = 0.75$ for both $d = 16$ and $d = 64$.) Variance reduction and dimension reduction techniques will also be used. The computational results are given in Tables 5.5 and 5.6.

5.3. Summary of the computational experiments. The computational results in sections 5.1 and 5.2 show the following:

- The weights γ_j have a strong influence on the efficiency for all types of lattice rules, not only for CBC lattice rules but also for Korobov lattice rules. Without weights (i.e., all weights are taken to be 1), all three types of lattice rules behave very badly: their performance is not only much worse than that of the Sobol sequence but also worse than that of MC. However, with the suitably chosen weights, the efficiency of all types of lattice rules is dramatically improved. In many cases (especially in the standard construction of Brownian motion), they are more efficient than the Sobol sequence.

TABLE 5.6
The same as Table 5.5 but with $d = 64$. For weighted lattice rules, the weights are $\gamma_j = a\tau^j$ with $a = 0.25$ and $\tau = 0.75$. The optimal multiplier b^* in the control variate is $b^* \approx 1.042$.

				Unweighted lattice rules			Weighted lattice rules		
		MC	Sobol	KBV	CBC	ARC	KBV	CBC	ARC
Std	Crude	2.80e-2	8.60e-3	4.65e-2	6.76e-2	4.23e-2	3.77e-3	3.64e-3	3.75e-3
		(1.00)	(11)	(0.36)	(0.17)	(0.44)	(55)	(59)	(56)
	AV	7.94e-3	8.67e-3	4.77e-2	6.74e-2	4.09e-2	2.61e-3	3.07e-3	3.29e-3
		(12)	(10)	(0.34)	(0.17)	(0.47)	(115)	(83)	(72)
	CV	1.39e-3	1.26e-3	5.59e-3	8.04e-3	5.44e-3	7.42e-4	6.83e-4	8.13e-4
		(403)	(494)	(25)	(12)	(27)	(1425)	(1682)	(1186)
	ACV	1.22e-3	1.07e-3	5.16e-3	8.13e-3	4.95e-3	6.60e-4	6.10e-4	6.44e-4
		(527)	(689)	(29)	(12)	(32)	(1797)	(2108)	(1946)
PCA	Crude	2.81e-2	9.75e-4	1.12e-3	4.27e-3	4.15e-3	7.55e-4	8.43e-4	8.16e-4
		(0.99)	(824)	(622)	(43)	(46)	(1374)	(1103)	(1174)
	AV	7.98e-3	7.84e-5	2.30e-4	1.51e-3	1.74e-3	6.85e-5	6.83e-5	7.43e-5
		(12)	(127590)	(14789)	(345)	(259)	(166960)	(167780)	(141880)
	CV	1.39e-3	1.24e-4	7.16e-4	4.65e-3	4.10e-3	1.43e-4	1.47e-4	1.46e-4
		(404)	(51329)	(1528)	(36)	(47)	(38521)	(36279)	(36318)
	ACV	1.23e-3	4.83e-5	2.33e-4	1.50e-3	1.74e-3	4.06e-5	3.79e-5	4.65e-5
		(518)	(335560)	(14415)	(347)	(260)	(475130)	(544500)	(362230)

- The lattice rule constructed by the acceptance-rejection control algorithm is as efficient as the full search CBC lattice rule. But the former can be constructed faster, and thus is more suitable for practical use. In the weighted case, both types of CBC lattice rules are in general more efficient than Korobov lattice rules.
- Dimension reduction and variance reduction techniques can increase the efficiency of lattice rules considerably (there is no advantage in using antithetic variates in QMC without BB or PCA, but there is some advantage for QMC in conjunction with BB or PCA). Their combinations lead to extremely efficient estimates. However, the unweighted lattice rules in conjunction with these techniques lead to results that are worse than MC. This means that even the control variates or dimension reduction techniques cannot rescue the unweighted lattice rules. In other words, the potential efficiency of dimension reduction and variance reduction techniques also depends on the quality of the point set.
- For the standard construction the full CBC lattice rules seem to perform better than the Sobol method. For the BB and PCA constructions the contest is more even. It should be noted that our matching strategy was carried out for the standard construction only, so a reduction in quality when applied to the BB and PCA constructions is not unexpected. (Therefore there is still room to improve the efficiency of lattice rules in the context of the BB or PCA constructions.)
- The weights $\gamma_j = a\tau^j$ found for our examples are quite different from the classical ones ($\gamma_j = 1$ for all j). The matching strategy of choosing weights for these problems is both practical and well suited to the finance problems considered here. Even a rough estimation of weights can offer a large improvement in efficiency.

5. To see how "robust" the parameters a and τ are, we selected several sets of model parameters (σ, K, r) for the case of the Asian option and determined a and τ by the method discussed in sections 4 and 5.1. Surprisingly, given the dimension d, the value of τ remained almost constant, while the parameter a changed slightly with a very clear trend: it increases as the strike price K and/or the volatility σ increases (and/or as the interest rate r decreases). The same phenomenon was observed for a multiasset option for fixed correlations ρ_{ij}.

The values of a have a common feature in all our examples: they are much smaller than 1. Such values of a capture the essential characteristics of finance problems. Small values of a guarantee that more attention is put on low-order ANOVA terms and less attention on high-order terms. The resulting lattice rules are guaranteed to have good low-dimensional projections, especially two-dimensional projections. Therefore, the lattice rules are robust in some sense. This makes it sensible to use the weights and the lattice rules found for one set of model parameters for other sets of model parameters without significant changes. In our experience, small changes of weights have no significant influence on the quality of the lattice rules. We omit the computational results relating to these aspects.

In general, lattice rules constructed with small a and small τ are well suited to functions with small superposition dimension (d_{su}) and small truncation dimension (d_{tr}), while lattice rules constructed with small a but with $\tau \approx 1$ are well suited to functions with small d_{su} but not necessarily small d_{tr}. Since finance problems often have small d_{su} (or even small d_{tr}), any constructions (based on any criterion, not just the one we used in this paper), which generate lattice rules with good low-dimensional projections (especially two-dimensional projections), may work well for such problems; see [16, 31].

6. Concluding remarks. This paper investigated lattice rules for certain high-dimensional problems in finance, concentrating on the problem of what weights should be used in practice for constructing lattice rules. We explained why the classical lattice rules may be inefficient for high-dimensional finance problems. Consistently with this, the computational results show that blind use of lattice rules may lead to estimates that are worse than the MC estimate. Exploiting information about the problem, we choose suitable weights by a matching strategy, which relates the weights to the sensitivity indices. The lattice rules constructed with the suitably chosen weights improve the performance significantly over the usual unweighted lattice rules. The computational experiments are quite encouraging. More complicated finance problems need to be investigated empirically, and more research is needed into the construction of robust efficient lattice rules.

Acknowledgment. The authors would like to thank the referees for their valuable comments.

REFERENCES

[1] P. Acworth, M. Broadie, and P. Glasserman, *A comparison of some Monte Carlo and quasi-Monte Carlo techniques for option pricing*, in Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing, P. Hellekallek and H. Niederreiter, eds., Springer-Verlag, New York, 1998, pp. 1–18.

[2] P. Boyle, M. Broadie, and P. Glasserman, *Monte Carlo methods for security pricing*, J. Econom. Dynam. Control, 21 (1997), pp. 1267–1321.

[3] R. E. Caflisch, W. Morokoff, and A. Owen, *Valuation of mortgage backed securities using Brownian bridges to reduce effective dimension*, J. Comp. Finance, 1 (1997), pp. 27–46.

[4] R. Cranley and T. N. L. Patterson, *Randomization of number theoretic methods for multiple integration*, SIAM J. Numer. Anal., 13 (1976), pp. 904–914.
[5] J. Dick, I. H. Sloan, X. Wang, and H. Woźniakowski, *Liberating the weights*, J. Complexity, 20 (2004), pp. 593–623.
[6] J. Dick, I. H. Sloan, X. Wang, and H. Woźniakowski, *Good lattice rules in weighted Korobov spaces with general weights*, Numer. Math., 103 (2006), pp. 63–97.
[7] F. J. Hickernell, *Quadrature error bounds with applications to lattice rules*, SIAM J. Numer. Anal., 33 (1996), pp. 1995–2016.
[8] F. J. Hickernell, *Erratum: Quadrature error bounds with applications to lattice rules*, SIAM J. Numer. Anal., 34 (1997), pp. 853–866.
[9] F. J. Hickernell, *A generalized discrepancy and quadrature error bound*, Math. Comp., 67 (1998), pp. 299–322.
[10] F. J. Hickernell, C. Lemieux, and A. Owen, *Control variates for quasi-Monte Carlo*, Statist. Sci., 20 (2005), pp. 1–31.
[11] F. J. Hickernell and X. Wang, *The error bounds and tractability of quasi-Monte Carlo algorithms in infinite dimension*, Math. Comp., 71 (2002), pp. 1641–1661.
[12] F. J. Hickernell and H. Woźniakowski, *Integration and approximation in arbitrary dimension*, Adv. Comput. Math., 12 (2000), pp. 25–58.
[13] J. C. Hull, *Options, Futures, and Other Derivatives*, 5th ed., Prentice–Hall, Upper Saddle River, NJ, 2003.
[14] N. M. Korobov, *Approximate evaluation of repeated integrals*, Dokl. Akad. Nauk SSSR, 124 (1959), pp. 1207–1210 (in Russian).
[15] F. Y. Kuo, *Component-by-component constructions achieve the optimal rate of convergence for multivariate integration in weighted Korobov and Sobolev spaces*, J. Complexity, 19 (2003), pp. 301–320.
[16] P. L'Ecuyer and C. Lemieux, *Variance reduction via lattice rules*, Management Sci., 46 (2000), pp. 1214–1235.
[17] B. Moskowitz and R. E. Caflisch, *Smoothness and dimension reduction in quasi-Monte Carlo methods*, Math. Comput. Modelling, 23 (1996), pp. 37–54.
[18] H. Niederreiter, *Random Number Generation and Quasi-Monte Carlo Methods*, SIAM, Philadelphia, 1992.
[19] D. Nuyens and R. Cools, *Fast algorithms for component-by-component construction of rank-1 lattice rules in shift invariant reproducing kernel Hilbert spaces*, Math. Comp., 75 (2006), pp. 903–920.
[20] A. B. Owen, *Randomly permuted* (t, m, s)*-nets and* (t, s)*-sequences*, in Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing, H. Niederreiter and P. J.-S. Shiue, eds., Springer-Verlag, New York, 1995, pp. 299–317.
[21] S. H. Paskov and J. F. Traub, *Faster valuation of financial derivatives*, J. Portfolio Management, 22 (1995), pp. 113–120.
[22] I. H. Sloan and S. Joe, *Lattice Methods for Multiple Integration*, Oxford University Press, Oxford, UK, 1994.
[23] I. H. Sloan, F. Y. Kuo, and S. Joe, *Constructing randomly shifted lattice rules in weighted Sobolev spaces*, SIAM J. Numer. Anal., 40 (2002), pp. 1650–1665.
[24] I. H. Sloan and A. V. Reztsov, *Component-by-component construction of good lattice points*, Math. Comp., 71 (2002), pp. 263–273.
[25] I. H. Sloan and H. Woźniakowski, *When are quasi-Monte Carlo algorithms efficient for high dimensional integrals?*, J. Complexity, 14 (1998), pp. 1–33.
[26] I. H. Sloan and H. Woźniakowski, *Tractability of multivariate integration for weighted Korobov classes*, J. Complexity, 17 (2001), pp. 697–721.
[27] I. H. Sloan and H. Woźniakowski, *Tractability of integration in non-periodic and periodic weighted tensor product Hilbert spaces*, J. Complexity, 18 (2002), pp. 479–499.
[28] I. M. Sobol', *Distribution of points in a cube and approximate evaluation of integrals*, Zh. Vychisl. Mat. Mat. Fiz., 7 (1967), pp. 784–802.
[29] I. M. Sobol', *Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates*, Math. Comput. Simulation, 55 (2001), pp. 271–280.
[30] X. Wang, *Strong tractability of multivariate integration using quasi-Monte Carlo algorithms*, Math. Comp., 72 (2003), pp. 823–838.
[31] X. Wang, *Constructing Robust Good Lattice Rules for Computational Finance*, manuscript, 2005.
[32] X. Wang and K. T. Fang, *Effective dimensions and quasi-Monte Carlo integration*, J. Complexity, 19 (2003), pp. 101–124.

[33] X. Wang and F. J. Hickernell, *Randomized Halton sequences*, Math. Comput. Modelling, 32 (2000), pp. 887–899.
[34] X. Wang and I. H. Sloan, *Why are high-dimensional finance problems often of low effective dimension?*, SIAM J. Sci. Comput., 27 (2005), pp. 159–183.
[35] X. Wang, I. H. Sloan, and J. Dick, *On Korobov lattice rules in weighted spaces*, SIAM J. Numer. Anal., 42 (2004), pp. 1760–1779.

SIAM J. SCI. COMPUT.
Vol. 28, No. 2, pp. 751–775
© 2006 Society for Industrial and Applied Mathematics

COMPUTATIONAL MODELING OF UNCERTAINTY IN TIME-DOMAIN ELECTROMAGNETICS*

C. CHAUVIÈRE†, J. S. HESTHAVEN‡, AND L. LURATI‡

Abstract. We discuss computationally efficient ways of accounting for the impact of uncertainty, e.g., lack of detailed knowledge about sources, materials, shapes, etc., in computational time-domain electromagnetics. In contrast to classic statistical Monte Carlo–based methods, we explore a probabilistic approach based on high-order accurate expansions of general stochastic processes. We show this to be highly efficient and accurate on both one- and two-dimensional examples, enabling the computation of global sensitivities of measures of interest, e.g., radar-cross-sections (RCS) in scattering applications, for a variety of types of uncertainties.

Key words. Maxwell's equations, discontinuous Galerkin methods, uncertainty quantification, chaos expansion

AMS subject classifications. 65N35, 65C05, 65C20, 65C30

DOI. 10.1137/040621673

1. Introduction.

While computational methods have become increasingly refined and accurate, their reliance on exact data, e.g., complete descriptions of geometries, materials, sources, etc., is emerging as a bottleneck in the modeling of problems of realistic complexity. For instance, if one attempts to model an experiment, a classic computational approach would require knowledge of a degree of detail which is unrealistic and often impossible to obtain, e.g., one cannot hope to control all elements of an experiment, measure all details of an initial condition or geometry, know the microstructure of all materials, etc.

The usual approach to dealing with this lack of knowledge or uncertainty is to simply assume some mean parameters and compute the corresponding solution. If the solution is robust to parameter variation, this is indeed a reasonable approach. However, for general problems, where the sensitivity of parts of the solution can be significant, a solution based on mean parameters is not likely to match very well with experiments and, thus, is not a good predictive tool. An experimentalist would naturally deal with this exact problem by repeating the experiments and, subsequently, compute not only mean results but also error bars—reflecting, at least partly, the sensitivity of the solution.

It is natural to strive to achieve similar abilities in computational modeling effort, e.g., compute solutions or measures of interest and their associated sensitivities. Clearly, if a particular measured entity is highly sensitive there is no reason to expect that this matches experiments exactly. Thus, we would like to be able to model the impact of the uncertainty, assumed to have certain properties derived from experiments or otherwise, on the computed results, essentially resulting in an ensample of

*Received by the editors December 28, 2004; accepted for publication (in revised form) November 30, 2005; published electronically May 26, 2006.
http://www.siam.org/journals/sisc/28-2/62167.html

†Laboratoire de Mathématiques, Université Blaise Pascal, 63177 Aubière Cedex, France (Cedric.Chauviere@math.univ-bpclermont.fr).

‡Division of Applied Mathematics, Brown University, Providence, RI 02912 (Jan.Hesthaven@brown.edu, laural@dam.brown.edu). The work of the second author was partly supported by NSF Career Award DMS0132967, AFOSR contract FA9550-04-1-0072, and the Alfred P. Sloan Foundation through a Sloan Research Fellowship. The work of the third author was supported by AFOSR contract FA9550-04-1-0072.

possible solution values with an associated probability. This would immediately enable the computation of statistical moments, e.g., means and variances, and of other valuable information about the sensitivity of solutions and derived quantities.

In this paper we pursue this goal and present a systematic, accurate, and efficient way of addressing this type of problem, essentially enabling one to compute with an ensample of data and, subsequently, obtain a full space-time ensample of solutions with an associated probability density. It is important to realize that is this not a simple situation, since solutions may vary nonlinearly in the uncertainty due to stochastic correlations even if the deterministic problem is linear, e.g., Maxwell's equations.

A standard way of addressing problems of the type mentioned in the above is through Monte Carlo sampling [13], e.g., run a deterministic code a large number of times and, subsequently, extract the statistics of interest. The main problem with this approach is the very slow convergence rate, $\mathcal{O}(N^{-1/2})$, with N being the number of samples, which makes even the computation of mean solutions expensive and accurate recovery of higher moments, e.g., variances, prohibitive. As we shall see through examples, the techniques proposed here suggests that a very considerable reduction is possible without impacting the accuracy.

Our platform on which to demonstrate this approach is the time-domain Maxwell's equations, solved using a high-order accurate discontinuous Galerkin method [9]. However, the basic elements of the formulation are general and can be used with any computational kernel.

What remains of the paper is organized as follows. In the next two sections, we recall the deterministic Maxwell's equations in the time domain and we give some details of its space discretization using a high-order discontinuous Galerkin method. In section 4 we continue with an introduction to homogeneous chaos and stochastic collocation methods, enabling the transition from deterministic to stochastic modeling. In the same section, we explain how to model uncertainties and how to extract statistics of interest (i.e., mean and variance) from the computed stochastic solution. This sets the stage for numerous examples presented in section 5, comparing the two approaches and validating the general approach. In section 6 we conclude and offer some suggestions for continued research in this direction.

2. Maxwell's equations. Let us consider a general domain Ω and let $\mathbf{E}^s$ and $\mathbf{H}^s$ denote the scattered electric and magnetic fields, respectively. With $\varepsilon(\mathbf{x})$ and $\mu(\mathbf{x})$ being the local permittivity and permeability, and $\sigma(\mathbf{x})$ the conductivity of the media, the time-dependent Maxwell's equations in the scattered field formulation are given as

$$\epsilon \frac{\partial \mathbf{E}^s}{\partial t} = \nabla \times \mathbf{H}^s + \sigma \mathbf{E}^s + \mathbf{S}^E, \tag{2.1}$$

$$\mu \frac{\partial \mathbf{H}^s}{\partial t} = -\nabla \times \mathbf{E}^s + \mathbf{S}^H, \tag{2.2}$$

where, as is common in time-domain schemes, we have neglected divergence constraints, assuming that these amount to constraints on the initial conditions.

The source terms, $\mathbf{S}^E$ and $\mathbf{S}^H$, appearing on the right-hand side of (2.1)–(2.2), take the form

$$\mathbf{S}^E = -(\epsilon - \epsilon^i)\frac{\partial \mathbf{E}^i}{\partial t} + (\sigma - \sigma^i)\mathbf{E}^i, \tag{2.3}$$

$$\mathbf{S}^H = -(\mu - \mu^i)\frac{\partial \mathbf{H}^i}{\partial t}, \tag{2.4}$$

where the incident field $(\mathbf{E}^i, \mathbf{H}^i)$ is a solution to Maxwell's equations in a media of permittivity, permeability, and conductivity—$\varepsilon^i(\mathbf{x})$, $\mu^i(\mathbf{x})$, $\sigma^i(\mathbf{x})$, respectively.

We now turn our attention to boundary conditions. Along a perfect electric conductor (PEC), the boundary conditions on the electric field are

$$\hat{\mathbf{n}} \times \mathbf{E}^t = \mathbf{0}, \tag{2.5}$$

where $\hat{\mathbf{n}}$ is the outward pointing normal vector at the surface and $\mathbf{E}^t = \mathbf{E}^i + \mathbf{E}^s$ is the total field. For the total magnetic field $\mathbf{H}^t$, the condition is

$$\mathbf{H}^t \cdot \hat{\mathbf{n}} = 0. \tag{2.6}$$

Along the interface of two dielectric bodies, denoted by the subscripts 1 and 2, we have

$$\hat{\mathbf{n}} \times (\mathbf{E}_1^s - \mathbf{E}_2^s) = \mathbf{0} \text{ and } \hat{\mathbf{n}} \times (\mathbf{H}_1^s - \mathbf{H}_2^s) = \mathbf{0}; \tag{2.7}$$

i.e., all tangential components are continuous.

3. Numerical scheme for the deterministic case. Before turning our attention to the case including uncertainties, let us briefly describe the computational methods used for solving Maxwell's equations in the physical space. We use a discontinuous Galerkin formulation; the solution will be discontinuous between elements. This offers a number of advantages over widely used alternative techniques, e.g., geometric flexibility through fully unstructured grids, high-order accuracy to enable accurate wave propagation over long distance with a coarse resolution, and very high computational efficiency through explicit time stepping and high parallel performance. The method has been discussed, analyzed, and validated extensively [9, 10, 11, 12], and we shall simply sketch its main components.

We rewrite Maxwell's equations (2.1)–(2.2) in conservation form

$$Q\frac{\partial \mathbf{q}}{\partial t} + \nabla \cdot \mathbf{F}(\mathbf{q}) = \mathbf{S}, \tag{3.1}$$

where $\mathbf{q}$ is the state vector formed by the scattered electric field and the magnetic field

$$\mathbf{q} = \begin{pmatrix} \mathbf{E} \\ \mathbf{H} \end{pmatrix}, \tag{3.2}$$

and the components of the tensor $\mathbf{F}$ are defined by

$$\mathbf{F}_i(\mathbf{q}) = \begin{pmatrix} -\mathbf{e}_i \times \mathbf{H} \\ \mathbf{e}_i \times \mathbf{E} \end{pmatrix}, \tag{3.3}$$

where $\mathbf{e}_i$ denotes the Cartesian unit vectors. On the right-hand side of (3.1), $\mathbf{S} = [\mathbf{S}^E, \mathbf{S}^H]$ is the source term, which depends on the incident field, and the material matrix Q is a diagonal matrix with values $(\epsilon, \epsilon, \epsilon, \mu, \mu, \mu)$ on its diagonal.

We assume that the computational domain, Ω, is tessellated by triangles or tetrahedrons, D, and we represent the local solution $\mathbf{q}_N$ as

$$\mathbf{q}_N(\mathbf{x}, t) = \sum_{i=1}^{N} \widetilde{\mathbf{q}}_i(t) L_i(\mathbf{x}), \tag{3.4}$$

where $\widetilde{\mathbf{q}}_i$ reflects nodal values, defined on the element, and $L_i(\mathbf{x})$ signifies an nth order Lagrange polynomial, associated with grid points; see [9, 10, 11, 12] for details.

The discrete solution, $\mathbf{q}_N$, of Maxwell's equations is required to satisfy

$$\int_D \left(Q\frac{\partial \mathbf{q}_N}{\partial t} + \nabla \cdot \mathbf{F}(\mathbf{q}_N) - \mathbf{S}_N \right) L_i(\mathbf{x}) d\mathbf{x} = \oint_{\partial D} \hat{\mathbf{n}} \cdot [\mathbf{F}(\mathbf{q}_N) - \mathbf{F}^*] L_i(\mathbf{x}) d\mathbf{x}. \tag{3.5}$$

In (3.5), $\mathbf{F}^*$ denotes a numerical flux and $\hat{\mathbf{n}}$ is an outward pointing unit vector defined at the boundary ∂D of the element D. Note that this is an entirely local formulation, and relaxing the continuity of the elements decouples the elements, resulting in a block-diagonal global mass matrix which can be trivially inverted in preprocessing. The price to pay for this is the additional degrees of freedom needed to support the local basis functions. For high-order elements, this is, however, only a small fraction of the total number of degrees of freedom. Given the linearity of Maxwell's equations, and for stability reasons, it is natural to use an upwinding flux which takes the form [9]

$$\hat{\mathbf{n}} \cdot [\mathbf{F}(\mathbf{q}_N) - \mathbf{F}^*] = \begin{pmatrix} \overline{Z}^{-1} \hat{\mathbf{n}} \times (\hat{\mathbf{n}} \times [\mathbf{E}_N] - Z^+ [\mathbf{H}_N]) \\ \overline{Y}^{-1} \hat{\mathbf{n}} \times (\hat{\mathbf{n}} \times [\mathbf{E}_N] - Z^+ [\mathbf{H}_N]) \end{pmatrix}, \tag{3.6}$$

where the bracket $[\mathbf{q}] = \mathbf{q}^- - \mathbf{q}^+$ denotes the jump in the field values ($\mathbf{q}^-$ is the local value and $\mathbf{q}^+$ is the neighboring value) across an interface, $Z^\pm$ is the local impedance, and $Y^\pm$ is the local conductance,

$$Z^\pm = \frac{1}{Y^\pm} = \sqrt{\frac{\mu^\pm}{\epsilon^\pm}}. \tag{3.7}$$

$\overline{Z}$ and $\overline{Y}$ in (3.6) are the sums

$$\overline{Z} = Z^+ + Z^-, \quad \overline{Y} = Y^+ + Y^-. \tag{3.8}$$

After discretization of the operators and evaluation of the integrals appearing in (3.5), the problem can be rewritten in matrix-vector form (see [10] for details)

$$QM\frac{d\mathbf{q}_N}{dt} + S \cdot \mathbf{F}_N - M\mathbf{S}_N = F\hat{\mathbf{n}} \cdot [\mathbf{F}_N - \mathbf{F}^*]. \tag{3.9}$$

The matrices M, S, and F represent the local mass-, stiffness-, and face-integration matrices, respectively, the exact entries of which can be found in [9], where it is also discussed how to compute these local operators efficiently and accurately. Note that the local nature of the scheme allows for the use of an explicit solver for the time discretization of (3.9). This is traditionally done using an explicit Runge–Kutta method, although suitable alternatives are plentiful.

4. Accounting for the uncertainty. To deal with the actual lack of detailed knowledge leading to the uncertainty, we must make some educated guesses—a model—about the nature of the possible variations in the data. These models can be based on pure speculation, on measured data, or on other available information. Once this is done, however, we must introduce this into the computational approach in an efficient, accurate, and robust manner.

Consider, as a simple example, the wave equation

$$\frac{\partial u}{\partial t} + a(\theta)\frac{\partial u}{\partial x} = 0, \tag{4.1}$$

where θ is a random parameter with some associated probability density function (PDF). Thus, this represents an ensemble of wave problems, each with an individual phase speed occurring with a probability given by the PDF; i.e., the solution u is not only a function of (x,t) but also of θ, and the equation is a stochastic wave equation.

The actual form of $a(\theta)$ may not be known, causing the introduction of the uncertainty. However, it seems quite reasonable that in many situations one can measure or conjecture the average of a and perhaps its variation; i.e., one can reasonably assume that $a(\theta)$ varies in a certain way related to a normal distribution with a given mean and variance.

The question remains, however, how this uncertainty will affect the solution, $u(x,t,\theta)$, and, more often, its moments such as the mean and the variance. This is not a trivial question, as the uncertainty essentially renders the simple linear problem considerably more complex due to possible stochastic dependence between a and u.

One could naturally solve the above problem for a large number of values of θ, taken from a proper distribution, and, subsequently, compute the appropriate moments. This is the essence of a Monte Carlo method and suffers from a very slowly converging result as $1/\sqrt{N}$, with N being the number of samples. If higher-order moments, e.g., the variance or sensitivity of the result, are required, this is often prohibitive.

In the above example, the uncertainty enters through the phase speed. However, as we shall see shortly, dealing with uncertainty in initial conditions, boundary conditions, sources, or computational domain/geometries can be done in an entirely similar fashion.

In this section, we shall discuss two variations, referred to as stochastic Galerkin and collocation, respectively, of the same underlying result. This enables one to discretize stochastic partial differential equations (SPDEs) to recover systems of deterministic problems which we can subsequently solve with the methods discussed in section 3. As we shall experience through examples, both techniques offer excellent accuracy at a severely reduced computational cost as compared to straightforward Monte Carlo methods.

4.1. The homogeneous chaos expansion. The key result on which we shall rely is due to Wiener [16] (see also Cameron and Martin [3], Schoutens [14], and Ghanem and Spanos [7]) and shows that the chaos expansion can be used to approximate any functional in $L^2(\Omega,\mathcal{P})$, where $\mathcal{P}$ is a Gaussian measure on Ω. For such a random variable $X(\theta)$, the chaos expansion is written as

$$
\begin{aligned}
X(\theta) = a_0 H_0 &\\
&+ \sum_{i_1=1}^{d} a_{i_1} H_1(\xi_{i_1}(\theta)) \\
&+ \sum_{i_1=1}^{d} \sum_{i_2=1}^{i_1} a_{i_1 i_2} H_2(\xi_{i_1}(\theta), \xi_{i_2}(\theta)) \\
&+ \sum_{i_1=1}^{d} \sum_{i_2=1}^{i_1} \sum_{i_3=1}^{i_2} a_{i_1 i_2 i_3} H_3(\xi_{i_1}(\theta), \xi_{i_2}(\theta), \xi_{i_3}(\theta)) \\
&+ \cdots,
\end{aligned}
\tag{4.2}
$$

where $\xi = (\xi_1(\theta), \ldots, \xi_d(\theta))$ represents d independent Gaussian variables with zero mean and unit variance, each depending on the random event θ, and H_n are the

multivariate Hermite polynomials defined as

$$H_n(\xi_{i_1}(\theta),\dots,\xi_{i_n}(\theta)) = e^{\frac{1}{2}\xi^T\xi}(-1)^n \frac{\partial^n}{\partial\xi_{i_1},\dots,\partial\xi_{i_n}} e^{-\frac{1}{2}\xi^T\xi}. \tag{4.3}$$

The number of terms in the expansion (4.2) grows as

$$P = \frac{(n+d)!}{n!d!}, \tag{4.4}$$

where n is the length of the Hermite expansion and d is the dimension of the Gaussian random space. The polynomial homogeneous chaos expansion forms a complete orthogonal basis in the space of Gaussian variables, i.e.,

$$\begin{aligned}\langle H_i(\xi_{i_1}(\theta),\dots,\xi_{i_n}(\theta)), H_j(\xi_{j_1}(\theta),\dots,\xi_{j_n}(\theta))\rangle \\ = \int_{R^d} \frac{H_i(\xi_{i_1}(\theta),\dots,\xi_{i_n}(\theta))H_j(\xi_{j_1}(\theta),\dots,\xi_{j_n}(\theta))}{\sqrt{(2\pi)^d}} e^{-\frac{1}{2}\xi^T\xi} d\xi = i!\delta_{ij},\end{aligned} \tag{4.5}$$

where i denotes the multi-index $(i_1,\dots,i_n)$ and $i! = i_1!,\dots,i_n!$. Thus there is an intimate relation between the Hermite polynomials, orthogonal under the Gaussian weight, and the representation of random variables taken from a Gaussian distribution. One way of interpreting the homogeneous chaos expansion is that a general random variable can be expressed in terms of simpler Gaussian variables for which we can construct an efficient computational approach. Clearly, if the random variable is far from Gaussian, many terms in the expansion will be needed; i.e., n must be large. An alternative is to use an expansion in terms of other random variables with an associated distribution closer to what is expected at input or output (see [14] for generalizations). For notational convenience, we can rewrite (4.2) in the form

$$X(\theta) = \sum_{j=1}^{P} b_j \Psi_j(\xi), \tag{4.6}$$

where there is a one-to-one correspondence between the functions H_i and Ψ_j and also between the coefficients $a_{i_1,\dots,i_p}$ and b_j. In the case of a general stochastic process, these coefficients will be time dependent.

To model the impact of uncertainty on the propagation of electromagnetic waves, we include the randomness in the usual spatial-temporal dimensions; i.e., the electric field and the magnetic field become $\mathbf{E}(\mathbf{x},t,\theta)$ and $\mathbf{H}(\mathbf{x},t,\theta)$, reflecting that the fields are functions of d independent random variables, $(\xi_{i_1}(\theta),\dots,\xi_{i_d}(\theta))$.

In the following we shall discuss in some detail how this can be utilized to construct an efficient computational method. For simplicity of the discussion, we assume in what follows that one Gaussian variable suffices to represent the process (i.e., $d = 1$). However, the formulation is general and applies to problems requiring many random variables to describe the stochastic processes.

4.2. Stochastic Galerkin formulation. Using the chaos expansion, we can express $\mathbf{q}(\mathbf{x},t,\theta) = (\mathbf{E}(\mathbf{x},t,\theta),\mathbf{H}(\mathbf{x},t,\theta))^T$ as

$$\mathbf{q}(\mathbf{x},t,\theta) = \sum_{i=1}^{P} \mathbf{q}^i(\mathbf{x},t)\Psi_i(\theta). \tag{4.7}$$

We can write the computational scheme, which takes into account randomness in a general setting, as

$$\left\{ \begin{array}{l} Q(\theta) M \dfrac{d\mathbf{q}_N}{dt} + S \cdot \mathbf{F}_N - M\mathbf{S}(\theta)_N = F\hat{\mathbf{n}} \cdot [\mathbf{F}_N - \mathbf{F}^*], \\ \mathbf{q}_N(\mathbf{x}, t = 0, \theta) = \mathbf{f}(\mathbf{x}, \theta), \end{array} \right. \tag{4.8}$$

where the initial conditions are given by the function $\mathbf{f} = \mathbf{f}(\mathbf{x}, \theta)$ and the unknown vector $\mathbf{q}_N$ is given by (4.7). As a first step, we discretize (4.8) in the random space using a Galerkin approach. Multiplying (4.8) by a test function $\Psi_k(\theta)$, replacing $\mathbf{q}_N$ by its chaos expansion, and using the scalar product defined by (4.5), we obtain

$$\forall k \in [1, P] \, : \, \sum_{i=1}^{P} \langle Q\Psi_i, \Psi_k \rangle \, M \frac{d\mathbf{q}_N^i}{dt} + k! S \cdot \mathbf{F}_N^k - M\mathbf{S}_N^k = F \sum_{i=1}^{P} \hat{\mathbf{n}} \cdot [\mathbf{F}_N^i - \mathbf{F}^{i*}], \tag{4.9}$$

where we have used property (4.5). The detailed expression of the terms on the right-hand side is

$$\hat{\mathbf{n}} \cdot [\mathbf{F}_N^i - \mathbf{F}^{i*}] = \left(\begin{array}{l} \left\langle \overline{Z}^{-1} \Psi_i, \Psi_k \right\rangle \hat{\mathbf{n}} \times \hat{\mathbf{n}} \times [\mathbf{E}^i] - \left\langle Z^+ \overline{Z}^{-1} \Psi_i, \Psi_k \right\rangle \hat{\mathbf{n}} \times [\mathbf{H}^i] \\ \left\langle \overline{Y}^{-1} \Psi_i, \Psi_k \right\rangle \hat{\mathbf{n}} \times \hat{\mathbf{n}} \times [\mathbf{H}^i] + \left\langle Y^+ \overline{Y}^{-1} \Psi_i, \Psi_k \right\rangle \hat{\mathbf{n}} \times [\mathbf{E}^i] \end{array} \right). \tag{4.10}$$

Recall that Z and Y may depend on the material properties and, thus, may include uncertainty/randomness.

The initial conditions in (4.8) also need to be projected onto the chaos basis to give an initial condition for each mode of $\mathbf{q}_N^i$ in the chaos expansion, i.e.,

$$\forall i \in [1, P] \, : \, \mathbf{q}_N^i(\mathbf{x}, t = 0) = \frac{1}{i!} \langle \mathbf{f}(\mathbf{x}, \theta), \Psi_i \rangle. \tag{4.11}$$

Once the vectors $\{\mathbf{q}_N^i\}_{1 \le i \le P}$ of the system (4.9) have been computed, we have available at every point in space an approximation to the probability density of the solution to the system.

Considering (4.9), we observe that we have managed to recast the general stochastic problem into a system of P coupled deterministic problems which we can now discretize in space/time exactly as discussed in section 3—or in any other preferred way.

4.3. Stochastic collocation formulation. The idea of the stochastic collocation formulation is to replace the expression of the electric and magnetic fields (4.7) in the polynomial chaos expansion with a Lagrangian polynomial basis; i.e., we would have

$$\mathbf{q}(\mathbf{x}, t, \theta) = \sum_{i=1}^{P} \mathbf{q}^i(\mathbf{x}, t) L_i(\theta), \tag{4.12}$$

where $\{L_i(\theta)\}_{1 \le i \le P}$ forms a Lagrangian polynomial basis of degree $(P - 1)$. In the deterministic community, (4.7) would be referred to as a modal expansion and (4.12) as a nodal expansion. Since θ in (4.12) is a Gaussian variable with zero mean and

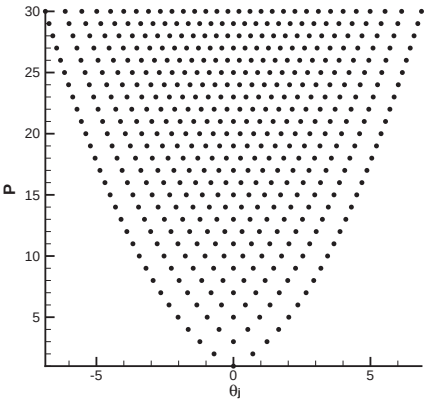


FIG. 4.1. *Distribution of Gauss–Hermite collocation points for* $P = 1, \ldots, 30$.

unit variance, it seems natural to use Gauss–Hermite collocation points as the basis for the Lagrange polynomials, as that also enables accurate evaluation of integrals. Figure 4.1 shows the collocation $\{\theta_j\}_{1\le j\le P}$ points for $P = 1, \ldots, 30$. It should be noted that those points are more concentrated near zero, where the probability of the normal law is highest.

For reasons to be explained shortly, the Lagrange polynomials of (4.12) will not be based on θ_j but on $\sqrt{2}\theta_j$. Therefore, by the property of Lagrange polynomials $L_i(\sqrt{2}\theta_j) = \delta_{ij}$, (4.12) gives

$$\mathbf{q}(\mathbf{x}, t, \sqrt{2}\theta_j) = \mathbf{q}^j(\mathbf{x}, t). \tag{4.13}$$

In the collocation formulation, we require that the residual of (4.8) be zero at collocation points $\{\sqrt{2}\theta_j\}_{1\le j\le P}$, i.e.,

$$\forall j \in [1, P] : \begin{cases} Q^j M \dfrac{d\mathbf{q}_N^j}{dt} + S \cdot \mathbf{F}_N^j - M\mathbf{S}_N^j = F\hat{\mathbf{n}} \cdot [\mathbf{F}_N^j - \mathbf{F}^{j*}], \\ \mathbf{q}_N^j(\mathbf{x}, t = 0) = \mathbf{f}(\mathbf{x}, \sqrt{2}\theta_j). \end{cases} \tag{4.14}$$

The stochastic collocation method is essentially a deterministic or lattice Monte Carlo method with samples $\{\sqrt{2}\theta_j\}_{1\le j\le P}$. The crucial difference is that a typical Monte Carlo simulation consisting of P realizations will converge at the slow rate $1/\sqrt{P}$, whereas the convergence of the stochastic collocation method is much faster, as will be shown by numerical examples (the interested reader can also refer to [17] for classical results in approximation theory). A simple analogy is that the approximation of a smooth function, the PDF, is done most efficiently by representing it by smooth polynomials based on points well suited for interpolation. In a simple Monte Carlo approach, the interpolation points are random, leading to a poor convergence rate.

We presented the stochastic collocation method for a random space of size $d = 1$. The generalization to higher random spaces is straightforward, using tensor products of quadrature points. Note that for higher random spaces and for purposes of efficiency, it might be necessary to use sparse grid methods (see [17] for details).

4.4. A brief comparison and discussion of the two techniques. It is worth highlighting a few essential differences between the two ways of introducing the uncertainty into the model. Although both rely on the use of the chaos expansion, the

two formulations lead to computationally different methods. In the Galerkin method, one recovers a system of P coupled equations, even if a scalar equation is considered initially. While the coupling coefficients can be precomputed, thus reducing the cost, essential questions, such as well-posedness of the system, do not follow trivially from the well-posedness of the scalar equation. However, analysis confirming the well-posedness of the system can be found in [5].

For the collocation approach, however, these problems never arise as one solves P decoupled problems, all of a type similar to that of the original problem. Thus, in terms of computational efficiency, the stochastic collocation method compares favorably to the stochastic Galerkin formulation.

On the other hand, there are situations in which the Galerkin formulation is superior to the collocation approach in terms of accuracy. To appreciate this, consider again (4.1) and assume, for simplicity, that the spatial direction is periodic. We use a Fourier spectral method in this direction; i.e., we assume

$$u(x,t,\theta) = \sum_{|n|\le N} \hat{u}_n(\theta,t)\exp(inx).$$

This yields the semidiscrete scheme

$$\forall |n| \le N \,:\, \frac{d\hat{u}_n(\theta,t)}{dt} + \tilde{a}(\theta)\hat{u}_n = 0,$$

with $\tilde{a}(\theta) = ina(\theta)$.

If we now represent the orthogonal projection of u onto the space spanned by the Hermite polynomial by $\mathcal{P}_P u$, we recover the error equation

$$\frac{d}{dt}\varepsilon + \tilde{a}(\theta)\hat{u}_n - \mathcal{P}_P(\tilde{a}(\theta)\hat{u}_n) = 0,$$

where $\varepsilon(t) = \hat{u}_n - \mathcal{P}_P\hat{u}_n$ with

$$\mathcal{P}_P\hat{u}_n = \sum_{i=1}^{P} \hat{u}_{ni}(t)H_i(\theta).$$

From classic approximation results for orthogonal polynomials [6], we recall

$$\|u - \mathcal{P}_P u\|_w \le P^{-k}\|u^{(k)}\|_w, \quad P > k \ge 0.$$

Here $\|\cdot\|_w$ is the Gaussian weighted inner product and $u^{(k)}$ reflects the kth derivative of u, i.e., it is a simple measure of the smoothness of the probability density associated with u.

Integration in θ and in time immediately yields the error bound

$$\|\varepsilon(t)\|_w \le \|\varepsilon(0)\|_w + tP^{-k}\max_{s\in[0,t]}\|(\tilde{a}\hat{u}_n)^{(k)}(s)\|_w$$

or

$$\|\varepsilon(t)\|_w \le P^{-k}\left(\|\hat{u}_n^{(k)}(0)\|_w + t\max_{s\in[0,t]}\|(\tilde{a}\hat{u}_n)^{(k)}(s)\|_w\right). \tag{4.15}$$

Thus, if $\hat{u}_n^{(k)}(0)$ and $(a\hat{u}_n)^{(k)}$ are well behaved, i.e., reasonably smooth, the scheme converges, although the details naturally depend on $\tilde{a}$ and the initial conditions.

If we now consider the collocation formulation in exactly the same way, we shall need the interpolation estimate [6]

$$\|u - \mathcal{I}_P u\|_w \leq P^{-k+1} \|u^{(k)}\|_w, \quad P > k \geq 1,$$

where

$$\mathcal{I}_P \hat{u}_n = \sum_{i=1}^{P} \hat{u}_n(\theta_i, t) L_i(\theta)$$

is the Lagrange form of the chaos expansion as discussed above. This immediately yields the estimate

$$\|\varepsilon(t)\|_w \leq P^{-k+1} \left(\|\hat{u}_n^{(k)}(0)\|_w + t \max_{s\in[0,t]} \|(\tilde{a}\hat{u}_n)^{(k)}(s)\|_w \right), \tag{4.16}$$

provided $k \geq 1$.

Comparing the bounds in (4.15) and (4.16) highlights several properties of the two different formulations. We notice, as mentioned above, that the Galerkin formulation in general is more accurate than the collocation approach, in particular for problems of low regularity (small k); i.e., for problems with very low regularity ($k \leq 1$) the collocation approach may well fail. Thus, for problems in which the probability density of $\hat{u}_n$ develops kinks or cusps, a Galerkin approach is likely to be advantageous. However, for problems in which both $a(\theta)$ and the probability density of the solution, u, are smooth in θ, the two methods are essentially equivalent in terms of accuracy. As we shall see shortly, for the different types of applications considered here, the two methods behave in very similar ways in terms of accuracy.

The analysis also shows that the error grows linearly in time—at least for problems of simple wave type (4.1). We can obtain a further appreciation of the impact of this by considering the exact solution

$$\hat{u}_n(\theta, t) = \hat{u}_n(\theta, 0) \exp(ina(\theta)t).$$

The length, P, of the chaos expansion must naturally be chosen to represent this solution adequately at all times. While this is easily accomplished for the initial conditions, $\hat{u}_n(\theta, 0)$, the other term reflects a wave, where the wavenumber, $a(\theta)nt$, grows with time. Again borrowing from classic results in approximation theory [8], we know that if we consider

$$\exp(ikx) = \sum_{l=0}^{L} \hat{u}_l H_l(x),$$

then exponential accuracy requires that

$$\frac{2^l (l/2)!}{k^l} \exp\left(\frac{l^2}{4}\right) > \exp\left(\frac{1}{2}\left(\frac{l\pi}{kp}\right)^2\right)$$

at a fixed value of x. Here

$$p = \frac{\lambda}{2x/l}$$

reflects the number of degrees of freedom used to represent the wave of wavelength λ.

Thus, if we want to accurately resolve waves within $x \in [-\pi, \pi]$, we find that $lp^{-1} = 1$, and the requirement above is fulfilled for $l = p > 6$. However, if we are interested in resolving the problem in $x \in [-2\pi, 2\pi]$, then $lp^{-1} = 2$, and we obtain $l = 2p > 15$, while $x \in [-4\pi, 4\pi]$ implies $lp^{-1} = 4$ and the bound $l = 4p > 44$.

These simple estimates have a few consequences for the methods discussed here. In particular, we find in the above that the effective wavenumber will grow linearly with time. This means that if we want to ensure an accurate representation of the local (in space) probability density, we shall generally expect that P, the number of terms in the chaos expansion, must grow linearly.

This requires, however, that the local PDF does not change too much over time; i.e., if it expands significantly in θ space beyond the initial range, we must ensure that it is well represented within a larger range of θ values, which requires, as seen above, a faster than linear growth. These arguments are independent of whether we consider the Galerkin or the collocation approach.

4.5. Computation of statistics. In practice, an expression of the stochastic fields in the form (4.7) or (4.12), containing a full space-time approximation to the local PDF, is of little use and often contains too much information. Quantities of interest—observables—are often macroscopic quantities, i.e., averages or variances of the fields. The way to extract those quantities depends on whether we have used a collocation formulation or a Galerkin formulation, and the procedure will also be different depending on whether we are interested in the statistics of linear or nonlinear functions of the electric or magnetic field. We shall sketch the different procedures in the next subsections, as this is an important component of the complete algorithm.

4.5.1. Statistics of linear quantities. As the treatment of the two methods is slightly different, we shall discuss each separately.

[illegible]. We assume that the solution $\mathbf{q}(\mathbf{x}, t, \theta)$ is available in the form (4.7), and we wish to extract some statistical information such as the average solution or its variance.

Let us first assume that we seek the moments of the solutions or a linear combination of them. Taking the average of (4.7), we get

$$\begin{aligned}\langle \mathbf{q}(\mathbf{x}, t, \theta), 1\rangle &= \sum_{i=1}^{P} \mathbf{q}^i(\mathbf{x}, t)\, \langle 1, \Psi_i(\theta)\rangle = \sum_{i=1}^{P} \mathbf{q}^i(\mathbf{x}, t)\, \langle \Psi_1(\theta), \Psi_i(\theta)\rangle \\ &= \sum_{i=1}^{P} \mathbf{q}^i(\mathbf{x}, t)\delta_{1i} = \mathbf{q}^1(\mathbf{x}, t). \end{aligned} \tag{4.17}$$

Thus, the average is simply the first mode in the chaos expansion. In a similar way, we can obtain the variance by first computing

$$\begin{aligned}\langle \mathbf{q}(\mathbf{x}, t, \theta), \mathbf{q}(\mathbf{x}, t, \theta)\rangle &= \sum_{i=1}^{P}\sum_{j=1}^{P} \mathbf{q}^i(\mathbf{x}, t)\mathbf{q}^j(\mathbf{x}, t)\, \langle \Psi_i(\theta), \Psi_i(\theta)\rangle \\ &= \sum_{i=1}^{P}\sum_{j=1}^{P} \mathbf{q}^i(\mathbf{x}, t)\mathbf{q}^j(\mathbf{x}, t) i!\delta_{ij} = \sum_{i=1}^{P} i!\mathbf{q}^i(\mathbf{x}, t)^2. \end{aligned} \tag{4.18}$$

The variance of the solution is then

$$
\begin{aligned}
var &= \langle \mathbf{q}(\mathbf{x},t,\theta), \mathbf{q}(\mathbf{x},t,\theta)\rangle - \langle \mathbf{q}(\mathbf{x},t,\theta), 1\rangle^2 \\
&= \sum_{i=1}^{P} i!\mathbf{q}^i(\mathbf{x},t)^2 - \mathbf{q}^1(\mathbf{x},t)^2 = \sum_{i=2}^{P} i!\mathbf{q}^i(\mathbf{x},t)^2.
\end{aligned} \tag{4.19}
$$

. We now assume that the solution $\mathbf{q}(\mathbf{x},t,\theta)$ is available in the form (4.12). First, we start by recalling that we have chosen to use the Gauss–Hermite collocation points $\{\theta_j\}_{1\le j\le P}$; i.e., we have a quadrature formula that is exact for all polynomial $f(x)$ of degree at most $2P-1$,

$$
\int_{-\infty}^{+\infty} e^{-\theta^2} f(\theta) d\theta = \sum_{j=1}^{P} \omega_j f(\theta_j), \tag{4.20}
$$

where $\{\omega_j\}_{1\le j\le P}$ are the integration weight [4]. This formula can be used to compute the average of $\mathbf{q}(\mathbf{x},t,\theta)$ given by (4.12):

$$
\langle \mathbf{q}(\mathbf{x},t,\theta)\rangle = \sum_{i=1}^{P} \mathbf{q}^i(\mathbf{x},t) \int_{-\infty}^{+\infty} \frac{1}{\sqrt{2\pi}} e^{-\frac{\theta^2}{2}} L_i(\theta) d\theta. \tag{4.21}
$$

After a simple change of variable and using (4.20), we get

$$
\langle \mathbf{q}(\mathbf{x},t,\theta)\rangle = \sum_{i=1}^{P} \mathbf{q}^i(\mathbf{x},t) \sum_{j=1}^{P} \frac{1}{\sqrt{\pi}} \omega_j L_i\left(\sqrt{2}\theta_j\right). \tag{4.22}
$$

This equation shows why we have decided to define Lagrange polynomials based on the collocation points $\sqrt{2}\theta_j$ instead of θ_j. Indeed, using the fact that $L_i(\sqrt{2}\theta_j) = \delta_{ij}$, the above equation simplifies as

$$
\langle \mathbf{q}(\mathbf{x},t,\theta)\rangle = \sum_{j=1}^{P} \frac{1}{\sqrt{\pi}} \omega_j \mathbf{q}^j(\mathbf{x},t). \tag{4.23}
$$

The procedure to compute the variance is similar, leading to

$$
var = \sum_{j=1}^{P} \frac{1}{\sqrt{\pi}} \omega_j \left(\mathbf{q}^j(\mathbf{x},t)\right)^2 - \langle \mathbf{q}(\mathbf{x},t,\theta)\rangle^2. \tag{4.24}
$$

It is important to emphasize that the simple, and most expensive, approach of using a tensor product grid of Gauss quadratures is only one among many alternatives. This is discussed in detail in [17], where special attention is given to methods of relevance to problems discussed here.

4.5.2. Statistics of nonlinear quantities. Often, however, we are interested in the statistics of some derived, possibly nonlinear, functional, $F(\mathbf{q})$ of $\mathbf{q}(\mathbf{x},t,\theta)$, e.g., computation of the impact on the RCS of the uncertainty in the scattering problem.

Let us utilize the idea behind the collocation approach and write

$$
F(\mathbf{q}(\theta)) = \sum_{j=1}^{P} F(\mathbf{q}(\theta_j)) L_j(\theta).
$$

Thus, we simply need to evaluate the general functionals at the values of θ_j. However, since we have already obtained full probabilistic information in the expansions, (4.7) or (4.12), we can use these results directly to obtain the required information and, thus, the probabilistic information on $F(\mathbf{q})$. All informations of interest, e.g., moments, can now be extracted from this in the same way as for the simple variables. Naturally, one can evaluate the integrals using a classic Monte Carlo approach. This can be done at little cost since it requires only evaluation of the expansions and not solution of Maxwell's equations.

For the Galerkin method, one just needs an extra step to express the fields from a modal polynomial basis (4.7) to a nodal polynomial basis (4.12), where the coefficients $\mathbf{q}^j(\mathbf{x},t)$ are computed by simple evaluation of $\mathbf{q}^j(\mathbf{x},t) = \sum_{i=1}^{P} \mathbf{q}^i(\mathbf{x},t)\Psi_i(\sqrt{2}\theta_j)$. Then, the procedure is the same as that for the collocation method.

5. Numerical examples. In the following we shall discuss a few examples used to validate the approach discussed above. These results are chosen largely to be simple enough to enable rigorous testing as well as to expose the generality and strength of the proposed technique.

5.1. One-dimensional material loaded cavity. As a first simple test case, we consider a one-dimensional PEC cavity loaded with two media, with a material interface at $x^{mat} = 0$ and perfectly conducting walls at $x_1^{pec} = -L$ and $x_2^{pec} = L$, as shown in Figure 5.1. The aim of the test is to compute the first few resonance frequencies of the cavity. For the simple case considered here, these frequencies are given as

$$\sqrt{\epsilon_1}\tan(\omega\sqrt{\epsilon_2}(L - x^{mat})) = -\sqrt{\epsilon_2}\tan(\omega\sqrt{\epsilon_1}(L + x^{mat})), \tag{5.1}$$

where ϵ_k reflects the two different permittivities. We assume for simplicity that the materials are nonmagnetic, i.e., $\mu_k = 1$, and solve on each domain D_k the equations

$$Q\frac{\partial \mathbf{q}}{\partial t} + \frac{\partial}{\partial x}\mathbf{F}(\mathbf{q}) = 0, \tag{5.2}$$

where

$$\mathbf{q} = \begin{pmatrix} \mathbf{E}_k^y \\ \mathbf{H}_k^z \end{pmatrix}, \quad \mathbf{F}(\mathbf{q}) = \begin{pmatrix} \mathbf{H}_k^z \\ \mathbf{E}_k^y \end{pmatrix}, \quad Q = \begin{pmatrix} \epsilon_k & 0 \\ 0 & \mu_k \end{pmatrix}. \tag{5.3}$$

To compute the resonant frequencies, we solve the one-dimensional Maxwell's equations in the time-domain, subject to a broadband initial condition, and collect one or several time-traces at various points in the cavity. The spectrum of the time-series yields the resonant frequencies as strong peaks which are found automatically.

5.1.1. Uncertainty in material. In this first test we assume an uncertainty or randomness in the permittivity of the material in domain D_2 of the form

$$\varepsilon(\mathbf{x},\theta) = \begin{cases} 1 & \text{if } \mathbf{x} \notin D_2, \\ 2.25\left(1 + 0.1\dfrac{\theta^2}{1+\theta^2}\right) & \text{otherwise,} \end{cases} \tag{5.4}$$

where, as everywhere else, θ is a Gaussian variable with zero mean and unit variance. This model ensures that the permittivity will remain positive in the domain D_2. Other parameters were fixed as $L = 1$ and $\mu_1 = \mu_2 = 1$.

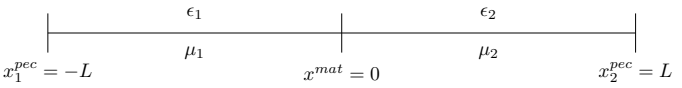


FIG. 5.1. *Illustration of one-dimensional loaded cavity.*

TABLE 5.1

Numerical mean and variance of resonance frequencies for cavity with random material. The discrepancy in the variance for ω_1 is caused by quantization errors in the frequency identification approach and is not related to the accuracy of the modeling of the uncertainty. This is confirmed by having an identical error in the Monte Carlo model.

Mean of Resonance Frequencies for Uncertainty in Material											
	iterations	ω_1	ω_2	ω_3	ω_4	ω_5	ω_6	ω_7	ω_8	ω_9	ω_{10}
Exact Formula	3600	1.1955	2.5689	3.6543	5.0158	6.2310	7.4079	8.7898	9.8795	11.2263	12.4614
	7200	1.1957	2.5692	3.6549	5.0168	6.2319	7.4093	8.7913	9.8811	11.2285	12.4632
	14400	1.1954	2.5688	3.6541	5.0155	6.2307	7.4074	8.7894	9.8790	11.2257	12.4609
Monte Carlo	300	1.1938	2.5715	3.6620	5.0209	6.2350	7.4083	8.7895	9.8837	11.2325	12.4675
	600	1.1938	2.5712	3.6618	5.0230	6.2346	7.4102	8.7916	9.8869	11.2341	12.4680
	1200	1.1938	2.5691	3.6583	5.0185	6.2301	7.4050	8.7862	9.8803	11.2262	12.4611
Stochastic Galerkin $P = 40$	3600	1.1938	2.5705	3.6595	5.0214	6.2313	7.4109	8.7936	9.8836	11.2214	12.4753
	7200	1.1938	2.5708	3.6603	5.0225	6.2328	7.4126	8.7952	9.8842	11.2231	12.4775
	14400	1.1938	2.5706	3.6594	5.0213	6.2316	7.4111	8.7937	9.8826	11.2203	12.4752
Stochastic Galerkin $P = 100$	3600	1.1938	2.5705	3.6592	5.0196	6.2330	7.4076	8.7901	9.8826	11.2291	12.4639
	7200	1.1938	2.5707	3.6595	5.0199	6.2330	7.4074	8.7901	9.8828	11.2293	12.4645
	14400	1.1938	2.5707	3.6599	5.0205	6.2334	7.4082	8.7908	9.8836	11.2300	12.4652

Variance of Resonance Frequencies for Uncertainty in Material											
	iterations	ω_1	ω_2	ω_3	ω_4	ω_5	ω_6	ω_7	ω_8	ω_9	ω_{10}
Exact Formula	3600	1.41e-04	2.81e-04	8.40e-04	1.87e-03	1.58e-03	4.32e-03	4.21e-03	5.26e-03	9.82e-03	6.42e-03
	7200	1.38e-04	2.75e-04	8.23e-04	1.83e-03	1.54e-03	4.23e-03	4.11e-03	5.15e-03	9.60e-03	6.28e-03
	14400	1.38e-04	2.76e-04	8.25e-04	1.83e-03	1.55e-03	4.24e-03	4.13e-03	5.16e-03	9.64e-03	6.30e-03
Monte Carlo	300	2.62e-29	2.69e-04	1.28e-03	2.41e-03	1.98e-03	4.03e-03	4.06e-03	5.72e-03	9.89e-03	7.04e-03
	600	2.15e-28	2.85e-04	1.36e-03	2.43e-03	2.09e-03	4.06e-03	4.13e-03	5.65e-03	9.92e-03	7.02e-03
	1200	1.66e-28	3.92e-04	1.49e-03	2.44e-03	2.30e-03	4.47e-03	4.52e-03	6.13e-03	1.08e-02	7.46e-03
Stochastic Galerkin $P = 40$	3600	4.06e-27	3.23e-04	1.43e-03	2.40e-03	2.18e-03	3.84e-03	3.78e-03	5.62e-03	9.76e-03	7.66e-03
	7200	5.84e-27	3.06e-04	1.42e-03	2.32e-03	2.18e-03	3.77e-03	3.72e-03	5.45e-03	9.62e-03	7.58e-03
	14400	2.74e-07	3.17e-04	1.41e-03	2.32e-03	2.18e-03	3.78e-03	3.72e-03	5.53e-03	9.61e-03	7.68e-03
Stochastic Galerkin $P = 100$	3600	4.06e-27	3.19e-04	1.42e-03	2.35e-03	2.12e-03	4.23e-03	4.19e-03	5.86e-03	1.03e-02	6.82e-03
	7200	5.84e-27	3.10e-04	1.41e-03	2.38e-03	2.16e-03	4.29e-03	4.24e-03	5.89e-03	1.06e-02	7.12e-03
	14400	6.82e-27	3.11e-04	1.42e-03	2.38e-03	2.13e-03	4.26e-03	4.21e-03	5.89e-03	1.02e-02	7.07e-03

For the stochastic Galerkin method with random permittivity, the terms Q, Z, $\overline{Z}$, Y, $\overline{Y}$ in the semidiscrete equation are all dependent on $\varepsilon(\mathbf{x}, \theta)$ so that the equations are given by (4.9). The case of random permittivity gives rise to a coupled system of P deterministic equations, where the coupling is through the many scalar products defined in (4.9). These scalar products are computed in the preprocessing stage and stored.

[illegible]. We evaluate the technique by computing the mean and variance of the resonance frequencies of the loaded cavity. In Table 5.1 we show values computed by three different methods. The results labeled "Exact Formula" are obtained by performing Monte Carlo sampling on the equation that describes the resonance frequencies (5.1). As a benchmark, we show results obtained using a standard Monte Carlo method on (5.2) with up to 1200 samples. The frequencies in the stochastic Galerkin method are computed using the method described in section 4.5.1. Results for two values of P, 40 and 100, are shown. The results show good agreement between the stochastic Galerkin approach and the statistics of the exact solution as well as the Monte Carlo method, although it is not clear that the latter results are

converged. The tables also show that while increasing P does increase the accuracy of the data, the gain in accuracy is not significant. Note that the iteration in the Monte Carlo solutions amounts to actually solving the Maxwell's equations, i.e., the computational time scales with the number of iterations. For the chaos expansion method, the computational work scales with P as the iterations simply amount to summing the final chaos expansions. Thus, the cost of the Monte Carlo approach is many times larger than the cost of using the techniques discussed here.

5.1.2. Uncertainty in boundary position. Let us now consider the case when the length of the cavity is unknown, i.e., when there is uncertainty associated with the position of the boundary. We choose to have uncertainty in the right boundary position so that the domain D_2 is of variable length. The position of the right boundary is assumed to be $x_2^{pec} = L + g(\theta)$, where

$$g(\theta) = 0.1\left(\frac{\theta}{1+\theta^2}\right). \tag{5.5}$$

All other parameters are fixed as $L = 1$, $\mu_1 = \mu_2 = 1$, $\epsilon_1 = 1$, and $\epsilon_2 = 2.25$.

For the stochastic Galerkin method with a random boundary position, we introduce a linear mapping from the variable x into the new variable $\xi \in [-1, 1]$, i.e., it is of fixed length. The mapping of the variable domain D_2 into a fixed domain of length L is

$$\xi = x^{mat} + (L - x^{mat})\frac{x - x^{mat}}{(L + g(\theta) - x^{mat})}. \tag{5.6}$$

We can then express (5.2) in the new variable ξ by

$$Q\frac{\partial \mathbf{q}}{\partial t} + \frac{\partial \xi}{\partial x}\frac{\partial \mathbf{F}(\mathbf{q})}{\partial \xi} = 0, \tag{5.7}$$

which is stated in a fixed computational domain. Note that this transformation should also be applied to the initial conditions. As the equations have changed only by the multiplication of the additional term $\frac{\partial \xi}{\partial x}$, and we now have a deterministic Q, the computational scheme is expressed as

$$\forall k \in [1, P] \,:\, QMk!\frac{d\mathbf{q}_N^k}{dt} + S\sum_{i=1}^{P}\left\langle \frac{\partial \xi}{\partial x}\Psi_i, \Psi_k \right\rangle \cdot \mathbf{F}_N^i = F\sum_{i=1}^{P}\hat{\mathbf{n}} \cdot [\mathbf{F}_N^i - \mathbf{F}^{i*}], \tag{5.8}$$

$$\hat{\mathbf{n}} \cdot [\mathbf{F}_N^i - \mathbf{F}^{i*}] = \left\langle \frac{\partial \xi}{\partial \mathbf{x}}\Psi_i, \Psi_k \right\rangle \begin{pmatrix} \overline{Z}^{-1}\hat{\mathbf{n}} \times \hat{\mathbf{n}} \times [\mathbf{E}^i] - Z^+\overline{Z}^{-1}\hat{\mathbf{n}} \times [\mathbf{H}^i] \\ \overline{Y}^{-1}\hat{\mathbf{n}} \times \hat{\mathbf{n}} \times [\mathbf{H}^i] + Y^+\overline{Y}^{-1}\hat{\mathbf{n}} \times [\mathbf{E}^i] \end{pmatrix}. \tag{5.9}$$

Thus, the random domain problem has been transformed into a random coefficient problem similar to the one discussed previously. Note that we apply this method in both domains, although in domain D_1, which is already of fixed length, the value of $\frac{\partial \xi}{\partial \mathbf{x}}$ is one.

. We compute again the mean and variance of the resonance frequencies of the loaded cavity. Since the boundary moves only in a small region of the domain $D_2 = [x^{mat}, x_2^{pec}]$, we split the domain D_2 into two artificial subdomains $D_{21} = [x^{mat}, x^{split}]$, $D_{22} = [x^{split}, x_2^{pec}]$ such that $D_2 = D_{21} \cup D_{22}$ and x^{split} is chosen depending on how far the boundary can move inward. Then we can apply the

TABLE 5.2
Numerical mean and variance of resonance frequencies for a cavity with an uncertain boundary.

Mean of Resonance Frequencies for Uncertainty in Boundary Position											
	iterations	ω_1	ω_2	ω_3	ω_4	ω_5	ω_6	ω_7	ω_8	ω_9	ω_{10}
Exact Formula	3600	1.2111	2.5906	3.6943	5.0714	6.2847	7.4973	8.8684	9.9860	11.3513	12.5698
	7200	1.2111	2.5904	3.6941	5.0712	6.2844	7.4971	8.8678	9.9857	11.3511	12.5695
	14400	1.2111	2.5904	3.6941	5.0712	6.2843	7.4969	8.8679	9.9854	11.3508	12.5691
Stochastic Galerkin $P = 40$	3600	1.2199	2.5830	3.6943	5.0654	6.2871	7.4874	8.8631	9.9898	11.3518	12.5519
	7200	1.2203	2.5834	3.6947	5.0652	6.2876	7.4881	8.8638	9.9903	11.3523	12.5534
	14400	1.2205	2.5832	3.6942	5.0644	6.2874	7.4883	8.8637	9.9900	11.3514	12.5529
Stochastic Galerkin $P = 100$	3600	1.2207	2.5820	3.6900	5.0693	6.2894	7.4934	8.8623	9.9982	11.3619	12.5679
	7200	1.2205	2.5820	3.6896	5.0682	6.2886	7.4915	8.8606	9.9965	11.3597	12.5661
	14400	1.2210	2.5821	3.6899	5.0691	6.2895	7.4930	8.8620	9.9981	11.3615	12.5675

Variance of Resonance Frequencies for Uncertainty in Boundary Position											
	iterations	ω_1	ω_2	ω_3	ω_4	ω_5	ω_6	ω_7	ω_8	ω_9	ω_{10}
Exact Formula	3600	2.39e-04	1.90e-03	3.88e-03	4.20e-03	1.53e-02	9.28e-03	2.22e-02	2.84e-02	2.27e-02	6.07e-02
	7200	2.43e-04	1.94e-03	3.95e-03	4.27e-03	1.55e-02	9.44e-03	2.25e-02	2.88e-02	2.37e-02	6.22e-02
	14400	2.39e-04	1.90e-03	3.88e-03	4.20e-03	1.52e-02	9.27e-03	2.22e-02	2.83e-02	2.29e-02	6.08e-02
Stochastic Galerkin $P = 40$	3600	9.59e-04	2.89e-03	7.56e-03	1.90e-02	1.25e-02	3.74e-02	3.93e-02	5.03e-02	8.90e-02	6.92e-02
	7200	9.63e-04	2.90e-03	7.64e-03	1.91e-02	1.25e-02	3.81e-02	3.97e-02	5.08e-02	8.97e-02	6.98e-02
	14400	9.65e-04	2.93e-03	7.66e-03	1.93e-02	1.26e-02	3.85e-02	4.01e-02	5.12e-02	9.07e-02	7.01e-02
Stochastic Galerkin $P = 100$	3600	9.67e-04	3.14e-03	7.69e-03	1.85e-02	1.41e-02	3.84e-02	4.16e-02	5.00e-02	9.05e-02	6.27e-02
	7200	9.65e-04	3.14e-03	7.74e-03	1.83e-02	1.41e-02	3.80e-02	4.12e-02	4.95e-02	9.00e-02	6.21e-02
	14400	9.69e-04	3.18e-03	7.78e-03	1.85e-02	1.42e-02	3.84e-02	4.16e-02	5.00e-02	9.05e-02	6.27e-02

transformation described in the previous section only to the subdomain D_{22}. This matches the physical properties of the case of an uncertain boundary more closely than mapping the entire domain D_2.

Table 5.2 shows statistics of the resonance frequencies computed by the stochastic Galerkin method and sampling of the exact solution. Since the Monte Carlo method is computationally far more expensive, we do not repeat the tests for all cases. The results again indicate that the stochastic Galerkin approach is in excellent agreement with the exact solution.

5.1.3. Uncertainty in the interface position. In this last example, we focus on the cavity where the position of an interface is uncertain, such that both domains are of variable length. We define the position of the material interface as $x^{mat} = 0 + g(\theta)$, where

$$g(\theta) = 0.1\left(\frac{\theta}{1+\theta^2}\right). \tag{5.10}$$

This allows the material interface to be positioned on both sides of the mean position. We fix other parameters as $L = 1$, $\mu_1 = \mu_2 = 1$, $\epsilon_1 = 1$, and $\epsilon_2 = 2.25$.

For the stochastic Galerkin method with random position of the interface, we proceed in a manner similar to that for the random boundary case and introduce a mapping of both domains from the variable x to the new variable ξ. The mapping of the variable domains D_1 and D_2 into domains of fixed length L is defined as

$$\xi = \begin{cases} -L + L\dfrac{(x+L)}{(g(\theta)+L)} & \text{if } x \in D_1, \\ L\dfrac{(x-g(\theta))}{(L-g(\theta))} & \text{if } x \in D_2. \end{cases} \tag{5.11}$$

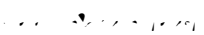
The rest follows from (5.7)–(5.9).

TABLE 5.3

Numerical mean and variance of resonance frequencies for cavity with an interface at random position.

Mean of Resonance Frequencies for Uncertainty in Interface Position											
	iterations	ω_1	ω_2	ω_3	ω_4	ω_5	ω_6	ω_7	ω_8	ω_9	ω_{10}
Exact Formula	3600	1.2115	2.5884	3.6975	5.0686	6.2832	7.5054	8.8486	10.0084	11.3353	12.5667
	7200	1.2116	2.5885	3.6976	5.0691	6.2832	7.5063	8.8491	10.0089	11.3367	12.5672
	14400	1.2116	2.5886	3.6976	5.0689	6.2832	7.5063	8.8488	10.0093	11.3361	12.5671
Stochastic Galerkin $P = 40$	3600	1.2166	2.5761	3.6902	5.0674	6.2832	7.5004	8.8567	10.0026	11.3383	12.5672
	7200	1.2166	2.5761	3.6906	5.0682	6.2832	7.5015	8.8575	10.0030	11.3401	12.5677
	14400	1.2168	2.5761	3.6902	5.0679	6.2832	7.5015	8.8571	10.0032	11.3391	12.5672
Stochastic Galerkin $P = 100$	3600	1.2172	2.5761	3.6903	5.0683	6.2832	7.5140	8.8631	10.0039	11.3378	12.5670
	7200	1.2166	2.5761	3.6908	5.0672	6.2832	7.5122	8.8621	10.0026	11.3353	12.5678
	14400	1.2166	2.5761	3.6905	5.0674	6.2832	7.5120	8.8626	10.0030	11.3361	12.5675

Variance of Resonance Frequencies for Uncertainty in Interface Position											
	iterations	ω_1	ω_2	ω_3	ω_4	ω_5	ω_6	ω_7	ω_8	ω_9	ω_{10}
Exact Formula	3600	2.85e-04	2.24e-04	4.71e-04	4.76e-03	1.70e-05	1.01e-02	2.94e-03	4.05e-03	2.02e-02	9.08e-04
	7200	2.84e-04	2.23e-04	4.70e-04	4.75e-03	1.69e-05	1.04e-02	3.13e-03	4.34e-03	2.06e-02	1.34e-03
	14400	2.88e-04	2.26e-04	4.76e-04	4.82e-03	1.73e-05	1.06e-02	3.27e-03	4.53e-03	2.09e-02	1.41e-03
Stochastic Galerkin $P = 40$	3600	9.12e-04	3.65e-28	7.76e-04	5.83e-03	1.54e-25	9.08e-03	3.74e-03	5.50e-03	2.03e-02	2.13e-03
	7200	9.13e-04	1.76e-26	7.63e-04	5.79e-03	2.27e-25	9.14e-03	3.73e-03	5.50e-03	2.02e-02	2.38e-03
	14400	9.16e-04	6.90e-27	7.75e-04	5.90e-03	7.78e-26	9.14e-03	3.73e-03	5.56e-03	2.05e-02	2.21e-03
Stochastic Galerkin $P = 100$	3600	9.23e-04	3.65e-28	7.73e-04	6.08e-03	1.54e-25	9.08e-03	3.26e-03	5.91e-03	2.22e-02	1.88e-03
	7200	9.13e-04	1.76e-26	7.59e-04	5.94e-03	2.27e-25	9.03e-03	3.22e-03	5.88e-03	2.20e-02	2.16e-03
	14400	9.12e-04	6.90e-27	7.67e-04	5.96e-03	7.78e-26	8.92e-03	3.20e-03	5.84e-03	2.17e-02	2.28e-03



. We split the domains into smaller regions, as described in the previous case, in order to match the physical properties of the cavity. In this case, the mapping is applied only to the two subdomains adjacent to the material interface. Table 5.3 shows statistics of the resonance frequencies computed by the stochastic Galerkin method and sampling of the exact solution. The quality of the results are similar to those of the previous test cases.

5.2. Two-dimensional circular cylinder. As a second and more advanced test, we consider plane wave scattering by a two-dimensional circular cylinder of radius unity. The measure of interest is the RCS, defined as

$$RCS_{db}(\phi) = 10 \log \left(2\pi \frac{|\mathbf{F}(\phi)|^2}{|\mathbf{E}_i|^2} \right), \tag{5.12}$$

where $\mathbf{E}_i$ is the incident field and $\mathbf{F}(\phi)$ is a nonlinear function of $\mathbf{E}(\mathbf{x}, t, \theta)$ and $\mathbf{H}(\mathbf{x}, t, \theta)$, computing the scattered far field as a function of the polar angle, ϕ. In this particular case, $\mathbf{F}(\phi)$ is the near-to-far-field transformation along some closed contour [15].

Typical meshes are represented in Figure 5.2, where the number of elements is 1141. The domain extends from -6 to $+6$, and a perfectly matched layer (PML) of width 1 is used to truncate the domain (see [1] for details). Degree five polynomials are enough to ensure that convergence is achieved in the physical space, and six modes in the chaos expansion (or six nodes for the collocation formulation, i.e., $P = 6$) are also enough for good convergence in probability space.

5.2.1. Uncertainty in the source term. Let us denote by Γ the boundary of a PEC two-dimensional circular cylinder contained in the domain Ω. In this first example, we consider the following model for the uncertainty in the source term appearing in (2.1):

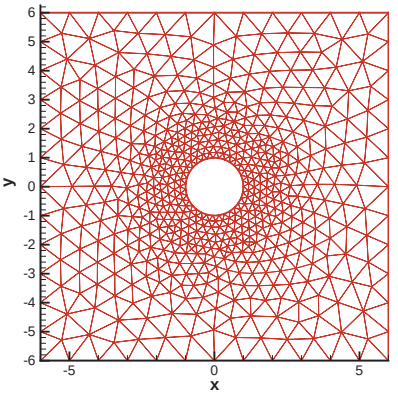


FIG. 5.2. *Typical mesh using* 1141 *spectral elements.*

$$\mathbf{S}^E = \begin{cases} \mathbf{E}_0 \sin(2\pi\omega(\mathbf{k}(\theta).\mathbf{x} - t) & \text{if } \mathbf{x} \in \Gamma, \\ \mathbf{0} & \text{otherwise,} \end{cases} \tag{5.13}$$

where $\mathbf{k}(\theta)$ is an uncertain wave vector. We assume that the value of the wave vector $\mathbf{k}(\theta)$ is subject to a small random perturbation from its deterministic value $\mathbf{k}_0 = (r_0\cos(\alpha_0), r_0\sin(\alpha_0))$. More precisely, we assume that this vector can vary in one of the following two ways:

1. Randomness in the direction of $\mathbf{k}$:

$$\mathbf{k}(\theta) = \begin{pmatrix} r_0\cos(\alpha_0 + 0.1\theta) \\ r_0\sin(\alpha_0 + 0.1\theta) \end{pmatrix}. \tag{5.14}$$

2. Randomness in the modulus of $\mathbf{k}$:

$$\mathbf{k}(\theta) = \begin{pmatrix} r_0(1 + 0.1\theta)\cos(\alpha_0) \\ r_0(1 + 0.1\theta)\sin(\alpha_0) \end{pmatrix}. \tag{5.15}$$

The latter case mimics a slight variation in the frequency of the source.

. For the stochastic Galerkin method, the source term in the semidiscrete equation (4.9) becomes

$$\mathbf{S}_N^k = \begin{cases} \mathbf{E}_0\langle \sin(2\pi\omega(\mathbf{k}(\theta)\cdot\mathbf{x} - t)), \Psi_k\rangle & \text{if } \mathbf{x} \in \Gamma, \\ \mathbf{0} & \text{otherwise,} \end{cases} \tag{5.16}$$

or, after a simple trigonometric manipulation,

$$\mathbf{S}_N^k = \begin{cases} \mathbf{E}_0\Big(\langle \sin(2\pi\omega\mathbf{k}(\theta)\cdot\mathbf{x}), \Psi_k\rangle \cos(2\pi\omega t) \\ \qquad\qquad - \langle \cos(2\pi\omega\mathbf{k}(\theta)\cdot\mathbf{x}), \Psi_k\rangle \sin(2\pi\omega t)\Big) & \text{if } \mathbf{x} \in \Gamma, \\ \mathbf{0} & \text{otherwise.} \end{cases} \tag{5.17}$$

The coefficients $\langle \sin(2\pi\omega\mathbf{k}(\theta)\cdot\mathbf{x}), \Psi_k\rangle$ and $\langle \cos(2\pi\omega\mathbf{k}(\theta)\cdot\mathbf{x}), \Psi_k\rangle$ in the above expression can be precomputed, stored, and then used to impose the boundary conditions at any subsequent time steps. Furthermore, by assuming that the permittivity ϵ and the permeability μ are deterministic, using the orthogonality of Hermite polynomials (4.5), equation (4.9) simplifies to

$$\forall k \in [1, P] \,:\, k!QM\frac{d\mathbf{q}_N^k}{dt} + k!S\cdot\mathbf{F}_N^k - M\mathbf{S}_N^k = F\hat{\mathbf{n}}\cdot[\mathbf{F}_N^k - \mathbf{F}^{k*}]. \tag{5.18}$$

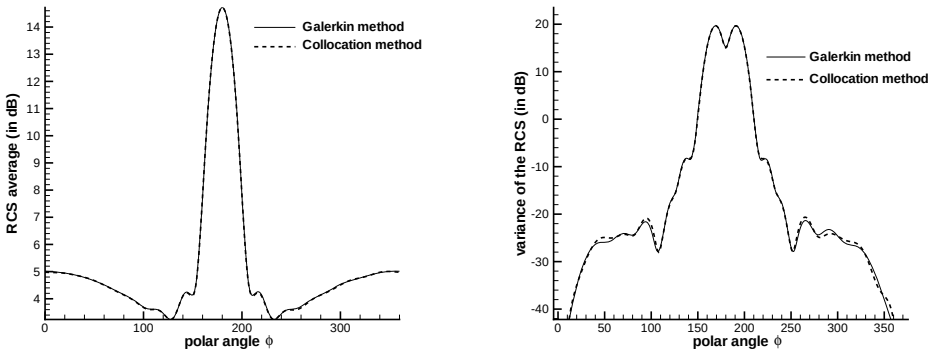


FIG. 5.3. *Comparison of the stochastic Galerkin method and the stochastic collocation method. Left: Average of the RCS for the randomness in the angle of the wave. Right: Variance of the RCS for the randomness in the angle of the wave vector.*

The above equation highlights that for this test case, all the modes are decoupled, and that the cost of solving it, compared to a deterministic simulation, is simply proportional to the number of modes in the chaos expansion. For this particular case, the costs of the Galerkin and collocation methods are the same.

. For the stochastic collocation method, the source term in the semidiscrete equation (4.14) becomes

$$\mathbf{S}_N^j = \begin{cases} \mathbf{E}_0 \sin(2\pi\omega(\mathbf{k}(\sqrt{2}\theta_j)\cdot\mathbf{x} - t)) & \text{if } \mathbf{x}\in\Gamma, \\ \mathbf{0} & \text{otherwise.} \end{cases} \tag{5.19}$$

As always for the stochastic collocation technique, the nodal solutions $\mathbf{q}_N^j$ are decoupled from each other and are obtained by solving P independent systems of the form

$$\forall j \in [1,P] \,:\, QM\frac{d\mathbf{q}_N^j}{dt} + S\cdot\mathbf{F}_N^j - M\mathbf{S}_N^j = F\hat{\mathbf{n}}\cdot[\mathbf{F}_N^j - \mathbf{F}^{j*}]. \tag{5.20}$$

. We conduct the first test to compare the results of the average and variance of the RCS computed by the stochastic collocation method and the stochastic Galerkin method. For the numerical experiments, we set $r_0 = 1$ and $\alpha_0 = \pi$ in (5.14)–(5.15). Figure 5.3 shows the average and the variance of the RCS, and it can be seen that for the same discretization, they give identical results in agreement with the discussion in section 4.4. This case corresponds to uncertainty in the angle of the wave vector, and similar results are obtained for uncertainty in the norm of the wave vector. Figure 5.4 shows the average of the RCS and its possible variations for the randomness in the angle of $\mathbf{k}$ and in the modulus of $\mathbf{k}$. It appears that the uncertainty in the angle of $\mathbf{k}$ primarily affects the RCS in the forward scatter, whereas the uncertainty in the norm of $\mathbf{k}$ mainly affects the RCS in the sidebands.

5.2.2. Uncertainty in materials. We now assume that the permittivity inside a cylindrical domain D is unknown, whereas the permittivity outside this domain is deterministic, e.g.,

$$\varepsilon(\mathbf{x},\theta) = \begin{cases} 1 & \text{if } \mathbf{x}\notin D, \\ 2.25e^{0.1\theta} & \text{otherwise.} \end{cases} \tag{5.21}$$

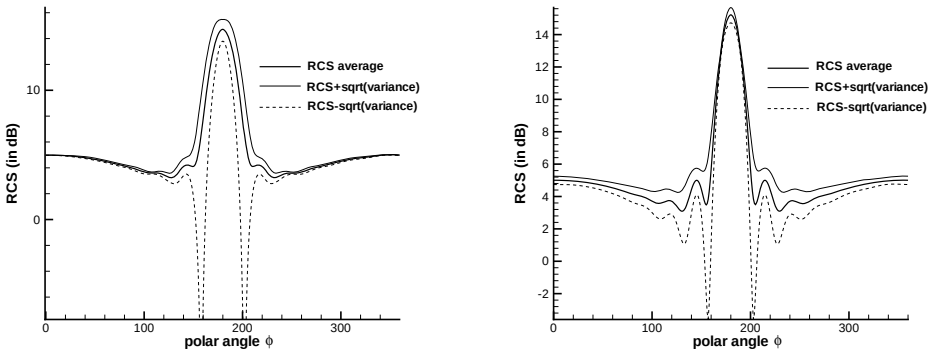


FIG. 5.4. *Left: RCS for the randomness in the angle of the wave vector. Right: RCS for the randomness in the norm of the wave vector. Results are shown with the mean RCS as well as* ± 1 *standard deviation.*

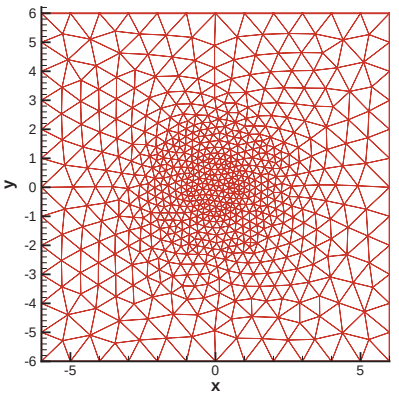


FIG. 5.5. *Typical mesh using* 1318 *spectral elements.*

Here again, our concern is not the correctness of the probabilistic law chosen in (5.21) for the uncertainty of the permittivity $\varepsilon(\mathbf{x}, \theta)$, since for any reasonable law, the techniques presented in this paper should work equally well. With an uncertainty of the form $e^{0.1\theta}$ (which is a log-normal law), the permittivity is guaranteed to remain positive. For uncertainty in the permittivity of the material, the source term of (2.1) takes the form

$$\mathbf{S}^E = -(\epsilon(\mathbf{x}, \theta) - \epsilon^i)\frac{\partial \mathbf{E}^i}{\partial t}, \tag{5.22}$$

where $(\mathbf{E}^i, \mathbf{H}^i)$ denotes the incident field which is a solution to Maxwell's equations. For that test case, it is necessary to mesh the entire domain, as shown in Figure 5.5 (here 1318 spectral elements are represented).

. Projecting the source term using the chaos basis, we obtain

$$\forall k \in [1, P] \ : \ \mathbf{S}_N^k = -\left(\langle \epsilon(\mathbf{x}, \theta), \Psi_k \rangle - \epsilon^i\right) \frac{\partial \mathbf{E}^i}{\partial t}. \tag{5.23}$$

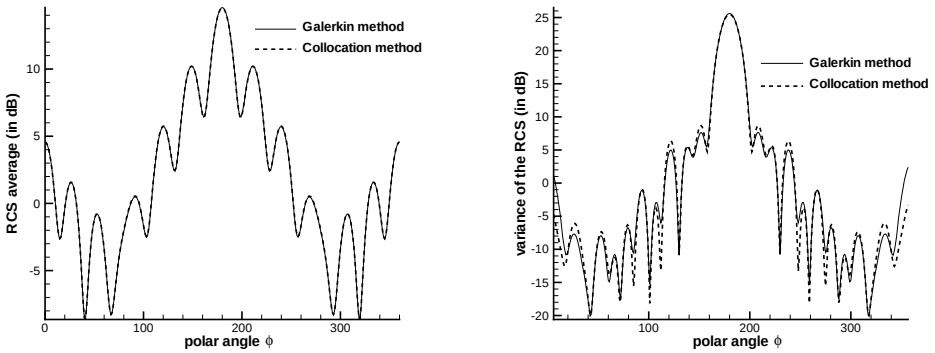


FIG. 5.6. *Comparison of the stochastic Galerkin method and the stochastic collocation method. Left: Average of the RCS for the randomness in material. Right: Variance of the RCS for the randomness in material.*

In this case, the equation to be solved is (4.9) and the source term is given by (5.23). Note that in this case, the size of the coupled system to be solved is P times the size of the system coming from a deterministic simulation.

. For the stochastic collocation method, the source term of (4.14) is simply

$$\mathbf{S}_N^j = -\left(\epsilon(\mathbf{x}, \sqrt{2}\theta_j) - \epsilon^i\right) \frac{\partial \mathbf{E}^i}{\partial t}. \tag{5.24}$$

In that case, the stochastic collocation method is cheaper than the stochastic Galerkin method since it only requires one to solve P decoupled systems of the size of a deterministic problem.

. Here we again compare the results of the RCS for both the stochastic collocation method and the stochastic Galerkin method. Figure 5.6 shows the average and the variance of the RCS. Although the results are not strictly identical, the small differences can be attributed to the nonsmooth nature of the RCS for this problem, which makes convergence more difficult to achieve. Figure 5.7 shows the average of the RCS and the possible variations around its average value. Note that for this test case, and contrary to the previous case, the whole RCS is affected by the uncertainty.

5.2.3. Uncertainty in shape. For this last example, the exact location of the boundary of an object is assumed to be unknown. One way to solve this problem is to generate different meshes for different radii of the cylinder and to perform a Monte Carlo simulation with those meshes. Since this cannot be achieved in an automatic and efficient way, in this paper we proceed differently, inspired by the one-dimensional approach based on mappings.

The idea is to keep the same mesh, the mean mesh, and to randomly move one side of the triangles sitting on the cylinder surface. For example, the triangle $(\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3)$ has the side $(\mathbf{u}_1, \mathbf{u}_3)$ sitting on the cylinder surface (see Figure 5.8). For the deterministic case, the physical element is mapped into the parent element $\mathcal{T}$ and the transformation is written as

$$\mathbf{x}(\xi, \eta) = (1 - \xi - \eta)\mathbf{u}_1 + \xi\mathbf{u}_2 + \eta\mathbf{u}_3 \quad \text{for } (\xi, \eta) \in \mathcal{T}. \tag{5.25}$$

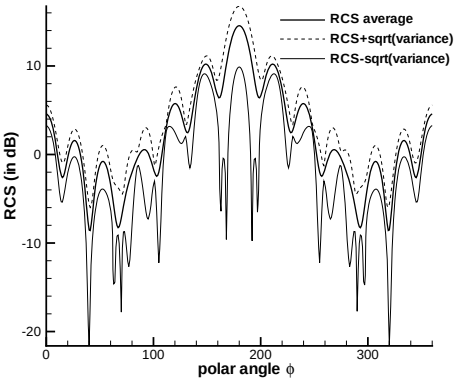


FIG. 5.7. *RCS for uncertainty in the material. Results are shown with the mean RCS as well as* ± 1 *standard deviation.*

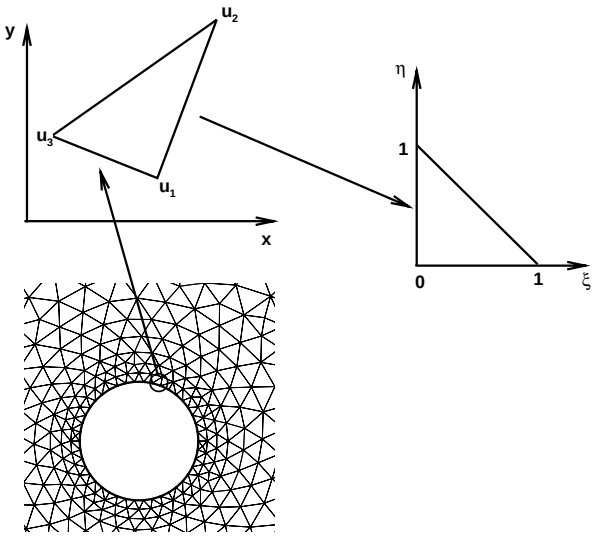


FIG. 5.8. *Illustration for the randomness in shape.*

Note that the Jacobian $J(\xi,\eta) = (\frac{\partial \mathbf{x}}{\partial \eta}, \frac{\partial \mathbf{x}}{\partial \xi})^T$ of transformation (5.25) is only a function of (ξ,η). Assuming randomness of the form $e^{0.1\theta}$, the points $\mathbf{u}_1$ and $\mathbf{u}_3$ take a new position $e^{0.1\theta}\mathbf{u}_1$ and $e^{0.1\theta}\mathbf{u}_3$, respectively, and the mapping (5.25) becomes

$$\mathbf{x}(\xi,\eta) = (1-\xi-\eta)e^{0.1\theta}\mathbf{u}_1 + \xi e^{0.1\theta}\mathbf{u}_2 + \eta \mathbf{u}_3 \quad \text{for } (\xi,\eta) \in \mathcal{T}. \tag{5.26}$$

The Jacobian of transformation (5.26) is now a function of (ξ,η,θ). The determinant of the Jacobian, which appears as a multiplicative coefficient in Maxwell's equations, is random, and this makes Maxwell's equation a stochastic PDE, which can be solved with the techniques described above (collocation or Galerkin stochastic method). The advantage of the procedure lies in the fact that one need only generate a single mesh, the randomness being modeled by the Jacobian of the transformation.

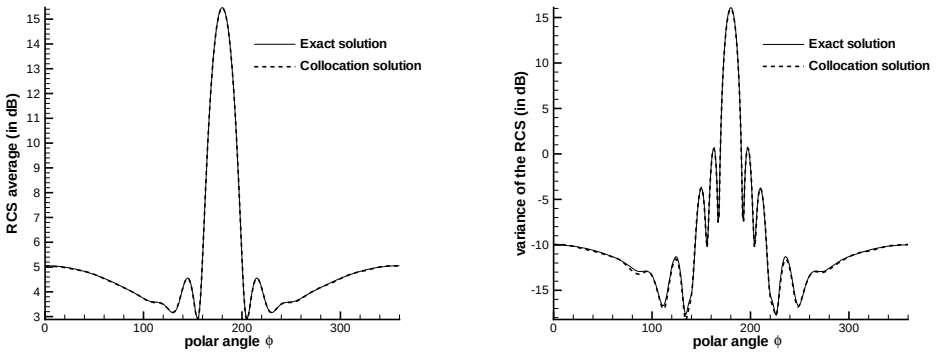


FIG. 5.9. *Comparison of the stochastic collocation solution with the exact solution. Left: Average of the RCS for the uncertainty in shape. Right: Variance of the RCS for uncertainty in shape.*

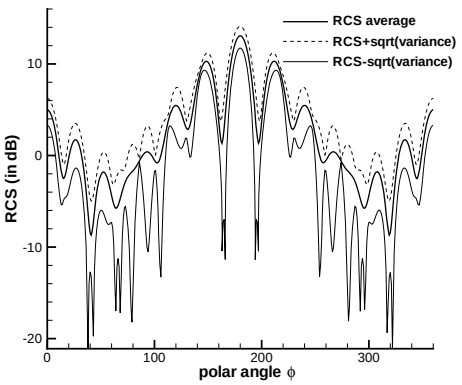


FIG. 5.10. *RCS for material cylinder with uncertain radius. Results are shown with the mean RCS as well as* ± 1 *standard deviation.*

. We test the approach by considering a PEC cylinder with an uncertain radius, the randomness being described by (5.26). For the deterministic problem, an exact RCS is available [2]. Therefore, the coefficients in (5.3) can be computed exactly and an exact stochastic solution for the RCS is available for the random radius problem. This enables us to compare the statistics of the numerical stochastic collocation solution with the exact one. This is what is done in Figure 5.9, which represents the average and the variance of the RCS (the mesh used for the numerical results is the same as in the previous section). We can see that the two solutions match very well.

. For this last example, the exact radius of the cylindrical boundary of a cylinder with a permittivity ϵ_1 is uncertain and is assumed to be described by (5.26). We also assume that outside the object, the relative permittivity is equal to one. Figure 5.10 shows the average of the RCS for this problem and the possible variations around its average value.

6. Concluding remarks. In this paper we discussed the use of probabilistic methods, rather than more traditional statistical methods such as Monte Carlo, to

enable the computation of quantitative measures for the impact of uncertainties on computational results, exemplified by the solution of the time-domain Maxwell's equations. These uncertainties can be due to a lack of knowledge, or lack of control over variations, or can simply be used as a model to compute the mean and variances over an ensemble of data, e.g., to compute the RCS of a fleet of aircraft, all of the same type but with minor differences.

The approach, based on chaos expansions, has the following significant advantages:

- It is simple, systematic, and accurate, and can be used in connection with existing validated codes with only minimal changes.
- The representation of probability distributions, which are most often smooth, by using high-order polynomials can be expected to be close to optimal and leads to a need for very few terms, i.e., few samples, without sacrificing the accuracy.
- It enables the computation of an ensample of solutions in space and time with an associated probability measure. It can handle large degrees of uncertainty simply by increasing the length of the expansion.
- It is flexible enough to handle a large variety of different types of uncertainty, e.g., materials, initial conditions, and shapes.

The computational results validate these claims and show the potential of computing high-order moments, in particular variances/sensitivities, thus enabling the computation of numerical error bars.

Throughout this paper, the dimension of the random space d is set to 1, and for the stochastic collocation method, the computational cost is proportional to P, where P is the number of collocation points. For more general problems with random dimensions $d > 1$, the computational cost of the collocation method would be proportional to P^d (for Galerkin polynomial chaos, it could be P^{2d}, where, in that case, P denotes the number of modes). This means that for problems where the correlation length is very small compared to the size of the problem, this method becomes more expensive, as a significant number of independent random variables is needed. For such cases requiring high-dimensional random spaces, the stochastic collocation methods discussed in [17] hold significant promise, as one can choose to limit the attention to high-dimensional integration formulas, thus dramatically reducing the number of samples even for high-dimensional randomness. As we have discussed in section 4.4, however, there is significant potential for reducing the computational cost by developing a theory that offers insight into how uncertainty can propagate through dynamical systems and what regularity one can expect for the underlying probability densities.

REFERENCES

[1] S. Abarbanel, D. Gottlieb, and J. S. Hesthaven, *Long time behavior of the perfectly matched layer equations in computational electromagnetics*, J. Sci. Comput., 17 (2002), pp. 405–422.

[2] J. J. Bowman, T. B. A. Senior, and P. L. Ushlenghi, eds., *Electromagnetic and Acoustic Scattering by Simple Shapes*, North-Holland, Amsterdam, 1969.

[3] R. H. Cameron and W. T. Martin, *The orthogonal development of nonlinear functionals in series of Fourier-Hermite functionals*, Ann. of Math. (2), 48 (1947), pp. 385–392.

[4] C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang, *Spectral Methods in Fluid Mechanics*, Springer-Verlag, New York, 1988.

[5] Q. Y. Chen, D. Gottlieb, and J. S. Hesthaven, *Uncertainty analysis for steady-state inviscid Burgers equation*, J. Comput. Phys., 204 (2005), pp. 378–398.

[6] D. Funaro, *Polynomial Approximations of Differential Equations*, Lecture Notes in Phys. New Ser. m Monogr. 8, Springer-Verlag, Berlin, 1992.
[7] R. G. Ghanem and P. Spanos, *Stochastic Finite Elements: A Spectral Approach*, Springer-Verlag, New York, 1991.
[8] D. Gottlieb and S. A. Orszag, *Numerical Analysis of Spectral Methods: Theory and Applications*, CBMS-NSF Regional Conf. Ser. in Appl. Math. 26, SIAM, Philadelphia, 1977.
[9] J. S. Hesthaven and T. Warburton, *High-order nodal methods on unstructured grids.* I. *Time-domain solution of Maxwell's equations*, J. Comput. Phys., 181 (2002), pp. 1–34.
[10] J. S. Hesthaven and T. Warburton, *High-order accurate methods for time-domain electromagnetics*, CMES–Computer Modeling in Engineering and Sciences, 5 (2004), pp. 395–408.
[11] J. S. Hesthaven and T. Warburton, *High-order nodal discontinuous Galerkin methods for the Maxwell eigenvalue problem*, Philos. Trans. R. Soc. Lond. Ser. A Math. Phys. Eng. Sci., 362 (2004), pp. 493–524.
[12] J. S. Hesthaven and T. Warburton, *Discontinuous Galerkin methods for the time-domain Maxwell's equations: An introduction*, ACES Newsletter, 19 (2004), pp. 10–29.
[13] M. E. J. Newman and G. T. Barkema, *Monte Carlo Methods in Statistical Physics*, Oxford University Press, Oxford, UK, 1999.
[14] W. Shoutens, *Stochastic Processes and Orthogonal Polynomials*, Springer-Verlag, New York, 2000.
[15] A. Taflove, *Computational Electrodynamics—The Finite-Difference Time-Domain Method*, Artech House, Boston, 1995.
[16] N. Wiener, *The homogeneous chaos,* Amer. J. Math., 60 (1938), pp. 897–936.
[17] D. Xiu and J. S. Hesthaven, *High-order collocation methods for differential equations with random inputs*, SIAM J. Sci. Comput., 27 (2005), pp. 1118–1139.
[18] D. Xiu and G. E. Karniadakis, *Modeling uncertainty in flow simulations via polynomial chaos*, J. Comput. Phys., 187 (2003), pp. 137–167.

SIAM J. SCI. COMPUT.
Vol. 28, No. 2, pp. 776–803
© 2006 Society for Industrial and Applied Mathematics

PRECONDITIONING MARKOV CHAIN MONTE CARLO SIMULATIONS USING COARSE-SCALE MODELS*

Y. EFENDIEV†, T. HOU‡, AND W. LUO‡

Abstract. We study the preconditioning of Markov chain Monte Carlo (MCMC) methods using coarse-scale models with applications to subsurface characterization. The purpose of preconditioning is to reduce the fine-scale computational cost and increase the acceptance rate in the MCMC sampling. This goal is achieved by generating Markov chains based on two-stage computations. In the first stage, a new proposal is first tested by the coarse-scale model based on multiscale finite volume methods. The full fine-scale computation will be conducted only if the proposal passes the coarse-scale screening. For more efficient simulations, an approximation of the full fine-scale computation using precomputed multiscale basis functions can also be used. Comparing with the regular MCMC method, the preconditioned MCMC method generates a modified Markov chain by incorporating the coarse-scale information of the problem. The conditions under which the modified Markov chain will converge to the correct posterior distribution are stated in the paper. The validity of these assumptions for our application and the conditions which would guarantee a high acceptance rate are also discussed. We would like to note that coarse-scale models used in the simulations need to be inexpensive but not necessarily very accurate, as our analysis and numerical simulations demonstrate. We present numerical examples for sampling permeability fields using two-point geostatistics. The Karhunen–Loève expansion is used to represent the realizations of the permeability field conditioned to the dynamic data, such as production data, as well as some static data. Our numerical examples show that the acceptance rate can be increased by more than 10 times if MCMC simulations are preconditioned using coarse-scale models.

Key words. preconditioning, multiscale, Markov chain Monte Carlo, porous media

AMS subject classifications. 65N99, 62P30, 62F15, 65C05, 65C40

DOI. 10.1137/050628568

1. Introduction.

Uncertainties on the detailed description of reservoir lithofacies, porosity, and permeability are major contributors to uncertainty in reservoir performance forecasting. Reducing this uncertainty can be achieved by integrating additional data in subsurface modeling. With the increasing interest in accurate prediction of subsurface properties, subsurface characterization based on dynamic data, such as production data, becomes more important.

To predict future reservoir performance, the reservoir properties, such as porosity and permeability, need to be conditioned to dynamic data, such as production data. In general it is difficult to calculate this probability distribution, because the process of predicting flow and transport in petroleum reservoirs is nonlinear. Instead, this probability distribution is estimated from the outcomes of flow predictions for a large number of realizations of the reservoir. It is essential that the permeability (and porosity) realizations adequately reflect the uncertainty in the reservoir properties; i.e., the probability distribution is sampled correctly. This problem is challenging because the permeability field is a function defined on a large number of grid blocks.

*Received by the editors April 5, 2005; accepted for publication (in revised form) January 13, 2006; published electronically May 26, 2006.

http://www.siam.org/journals/sisc/28-2/62856.html

†Department of Mathematics, Texas A&M University, College Station, TX 77843-3368 (efendiev@math.tamu.edu). The research of this author was partially supported by DOE grant DE-FG02-05ER25669.

‡Applied Mathematics, Caltech, Pasadena, CA 91125 (hou@acm.caltech.edu, wuan@acm.caltech.edu). The research of the second author was partially supported by NSF ITR grant ACI-0204932 and NSF FRG grant DMS-0353838.

The Markov chain Monte Carlo (MCMC) method and its modifications have been used previously to sample the posterior distribution. In this paper, we design a two-stage MCMC method which employs coarse-scale models based on multiscale finite volume methods.

The direct MCMC simulations are generally very CPU demanding because each proposal requires solving a forward coupled nonlinear PDE over a large time interval. The forward fine-scale problem is usually formulated on a large number of grid blocks, which makes it prohibitively expensive to perform a sufficient number of MCMC simulations. There have been a few attempts to propose MCMC methods with high acceptance rates, for example, the randomized maximum likelihood method [19, 20]. This approach uses unconditional realizations of the production and permeability data and solves a deterministic gradient-based inverse problem. The solution of this minimization problem is taken as a proposal, and is accepted with probability 1, because the rigorous acceptance probability is very difficult to estimate. In addition to the need of solving a gradient-based inverse problem, this method does not properly sample the posterior distribution. Thus, developing efficient rigorous MCMC calculations with high acceptance rates remains a challenging problem.

In this paper, we show that using inexpensive coarse-scale computations one can increase the acceptance rate of MCMC calculations. Here the acceptance rate refers to the ratio between the number of accepted permeability samples and the number of times of solving the fine-scale nonlinear PDE system. The method consists of two stages. At the first stage, using coarse-scale runs we determine whether or not to run the fine-scale simulations. If the proposal is accepted at the first-stage, then a fine-scale simulation is performed at the second stage to determine the acceptance probability of the proposal. The first stage of the MCMC method modifies the proposal distribution. We show that the modified Markov chain satisfies the detailed balance condition for the correct distribution. Moreover, we point out that the chain is ergodic and converges to the correct posterior distribution under some technical assumptions. The validity of the assumptions for our application is discussed in the paper. We would like to note that two-stage MCMC algorithms have been used previously (e.g., [2, 15, 21, 10]) in different situations.

In this paper, we use a coarse-scale model based on multiscale finite volume methods. Note that it is essential that these coarse-scale models are inexpensive but not necessarily very accurate. The main idea of multiscale finite volume methods is to construct multiscale basis functions that contain the small scale information. Constructing these basis functions based on the single-phase flow is equivalent to single-phase flow upscaling, provided the transport equation is solved on a coarse grid. This method is inexpensive, since the basis functions are constructed only once, and the transport equation is solved on the coarse grid. The use of multiscale finite volume methods has another advantage: it can be further used as an accurate approximation for the production data if the transport equation is solved on the fine grid. For this purpose, one needs to compute the fine-scale velocity fields from the precomputed multiscale basis functions and solve the saturation on the fine grid. This provides an accurate approximation for the production data [12, 13, 1]. Since one can reuse the basis functions from the first stage, the resulting method is very efficient. We would like to note that upscaled models are used in MCMC simulations in previous findings. In an interesting work [9], the authors employ error models between coarse- and fine-scale simulations to quantify the uncertainty.

Numerical results for permeability fields generated using two-point geostatistics are presented in the paper. Using the Karhunen–Loève expansion, we can represent

the high dimensional permeability field by a small number of parameters. Furthermore, static data (the values of permeability field at some sparse locations) can be easily incorporated into the Karhunen–Loève expansion to further reduce the dimension of the parameter space. Numerical results are presented for both single-phase and two-phase flows for side-to-side and corner-to-corner flows. In all the simulations, we observe more than 10 times an increase in the acceptance rate. In other words, the preconditioned MCMC method can accept the same number of samples with much fewer fine-scale runs.

The paper is organized in the following way. In the next section, we briefly describe the model equations and their upscaling. Section 3 is devoted to the analysis of the preconditioned MCMC method and its relevance to our particular application. Numerical results are presented in section 4.

2. Fine and coarse models. We consider two-phase flows in a domain Ω under the assumption that the displacement is dominated by viscous effects. We neglect the effects of gravity, compressibility, and capillary pressure. The two phases are referred to as water (aqueous phase) and oil (nonaqueous phase liquid), designated by subscripts w and o, respectively. We write Darcy's law, with all quantities dimensionless, for each phase as follows:

$$\boldsymbol{v}_j = -\frac{k_{rj}(S)}{\mu_j}\boldsymbol{k}\cdot\nabla p, \tag{2.1}$$

where $\boldsymbol{v}_j$, $j = w, o$, is the phase velocity, $\boldsymbol{k}$ is the permeability tensor, k_{rj} is the relative permeability of the phase j, S is the water saturation (volume fraction), and p is the pressure. In this work, a single set of relative permeability curves is used and $\boldsymbol{k}$ is taken to be a diagonal tensor. Combining Darcy's law with a statement of conservation of mass allows us to express the governing equations in terms of the so-called pressure and saturation equations:

$$\nabla\cdot(\lambda(S)\boldsymbol{k}\nabla p) = q, \tag{2.2}$$

$$\frac{\partial S}{\partial t} + \boldsymbol{v}\cdot\nabla f(S) = -q_w, \tag{2.3}$$

where $\lambda(S)$ is the total mobility, q and q_w are the source terms, $\boldsymbol{v}$ is the total velocity, and $f(S)$ is the flux function, which are, respectively, given by

$$\lambda(S) = \frac{k_{rw}(S)}{\mu_w} + \frac{k_{ro}(S)}{\mu_o}, \tag{2.4}$$

$$\boldsymbol{v} = \boldsymbol{v}_w + \boldsymbol{v}_o = -\lambda(S)\boldsymbol{k}\,\nabla p, \tag{2.5}$$

$$f(S) = \frac{k_{rw}(S)/\mu_w}{k_{rw}(S)/\mu_w + k_{ro}(S)/\mu_o}. \tag{2.6}$$

The above description is referred to as the fine model of the two-phase flow problem. For the single-phase flow, we have $\lambda(S) = 1$ and $f(S) = S$. Throughout, the porosity is assumed to be constant and the permeability is assumed to have the form $\boldsymbol{k}(\boldsymbol{x}) = k(\boldsymbol{x})\boldsymbol{I}$, where $\boldsymbol{I}$ is a unit matrix and $k(\boldsymbol{x})$ is a scalar function.

The proposed coarse-scale model consists of upscaling the pressure equation (2.2) to obtain the velocity field on the coarse grid and then using it in (2.3) to resolve the

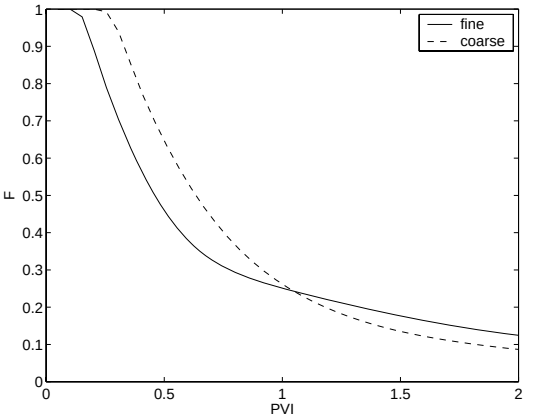


FIG. 2.1. *Typical fine- and coarse-scale fractional flows.*

saturation on the coarse grid. The pressure equation is upscaled using the multiscale finite volume method. The details of the method are presented in Appendix A. Using the multiscale finite volume method, we obtain the coarse-scale velocity field, which is used in solving the saturation equation on the coarse grid. Since no subgrid modeling is performed for the saturation equation, this upscaling procedure introduces errors. In Figure 2.1, we present a comparison of the typical fractional flows computed by fine- and coarse-scale models. The fractional flows are plotted against the dimensionless time "pore volume injected." The pore volume injected (PVI) at time T is defined as $\frac{1}{V_p}\int_0^T q_t(\tau)d\tau$, where q_t is the combined flow rates of water and oil at the production edge, and V_p is the total pore volume of the system. PVI provides the dimensionless time for the flow displacement. The fractional flow $F(t)$ (denoted simply by F thereafter) is the fraction of oil in the produced fluid and is defined as $F = q_o/q_t$, where $q_t = q_o + q_w$, with q_o and q_w denoting the flow rates of oil and water at the production edge of the model. More specifically,

$$F(t) = 1 - \frac{\int_{\partial\Omega^{out}} f(S)\, v_n\, dl}{\int_{\partial\Omega^{out}} v_n\, dl},$$

where $\partial\Omega^{out}$ is the outflow boundary and $v_n = \boldsymbol{v}\cdot\boldsymbol{n}$ is the normal velocity on the boundary. In future analysis, the notations q_o, q_w, or q_t will not be used, and q will be reserved for the proposal distributions. The proposed coarse-scale model is somewhat similar to the single-phase flow upscaling [4]. One can improve the accuracy of the above coarse model by solving the transport equation on the fine grid using the fine-scale velocity field which can be computed employing precomputed multiscale basis functions. This makes solving the coarse model more expensive because the transport update is performed on the fine grid with smaller time steps. However, it can provide an efficient numerical solver for the second stage of the preconditioned MCMC method, as we will discuss later.

3. Preconditioning MCMC simulations using coarse-scale models.

3.1. Problem setting. The problem under consideration consists of sampling the permeability field given fractional flow measurements. Typically, the permeability field is known at some sparse locations. This information should be incorporated into

the prior models (distributions) of the permeability. Since the fractional flow is an integrated response, the map from the permeability field to the fractional flow is not one-to-one. Hence this problem is ill-posed in the sense that there exist many different permeability realizations for the given production data.

From the probabilistic point of view, this problem can be regarded as sampling the permeability field conditioning on the fractional flow data with measurement errors. Consequently, our goal is to sample from the conditional distribution $P(k|F)$, where k is the fine-scale permeability field and F is the fractional flow curve measured from the production data. Using the Bayes theorem we can write

$$P(k|F) \propto P(F|k)P(k). \tag{3.1}$$

In the above formula, $P(k)$ is the prior distribution of the permeability field, which is assumed to be log-normal. The prior distribution $P(k)$ will also incorporate the additional information of the permeability field at the sparse locations. The likelihood function $P(F|k)$ denotes the conditional probability that the outcome of the measurement is F when the true permeability is k.

In practice, the measured fractional flow F contains measurement errors. Denote the fractional flow for a given k as F_k. F_k can be computed by solving the model equation (2.1)–(2.3) on the fine grid. The computed F_k will contain a modeling error as well as a numerical error. In this paper, we assume that the combined errors from the measurement, modeling, and numerics satisfy a Gaussian distribution. That is, the likelihood function $P(F|k)$ takes the form

$$P(F|k) \propto \exp\left(-\frac{\|F - F_k\|^2}{\sigma_f^2}\right), \tag{3.2}$$

where F is the observed fractional flow, F_k is the fractional flow computed by solving the model equations (2.1)–(2.3) on the fine grid for a given k, and σ_f is the precision associated with the measurement F and the numerical solution F_k. Since both F and F_k are functions of t, $\|F - F_k\|^2$ denotes the L_2 norm

$$\|F - F_k\|^2 = \int_0^T [F(t) - F_k(t)]^2 \, dt.$$

It is worth noting that the method discussed in this paper does not depend on the specific form of the error functions. A more general error model can be used in the simulations. We would like to emphasize that different permeability fields may produce the same fractional flow curve. Thus, the likelihood distribution $P(F|k)$ is a multimodal function of k with multiple local maxima.

Denote the posterior distribution as

$$\pi(k) = P(k|F) \propto \exp\left(-\frac{\|F - F_k\|^2}{\sigma_f^2}\right) P(k). \tag{3.3}$$

Sampling from the distribution $\pi(k)$ can be accomplished by using the MCMC method. The main idea of the MCMC method is to generate a Markov chain with $\pi(k)$ as its stationary distribution. A key step to this approach is to construct the desired transition kernel for the Markov chain. In this paper, we use the Metropolis-Hasting algorithm. Suppose $q(y|x)$ is a general transitional probability distribution, which is

easy to sample and has an explicit form. The Metropolis-Hasting MCMC algorithm (see, e.g., [22]) consists of the following steps.

ALGORITHM (METROPOLIS-HASTING MCMC [22]).

- Step 1. At state k_n generate k from $q(k|k_n)$.
- Step 2. Accept k as a sample with probability

$$p(k_n, k) = \min\left(1, \frac{q(k_n|k)\pi(k)}{q(k|k_n)\pi(k_n)}\right); \tag{3.4}$$

  i.e., take $k_{n+1} = k$ with probability $p(k_n, k)$ and $k_{n+1} = k_n$ with probability $1 - p(k_n, k)$.

Starting with an arbitrary initial permeability sample k_0, the MCMC algorithm generates a Markov chain $\{k_n\}$. At each iteration, the probability of moving from state k_n to a next state k is $q(k|k_n)p(k_n, k)$, so the transition kernel for the Markov chain $\{k_n\}$ is

$$K(k_n, k) = p(k_n, k)q(k|k_n) + \left(1 - \int p(k_n, k)q(k|k_n)dk\right)\delta_{k_n}(k).$$

Using the explicit formula of the transition kernel, it is not difficult to prove that the target distribution $\pi(k)$ is indeed the stationary distribution of the Markov chain $\{k_n\}$. As a result, we can take k_n as samples of the distribution $\pi(k)$ after the chain reaches the steady state.

3.2. The preconditioned MCMC method. In the above Metropolis-Hasting MCMC algorithm, the major computational cost is to compute F_k in the target distribution $\pi(k)$, which involves solving the coupled nonlinear PDE system (2.1)–(2.3) on the fine grid. Generally, the MCMC method requires thousands of iterations before it converges to the steady state. To quantify the uncertainty of the permeability field accurately, one also needs to generate a large number of different samples. However, the acceptance rate of the direct MCMC method is very low, due to the large dimensionality of the permeability field. The algorithm needs to test many proposals to accept only a few permeability samples. Most of the CPU time is spent on simulating the rejected samples. That makes the direct (full) MCMC simulations prohibitively expensive in practice.

One way to improve the direct MCMC method is to increase its acceptance rate by modifying the proposal distribution $q(k|k_n)$. In this paper, we propose an algorithm in which the proposal distribution $q(k|k_n)$ is adapted to the target distribution using the coarse-scale model. Instead of testing each proposal by fine-scale computations directly, the algorithm first tests the proposal by the coarse-scale model. This is achieved by comparing the fractional flow curves on the coarse grid first. If the proposal is accepted by the coarse-scale test, then a full fine-scale computation will be conducted and the proposal will be further tested as in the direct MCMC method. Otherwise, the proposal will be rejected by the coarse-scale test and a new proposal will be generated from $q(k|k_n)$. The coarse-scale test filters the unacceptable proposals and avoids the expensive fine-scale tests for those proposals. The filtering process essentially modifies the proposal distribution $q(k|k_n)$ by incorporating the coarse-scale information of the problem. That is why the modified method is called a

Recall that the fine-scale target distribution is given by (3.3). We approximate

the distribution $\pi(k)$ on the coarse scale by

$$\pi^*(k) \propto \exp\left(-\frac{\|F - F_k^*\|^2}{\sigma_c^2}\right) P(k), \tag{3.5}$$

where F_k^* is the fractional flow computed by solving the coarse-scale model of (2.1)–(2.3) for the given k, and σ_c is the precision associated with the coarse-scale model. The parameter σ_c plays an important role in improving the acceptance rate of the preconditioned MCMC method. The optimal value of σ_c depends on the correlation between $\|F - F_k\|$ and $\|F - F_k^*\|$, which can be estimated by numerical simulations. (cf. Figure 3.1 and later discussion). Using the coarse-scale distribution $\pi^*(k)$ as a filter, the preconditioned MCMC method can be described as follows.

Algorithm (preconditioned MCMC).

- Step 1. At k_n, generate a trial proposal k' from distribution $q(k'|k_n)$.
- Step 2. Take the real proposal as

$$k = \begin{cases} k' & \text{with probability } g(k_n, k'), \\ k_n & \text{with probability } 1 - g(k_n, k'), \end{cases}$$

  where

$$g(k_n, k') = \min\left(1, \frac{q(k_n|k')\pi^*(k')}{q(k'|k_n)\pi^*(k_n)}\right). \tag{3.6}$$

  Therefore, the final proposal k is generated from the effective instrumental distribution

$$Q(k|k_n) = g(k_n, k)q(k|k_n) + \left(1 - \int g(k_n, k)q(k|k_n)dk\right)\delta_{k_n}(k). \tag{3.7}$$

- Step 3. Accept k as a sample with probability

$$\rho(k_n, k) = \min\left(1, \frac{Q(k_n|k)\pi(k)}{Q(k|k_n)\pi(k_n)}\right); \tag{3.8}$$

  i.e., take $k_{n+1} = k$ with probability $\rho(k_n, k)$ and $k_{n+1} = k_n$ with probability $1 - \rho(k_n, k)$.

In the above algorithm, if the trial proposal k' is rejected by the coarse-scale test (Step 2), k_n will be passed to the fine-scale test as the proposal. Since $\rho(k_n, k_n) \equiv 1$, no further (fine-scale) computation is needed. Thus, the expensive fine-scale computations can be avoided for those proposals which are unlikely to be accepted. In comparison, the regular MCMC method requires a fine-scale simulation for every proposal k, even though most of the proposals will be rejected at the end.

It is worth noting that there is no need to compute $Q(k|k_n)$ and $Q(k_n|k)$ in (3.8) by the integral formula (3.7). The acceptance probability (3.8) can be simplified as

$$\rho(k_n, k) = \min\left(1, \frac{\pi(k)\pi^*(k_n)}{\pi(k_n)\pi^*(k)}\right). \tag{3.9}$$

In fact, (3.9) is obviously true for $k = k_n$ since $\rho(k_n, k_n) \equiv 1$. For $k \neq k_n$,

$$Q(k_n|k) = g(k, k_n)q(k_n|k) = \frac{1}{\pi^*(k)} \min\Big(q(k_n|k)\pi^*(k),\ q(k|k_n)\pi^*(k_n)\Big)$$

$$= \frac{q(k|k_n)\pi^*(k_n)}{\pi^*(k)} g(k_n, k) = \frac{\pi^*(k_n)}{\pi^*(k)} Q(k|k_n).$$

Substituting the above formula into (3.8), we immediately get (3.9).

Since the computation of the coarse-scale solution is very cheap, Step 2 of the preconditioned MCMC method can be implemented very quickly to decide whether or not to run the fine-scale simulation. The second step of the algorithm serves as a filter that avoids unnecessary fine-scale runs for the rejected samples. It is possible that the coarse-scale test may reject an individual sample which will otherwise have a (small) probability to be accepted in the fine-scale test. However, that does not play a crucial role, since we are interested only in the statistical property of the samples. We will show later that the preconditioned MCMC algorithm converges under some mild assumptions.

We would like to note that the Gaussian error model for the coarse-scale distribution $\pi^*(k)$ is not very accurate. We use it only in the filtering stage to decide whether or not to run the fine-scale simulations. The choice of the coarse-scale precision parameter σ_c is important for increasing the acceptance rate. If σ_c is too large, then too many proposals can pass the coarse-scale tests and the filtering stage will become less effective. If σ_c is too small, then eligible proposals may be incorrectly filtered out, which will result in biased sampling. Our numerical results show that the acceptance rate is optimal when σ_c is of the same order as σ_f. The optimal value of σ_c can be estimated based on the correlation between $\|F - F_k\|$ and $\|F - F_k^*\|$ (cf. Figure 3.1).

Based on the Gaussian precision models (3.3) and (3.5), the acceptance probability (3.9) has the form

$$\rho(k_n, k) = \min\left(1, \frac{\pi(k)\pi^*(k_n)}{\pi(k_n)\pi^*(k)}\right) = \min\left(1, \frac{\exp\left(-\frac{E_k - E_{k_n}}{\sigma_f^2}\right)}{\exp\left(-\frac{E_k^* - E_{k_n}^*}{\sigma_c^2}\right)}\right), \tag{3.10}$$

where

$$E_k = \|F - F_k\|^2, \qquad E_k^* = \|F - F_k^*\|^2.$$

If E_k^* is strongly correlated with E_k, then the acceptance probability (3.10) could be close to 1 for certain choice of σ_c. Hence a high acceptance rate can be achieved at Step 3 of the preconditioned MCMC method. To demonstrate that E_k^* is indeed strongly correlated with E_k, we compute E_k and E_k^* for many different permeability samples k (see the second example of section 4, Figure 4.7, for details of the permeability field) and plot E_k against E_k^* in Figure 3.1. We find that the correlation coefficient between E_k^* and E_k is approximately 0.9. If the correlation between E_k and E_k^* is strong, we can write

$$E_k \simeq \alpha E_k^* + \beta.$$

Substituting this into (3.10) and choosing $\sigma_c^2 = \sigma_f^2/\alpha$, we can obtain the acceptance rate close to 1 in Step 3. In practice, however, one does not know a priori the correlation constant α. The approximate value of α can be estimated by a priori numerical simulations where E_k and E_k^* are computed for a number of permeability samples.

The preconditioned MCMC method uses the coarse-scale distribution (3.5) with the reference fractional flow being the observed fine-scale fractional flow. One can also use a different reference fractional flow curve in Step 2 of the preconditioned MCMC method to improve the acceptance rate. In our numerical simulations (not presented here), we have used the coarse-scale fractional flow corresponding to the

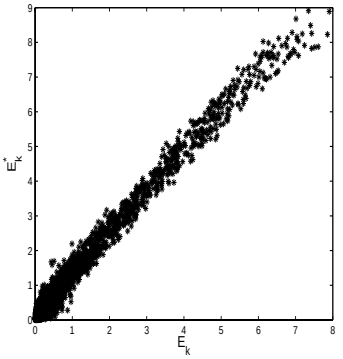


FIG. 3.1. *Cross-plot between* $E_k = \|F - F_k\|^2$ *and* $E_k^* = \|F - F_k^*\|^2$.

observed fractional flow as the reference fractional flow in the preconditioned MCMC simulations. We have observed similar numerical results. Since the coarse-scale fractional flow corresponding to the observed fractional flow is generally not known, we do not present these numerical results here. However, we note that one can possibly improve the preconditioning by a careful choice of the reference fractional flow.

The preconditioned MCMC method employs multiscale finite volume methods in the preconditioning step. If a proposal is accepted by the coarse-scale test (Step 2), one can use the precomputed multiscale basis functions to reconstruct the velocity field on the fine scale. Then the transport equation can be solved on the fine grid coupled with the coarse-grid pressure equation [6, 12, 13, 1]. This approach provides an accurate approximation to the production data on the fine grid and can be used to replace the fine-scale computation in the second stage (Step 3). In this procedure, the basis functions are not updated in time, or updated only in a few coarse blocks. Thus the fine-scale computation in the second stage of the preconditioned MCMC method (Step 3) can also be implemented quickly. Since the basis functions from the first-stage are reused for the fine-scale computation, this combined multiscale approach can be very efficient for our sampling problem.

3.3. Analysis of the preconditioned MCMC method. Next we will analyze the preconditioned MCMC method in more details. Denote

$$\begin{aligned} &\mathcal{E} = \big\{k;\ \pi(k) > 0\big\}, \\ &\mathcal{E}^* = \big\{k;\ \pi^*(k) > 0\big\}, \\ &\mathcal{D} = \big\{k;\ q(k|k_n) > 0 \text{ for some } k_n \in \mathcal{E}\big\}. \end{aligned} \tag{3.11}$$

The set $\mathcal{E}$ is the support of the posterior (target) distribution $\pi(k)$. $\mathcal{E}$ contains all the permeability field k which has a positive probability of being accepted as a sample. Similarly, $\mathcal{E}^*$ is the support of the coarse-scale distribution $\pi^*(k)$, which contains all the k acceptable by the coarse-scale test. $\mathcal{D}$ is the set of all the proposals which can be generated by the instrumental distribution $q(k|k_n)$. For the preconditioned MCMC method to work properly, the conditions $\mathcal{E} \subseteq \mathcal{D}$ and $\mathcal{E} \subseteq \mathcal{E}^*$ must hold (up to a zero measure set) simultaneously. If one of these conditions is not true, say, $\mathcal{E} \not\subseteq \mathcal{E}^*$, then

there will exist a subset $A \subset (\mathcal{E} \setminus \mathcal{E}^*)$ such that

$$\pi(A) = \int_A \pi(k)dk > 0 \qquad \text{and} \qquad \pi^*(A) = \int_A \pi^*(k)dk = 0,$$

which means no element of A can pass the coarse-scale test and A will never be visited by the Markov chain $\{k_n\}$. Thus, $\pi(k)$ cannot be sampled properly.

For most practical proposals $q(k|k_n)$, such as the random walk samplers and independent samplers, the conditions $\mathcal{E}, \mathcal{E}^* \subset \mathcal{D}$ can be naturally satisfied. By choosing the parameter σ_c in $\pi^*(k)$ properly, the condition $\mathcal{E} \subset \mathcal{E}^*$ can also be satisfied (see the discussion below). As a result, we have $\mathcal{E} \subset \mathcal{E}^* \subset \mathcal{D}$. In this case, $\mathcal{E}^*$ is identical to the support of the effective proposal $Q(k|k_n)$:

$$\mathcal{E}^* = \big\{k;\ Q(k|k_n) > 0 \text{ for some } k_n \in \mathcal{E}\big\}.$$

Due to the high dimension of the permeability field k, the support $\mathcal{E}$ of the target distribution $\pi(k)$ is much smaller than the support $\mathcal{D}$ of the proposal $q(k|k_n)$ distribution. For all the proposals $k \in (\mathcal{D} \setminus \mathcal{E})$, they will never be accepted as samples in the MCMC method since $\pi(k) = 0$. In the preconditioned MCMC algorithm, the effective proposal distribution $Q(k|k_n)$ samples from a much smaller support $\mathcal{E}^*$, and hence avoids solving the fine-scale problems for all $k \in (\mathcal{D} \setminus \mathcal{E}^*)$. Suppose that we sample the posterior distribution $\pi(k)$ by both the regular MCMC method and preconditioned MCMC method. For each proposal k generated from $q(k|k_n)$, the regular MCMC method accepts it as a sample with probability $p(k_n, k)$ as defined by (3.4), while the preconditioned MCMC method accepts it with probability $g(k_n, k)\rho(k_n, k)$, where $g(k_n, k)$ is the acceptance probability (3.6) of the coarse-scale test and $\rho(k_n, k)$ is the acceptance probability (3.8) of the fine-scale test. When $g(k_n, k) < 1$ and $\rho(k_n, k) < 1$, which is true for most proposals k, it is easy to show that $g(k_n, k)\rho(k_n, k) = p(k_n, k)$. That is, the two methods accept k as an example with the same probability. In numerical experiments, both methods indeed accept approximately the same amount of proposals for a fixed number of iterations. However, the regular MCMC method needs to solve a fine-scale problem for each MCMC iteration, while the preconditioned MCMC method solves only the fine-scale problem when the proposal passes the coarse-scale test. For all the proposals $k \in (\mathcal{D} \setminus \mathcal{E}^*)$, they will be rejected directly by the coarse-scale criteria and do not require fine-scale computations. For each iteration, the preconditioned MCMC solves only the fine-scale problem r time in average, where

$$r = \int_{\mathcal{E}^*} g(k_n, k)q(k|k_n)dk < 1.$$

Note that $\int_{\mathcal{D}} q(k|k_n)dk = 1$ and $g(k_n, k) \le 1$. If $\mathcal{E}^*$ is close to $\mathcal{E}$, and hence much smaller than $\mathcal{D}$, then $r \ll 1$. Therefore, the preconditioned MCMC method requires much less fine-scale simulation while still accepting approximately the same amount of proposals. In other words, the preconditioned MCMC method can achieve a much higher acceptance rate for each fine-scale computation.

Next we will discuss the stability property of the preconditioned MCMC method. We shall show that the preconditioned MCMC method shares the same convergence property as the regular MCMC method. Denote by K the transition kernel of the Markov chain $\{k_n\}$ generated by the preconditioned MCMC method. Since its effec-

tive proposal is $Q(k|k_n)$ as defined by (3.7), we get

$$K(k_n, k) = \rho(k_n, k)Q(k|k_n) \qquad \text{for } k \neq k_n, \tag{3.12}$$

$$K(k_n, \{k_n\}) = 1 - \int_{k \neq k_n} \rho(k_n, k)Q(k|k_n)dk. \tag{3.13}$$

That is, the transition kernel $K(k_n, \cdot)$ is continuous when $k \neq k_n$ and has a positive probability for the event $\{k = k_n\}$.

As in the regular MCMC method, it is easy to show that $K(k_n, k)$ satisfies the detailed balance condition

$$\pi(k_n)K(k_n, k) = \pi(k)K(k, k_n) \tag{3.14}$$

for any $k, k_n \in \mathcal{E}$. In fact, the equality (3.14) is obviously true when $k = k_n$. If $k \neq k_n$, then from (3.12) we have

$$\pi(k_n)K(k_n, k) = \pi(k_n)\rho(k_n, k)Q(k|k_n) = \min\Big(Q(k|k_n)\pi(k_n),\ Q(k_n|k)\pi(k)\Big)$$

$$= \min\left(\frac{Q(k|k_n)\pi(k_n)}{Q(k_n|k)\pi(k)},\ 1\right) Q(k_n|k)\pi(k) = \rho(k, k_n)Q(k_n|k)\pi(k) = \pi(k)K(k, k_n).$$

So the detailed balance condition (3.14) is always satisfied. Using (3.14), we can easily show that $\pi(A) = \int K(k, A)dk$ for any $A \in \mathcal{B}(\mathcal{E})$, where $\mathcal{B}(\mathcal{E})$ denotes all the measurable subsets of $\mathcal{E}$. Thus, $\pi(k)$ is indeed the stationary distribution of the transition kernel $K(k_n, k)$.

In the regular MCMC method, the proposal $q(k|k_n)$ is usually chosen to satisfy

$$q(k|k_n) > 0 \text{ for any } (k_n, k) \in \mathcal{E} \times \mathcal{E}, \tag{3.15}$$

which guarantees that the resulting regular MCMC method is irreducible. A similar statement is true for the preconditioned MCMC method.

LEMMA 3.1. [illegible] $q(k|k_n)$ [illegible] (3.15) [illegible] $\mathcal{E} \subset \mathcal{E}^*$ [illegible] $\{k_n\}$ [illegible] π [illegible].

[illegible]. According to the definition of strong irreducibility, we need only to show that $K(k_n, A) > 0$ for all $k_n \in \mathcal{E}$ and any measurable set $A \subset \mathcal{E}$ with $\pi(A) > 0$. Note that

$$K(k_n, A) \geq \int_{A \setminus k_n} K(k_n, k)dk = \int_{A \setminus k_n} \rho(k_n, k)Q(k_n, k)dk$$

$$= \int_{A \setminus k_n} \rho(k_n, k)g(k_n, k)q(k|k_n)dk.$$

In the above inequality, the equal sign holds when $k_n \notin A$. Since $\pi(A) = \int_A \pi(k)dk > 0$, it follows that $m(A) = m(A \setminus k_n) > 0$, where $m(A)$ is the Lebesgue measure. Since $A \subset \mathcal{E}$ and $\mathcal{E} \subset \mathcal{E}^*$, both $\rho(k_n, k)$ and $g(k_n, k)$ are positive for $k \in A$. Combining the positivity assumption (3.15), we can easily conclude that $K(k_n, A) > 0$. □

Most practical proposal distributions, such as random walk samplers or independent samplers, satisfy the positivity condition (3.15). Thus condition (3.15) poses only a mild restriction in practice. As we will see later, the proposals used in our numerical experiment naturally satisfy the condition (3.15).

Based on the stability property of Markov chains [22, 18], the following convergence result is readily available.

THEOREM 3.2 (see [22]). [illegible] (3.15) [illegible] $\mathcal{E} \subset \mathcal{E}^*$ [illegible] $\{k_n\}$ [illegible] $h(k)$

$$\lim_{N\to\infty} \frac{1}{N} \sum_{n=1}^{N} h(k_n) = \int h(k)\pi(k)dk. \tag{3.16}$$

[illegible] $\{k_n\}$ [illegible] k_n [illegible] $\pi(k)$ [illegible]

$$\lim_{n\to\infty} \sup_{A\in\mathcal{B}(\mathcal{E})} \left|K^n(k_0, A) - \pi(A)\right| = 0 \tag{3.17}$$

[illegible] k_0

To get the convergence property (3.17), we need to show that the Markov chain $\{k_n\}$ generated by the preconditioned MCMC method is aperiodic. Recall that a simple sufficient condition for aperiodicity is that $K(k_n, \{k_n\}) > 0$ for some $k_n \in \mathcal{E}$. In other words, the event $\{k_{n+1} = k_n\}$ happens with a positive probability in the preconditioned MCMC method. From the definition (3.13), we have

$$K(k_n, \{k_n\}) = 1 - \int_{k\neq k_n} \rho(k_n, k)Q(k|k_n)dk = 1 - \int_{k\neq k_n} \rho(k_n, k)g(k_n, k)q(k|k_n)dk.$$

Consequently, $K(k_n, \{k_n\}) \equiv 0$ requires $g(k_n, k) = 1$ and $\rho(k_n, k) = 1$ for almost all $k \in \mathcal{D}$, which means that all the proposals generated by $q(k|k_n)$ are correct samples of distributions $\pi(k)$ and $\pi^*(k)$. This is obviously not true in practice. Thus, the practical preconditioned MCMC method is always aperiodic and converges to the target distribution $\pi(k)$ in the sense of (3.17).

Next we discuss the necessary condition $\mathcal{E} \subseteq \mathcal{E}^*$, which is essential to guarantee the convergence of the preconditioned MCMC method. Due to the Gaussian form of the posterior distribution, $\pi(k)$ and $\pi^*(k)$ do not have a compact support and the domain $\mathcal{E}$ (or $\mathcal{E}^*$) is the whole space spanned by all k. However, if the precision parameters σ_f and σ_c are relatively small, then $\pi(k)$ and $\pi^*(k)$ are very close to zero for most proposals. From the numerical point of view, the proposal k is very unlikely to be accepted if $\pi(k)$ or $\pi^*(k)$ is close to zero. Consequently, the support of the distributions should be interpreted as $\mathcal{E} = \{k; \pi(k) > \delta\}$ and $\mathcal{E}^* = \{k; \pi^*(k) > \delta\}$, where δ is a small positive number.

If $k \in \mathcal{E}$, then $\pi(k) > \delta$ and $\|F_k - F\|^2/\sigma_f^2$ is not very large. To make $k \in \mathcal{E}^*$, $\|F_k^* - F\|^2/\sigma_c^2$ should not be very large either. If $\|F_k^* - F\|^2$ is bounded by $\|F_k - F\|^2$ up to a multiplicative constant, then the condition $\mathcal{E} \subseteq \mathcal{E}^*$ can be satisfied by choosing the parameter σ_c properly. For most upscaled models, the coarse-scale quantity is indeed bounded by the corresponding fine-scale quantity. For example, the upscaled velocity $\boldsymbol{v}^*$ in the saturation equation is obtained by averaging the fine-scale velocity $\boldsymbol{v}$ over the coarse-grid blocks

$$\boldsymbol{v}^*(x) = \sum_i \left(\frac{1}{|\Omega_i|} \int_{\Omega_i} \boldsymbol{v}(y)dy\right) 1_{\Omega_i}(x),$$

where $\Omega_i \subset \Omega$ are the coarse blocks. It is easy to show that

$$\begin{aligned} \|\boldsymbol{v}^*\|^2_{L^2(\Omega)} &= \sum_i \frac{1}{|\Omega_i|} \left(\int_{\Omega_i} \boldsymbol{v}(y) dy \right)^2 \\ &\leq \sum_i \frac{1}{|\Omega_i|} \left(\int_{\Omega_i} 1(y)^2 dy \right) \left(\int_{\Omega_i} \boldsymbol{v}^2(y) dy \right) = \|\boldsymbol{v}\|^2_{L^2(\Omega)}. \end{aligned} \tag{3.18}$$

Thus, the coarse-scale velocity is bounded by the corresponding fine-scale velocity. We would like to remark that for some nonlinear averaging operators, one can also show that the coarse-scale quantities are bounded by the corresponding fine-scale quantities. One of the examples is the homogenization operator for linear elliptic equations.

In general, it is difficult to carry out such estimates for fractional flows. However, coarse-scale fractional flows can be interpreted as some type of average of the corresponding fine-scale fractional flows. Indeed, the fine-scale fractional flow curve can be regarded as the travel times along the characteristics of the particles that start at the inlet. The coarse-scale fractional flow, on the other hand, represents an average of these travel times over characteristics within the coarse domain. In general, the estimation similar to (3.18) does not hold for fractional flow curves, as our next counterexample shows. For simplicity, we present the counterexample for the single-phase flow in porous media with four layers. This example can be easily generalized. Denote by t_i, $i = 1, 2, 3, 4$, the breakthrough times for the layers. Consider two fine-scale (with four layers) permeability fields with breakthrough times $t_1 = T_1$, $t_2 = T_2$, $t_3 = T_1$, $t_4 = T_2$ and $t_1 = T_1$, $t_2 = T_1$, $t_3 = T_2$, $t_4 = T_2$, respectively. These two fine-scale permeability fields will give the same fractional flows, since the times of the flights are the same up to a permutation. Now we consider the upscaling of these two fine scale permeability fields to two-layered media. Upscaling is equivalent to averaging the breakthrough times. Consequently, the breakthrough times for the corresponding upscaled models are $t_1^* = 0.5(T_1 + T_2)$, $t_2^* = 0.5(T_1 + T_2)$ and $t_1^* = 0.5(T_1 + T_1) = T_1$, $t_2^* = 0.5(T_2 + T_2) = T_2$, respectively. Thus, the coarse-scale models give different fractional flows, even though the fractional flows are identical for the fine-scale models. However, this type of counterexample can be avoided in practice, because the near-well values of the permeability are known, and consequently, permutation of the layers can be avoided.

4. Numerical results. In this section we discuss the implementation details of the preconditioned MCMC method and present some representative numerical results. Suppose the permeability field $k(\boldsymbol{x})$, where $\boldsymbol{x} = (x, z)$, is defined on the unit square $\Omega = [0, 1]^2$. We assume that the permeability field $k(\boldsymbol{x})$ is a log-normal process and its covariance function is known. The observed data include the fractional flow curve F and the values of the permeability at sparse locations. We discretize the domain Ω by a rectangular mesh, and the permeability field k is represented by a matrix (thus k is a high dimensional vector). As for the boundary conditions, we have tested various boundary conditions and observed similar results for the preconditioned MCMC method. In the following numerical experiments we assume $p = 1$ and $S = 1$ on $x = 0$ and $p = 0$ on $x = 1$ and no flow conditions on the lateral boundaries $z = 0$ and $z = 1$. We call this type of boundary condition side-to-side. We have chosen this type of boundary condition because they provide large deviations between coarse- and fine-scale simulations for the permeability fields considered in the paper. The other

type of boundary condition is set by specifying $p = 1$, $S = 1$ along the $x = 0$ edge for $0.5 \le z \le 1$ and $p = 0$ along the $x = 1$ edge for $0 \le z \le 0.5$. On the rest of the boundaries, no-flow boundary conditions are assumed. We call this type of boundary condition corner-to-corner. We will consider both single-phase and two-phase flow displacements.

Using the Karhunen–Loève expansion [17, 23], the permeability field can be expanded in terms of an optimal L^2 basis. By truncating the expansion we can represent the permeability matrix by a small number of random parameters. To impose the hard constraints (the values of the permeability at prescribed locations), we will find a linear subspace of the random parameter space (a hyperplane) which yields the desired permeability fields satisfying the hard constraints.

We first briefly recall the basic idea of the Karhunen–Loève expansion. Denote $Y(\boldsymbol{x}, \omega) = \log[k(\boldsymbol{x}, \omega)]$, where the sample variable ω is included to remind us that k is a random field. Suppose $Y(\boldsymbol{x}, \omega)$ is a second order stochastic process; that is, $Y(\boldsymbol{x}, \omega) \in L^2(\Omega)$ with probability 1. Without loss of generality, we assume that $E[Y(\boldsymbol{x}, \omega)] = 0$. Given an arbitrary orthonormal basis $\{\phi_k\}$ in $L^2(\Omega)$, we can expand $Y(\boldsymbol{x}, \omega)$ in Fourier series

$$Y(\boldsymbol{x}, \omega) = \sum_{k=1}^{\infty} Y_k(\omega)\phi_k(\boldsymbol{x}),$$

where

$$Y_k(\omega) = \int_\Omega Y(\boldsymbol{x}, \omega)\, \phi_k(\boldsymbol{x}) d\boldsymbol{x}, \qquad k = 1, 2, \ldots,$$

are random variables with zero means. We are interested in the special L^2 basis $\{\phi_k\}$ which makes Y_k uncorrelated: $E(Y_i Y_j) = 0$ for all $i \ne j$. Denote the covariance function of Y as $R(\boldsymbol{x}, \boldsymbol{y}) = E[Y(\boldsymbol{x})Y(\boldsymbol{y})]$. Then such basis functions $\{\phi_k\}$ satisfy

$$E[Y_i Y_j] = \int_\Omega \phi_i(\boldsymbol{x}) d\boldsymbol{x} \int_\Omega R(\boldsymbol{x}, \boldsymbol{y})\phi_j(\boldsymbol{y}) d\boldsymbol{y} = 0, \qquad i \ne j.$$

Since $\{\phi_k\}$ is complete and orthonormal in $L^2(\Omega)$, it follows that $\phi_k(\boldsymbol{x})$ are eigenfunctions of $R(\boldsymbol{x}, \boldsymbol{y})$:

$$\int_\Omega R(\boldsymbol{x}, \boldsymbol{y})\phi_k(\boldsymbol{y}) d\boldsymbol{y} = \lambda_k \phi_k(\boldsymbol{x}), \qquad k = 1, 2, \ldots, \tag{4.1}$$

where $\lambda_k = E[Y_k^2] > 0$. Furthermore, we have

$$R(\boldsymbol{x}, \boldsymbol{y}) = E[Y(\boldsymbol{x})Y(\boldsymbol{y})] = \sum_{k=1}^{\infty} \lambda_k \phi_k(\boldsymbol{x})\phi_k(\boldsymbol{y}). \tag{4.2}$$

Denote $\theta_k = Y_k/\sqrt{\lambda_k}$; then θ_k satisfy $E(\theta_k) = 0$ and $E(\theta_i \theta_j) = \delta_{ij}$. It follows that

$$Y(\boldsymbol{x}, \omega) = \sum_{k=1}^{\infty} \sqrt{\lambda_k}\theta_k(\omega)\phi_k(\boldsymbol{x}), \tag{4.3}$$

where ϕ_k and λ_k satisfy (4.1). We assume that the eigenvalues λ_k are ordered $\lambda_1 \ge \lambda_2 \ge \cdots$. The expansion (4.3) is called the Karhunen–Loève expansion (KLE) of

the stochastic process $Y(\boldsymbol{x},\omega)$. For finite discrete processes, the KLE reduces to the principal component decomposition.

In (4.3), the L^2 basis functions $\phi_k(\boldsymbol{x})$ are deterministic and resolve the spatial dependence of the permeability field. The randomness is represented by the scalar random variables θ_k. Generally, we need only to keep the leading order terms (quantified by the magnitude of λ_k) and still capture most of the energy of the stochastic process $Y(\boldsymbol{x},\omega)$. For a N-term KLE approximation $Y_N = \sum_{k=1}^{N} \sqrt{\lambda_k}\theta_k\phi_k$, we define the energy ratio of the approximation as

$$e(N) := \frac{E\|Y_N\|^2}{E\|Y\|^2} = \frac{\sum_{k=1}^{N}\lambda_k}{\sum_{k=1}^{\infty}\lambda_k}.$$

If $\lambda_k, k = 1, 2, \ldots$, decay very fast, then the truncated KLE would be good approximations of the stochastic process in the L^2 sense.

Suppose the permeability field $k(\boldsymbol{x},\omega)$ is a log-normal homogeneous stochastic process. Then $Y(\boldsymbol{x},\omega)$ is a Gaussian process and θ_k are independent standard Gaussian random variables. We assume that the covariance function of $Y(\boldsymbol{x},\omega)$ has the form

$$R(\boldsymbol{x},\boldsymbol{y}) = \sigma^2 \exp\left(-\frac{|x_1-y_1|^2}{2L_1^2} - \frac{|x_2-y_2|^2}{2L_2^2}\right). \tag{4.4}$$

In the above formula, L_1 and L_2 are the correlation lengths in each dimension, and $\sigma^2 = E(Y^2)$ is a constant. In our first example, we set $L_1 = 0.2$, $L_2 = 0.2$, and $\sigma^2 = 2$. We first solve the eigenvalue problem (4.1) numerically and obtain the eigenpairs $\{\lambda_k, \phi_k\}$. In Figure 4.1 the first 50 eigenvalues are plotted. As we can see, the eigenvalues of the KLE decay very fast. It has been shown in [8] that the eigenvalues decay exponentially fast for the covariance function (4.4). Therefore, only a small number of terms need to be retained in the truncated expansion (4.3). We can sample $Y(\boldsymbol{x},\omega)$ easily from the truncated KLE (4.3) by generating independent Gaussian random variables θ_k.

It is worth noting that for a different covariance function such as $R(\boldsymbol{x},\boldsymbol{y}) = \sigma^2 \exp\left(-\frac{|x_1-y_1|}{L_1} - \frac{|x_2-y_2|}{L_2}\right)$, the eigenvalues of the integral equation (4.1) may decay slowly (only algebraically [8]). To achieve the same accuracy, more terms should be retained in the truncated expansion (4.3), which will increase the dimension of the parameter space to represent the permeability. As a result, sampling the permeability from the distribution will be more expensive for both the direct MCMC method and the preconditioned MCMC method. However, small parameter space does not favor the preconditioned MCMC method, and the preconditioning technique is applicable independent of the problem dimension. For permeabilities with higher dimensional parameters, the acceptance rates of the direct MCMC method will be even lower. Consequently, the preconditioned MCMC method will be more preferable since its filtering procedure can increase the acceptance rates dramatically. Note that if the permeability field is not a log-normal process, then θ_k in the expansion (4.3) are not necessarily Gaussian random variables. However, we can still sample the permeability field from the truncated expansion (4.3) by sampling the random variables θ_k.

In the numerical experiments, we first generate a reference permeability field using all eigenvectors and compute the corresponding fractional flows. To propose permeability fields from the prior (unconditioned) distribution, we keep 20 terms in the KLE. Suppose the permeability field is known at eight distinct points. This

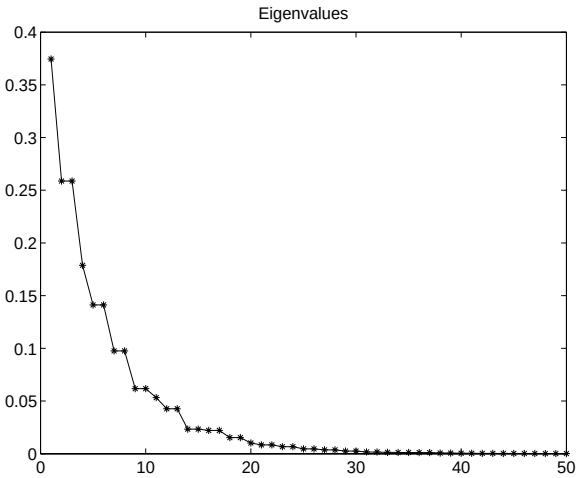


FIG. 4.1. *Eigenvalues of the KLE for the Gaussian covariance with $L_1 = L_2 = 0.2$. The eigenvalues decay very fast.*

condition is imposed by setting

$$\sum_{k=1}^{20} \sqrt{\lambda_k}\theta_k\phi_k(\boldsymbol{x}_j) = \alpha_j, \tag{4.5}$$

where α_j $(j = 1, \ldots, 8)$ are prescribed constants. For simplicity, we set $\alpha_j = 0$ for all $j = 1, \ldots, 8$. In the simulations we propose 12 θ_i and calculate the rest of θ_i by solving the linear system (4.5). In all the simulations, we test 50000 proposals and iterate the Markov chain 50000 times. Because the direct MCMC computations are expensive, we do not select the large model problems and consider only 40×40 and 60×60 fine-scale models. However, the preconditioned MCMC method is applicable independent of the size of the permeability field.

We have considered two types of instrumental proposal distributions $q(k|k_n)$: the independent sampler and the random walk sampler. In the case of the independent sampler, the proposal distribution $q(k|k_n)$ is chosen to be independent of k_n and equal to the prior (unconditioned) distribution. In the random walk sampler, the proposal distribution depends on the previous value of the permeability field and is given by

$$k = k_n + \epsilon_n, \tag{4.6}$$

where ϵ_n is a random perturbation with prescribed distribution. If the variance of ϵ_n is chosen to be very large, then the random walk sampler becomes similar to the independent sampler. Although the random walk sampler allows us to accept more realizations, it often gets stuck in the neighborhood of a local maximum of the target distribution. For both proposal distributions, we have observed consistently more than 10 times an increase in the acceptance rate when the preconditioned MCMC method is used.

For the first set of numerical tests, we use a 40×40 fine-scale permeability field and 10×10 coarse-scale models. The permeability field is assumed to be log-normal, with $L_1 = L_2 = 0.2$ and $\sigma^2 = 2$ for the covariance function (4.4). In Figure 4.2, the acceptance rates are plotted against different coarse-scale precisions, σ_c. Here

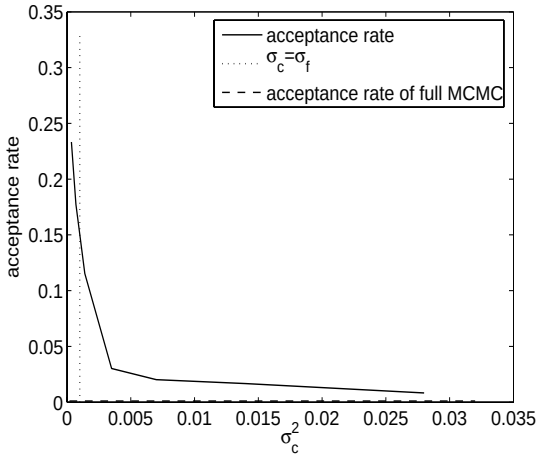


FIG. 4.2. *Acceptance rate versus different coarse-scale precisions for the preconditioned MCMC method. Single-phase flow and* $\sigma_f^2 = 0.001$.

the acceptance rate refers to the ratio between the number of accepted permeability samples and the number of fine-scale simulations that are performed. The acceptance rate for the direct (full) MCMC method is plotted using the dashed line, and it is equal to 0.001. The vertical dotted line marks the coarse-scale precision $\sigma_c = \sigma_f$. If σ_c is very small, then the total number of accepted realizations is also small, even though the acceptance rate is higher. We have found that if σ_c is of the same order as σ_f, then the preconditioned MCMC method accepts almost the same number of proposals as the direct MCMC method but requires only 10% of the fine-scale runs. Note that as σ_c increases the acceptance rate decreases and reaches the acceptance rate of the full MCMC method. Indeed, if σ_c is very large, then all the proposals will be accepted by the coarse-scale test, and there is no gain in preconditioning. In general, one can estimate the optimal σ_c based on a limited number of simulations, prior to the full simulations as described above.

In Figure 4.3 we plot the fractional flows of the accepted permeability realizations. On the left plot, the cross-plot between the reference fractional flow and the sampled fractional flows (of accepted realizations) is plotted. Since the reference fractional flow is the same for every accepted sample, the curve has jumps in the vertical direction. On the right plot, fractional flows of accepted samples are plotted using dotted lines. The bold solid line is the reference fractional flow curve. As we can see from these figures, the fractional flows of accepted realizations are very close to the observed fractional flow, because the precision is taken to be $\sigma_f^2 = 0.001$. In Figure 4.4, we plot the fractional flow error $E_k = \|F - F_k\|^2$ of the accepted samples for both the direct and preconditioned MCMC methods. We observe that the errors of both of the Markov chains converge to a steady state within 20 accepted iterations (corresponds to 20000 proposals). Note that we can assess the convergence of the MCMC methods based on the fractional flow errors. This is a reasonable indicator for the convergence and is frequently used in practice. Given the convergence result of the MCMC method, a longer chain can be easily generated when it is needed. We present a few accepted permeability realizations generated by the preconditioned MCMC method in Figure 4.5. The first plot is the reference (true) permeability field and the others are the last

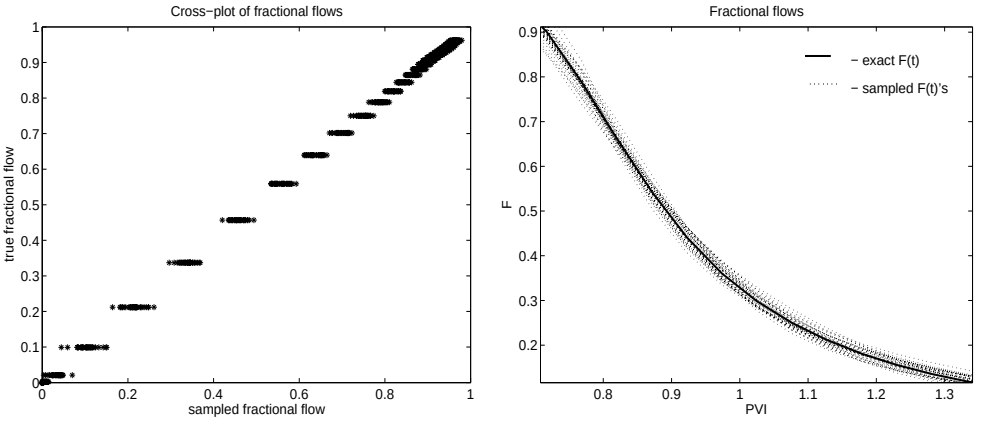


FIG. 4.3. *Fractional flow comparisons. Left: Cross-plot between the reference fractional flow and sampled fractional flows. Right: the solid line designates the fine-scale reference fractional flow, and dotted lines designate fractional flows corresponding to sampled permeability fields.*

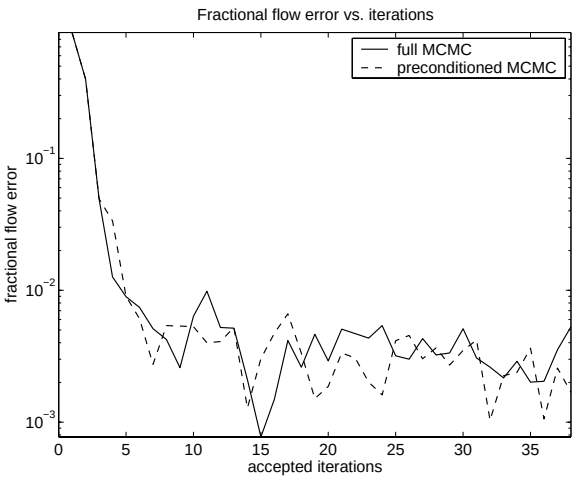


FIG. 4.4. *Fractional flow errors versus accepted iterations.*

five accepted permeability realizations. Some of these realizations closely resemble the true permeability field. Note that the fractional flows of these accepted realizations are in good agreements with the reference (true) fractional flow. One can use these samples for the uncertainty estimation.

For the next set of numerical examples, we consider an anisotropic permeability field with $L_1 = 0.4$, $L_2 = 0.05$, and $\sigma^2 = 2$ defined on a 60×60 fine grid. As in the previous example, we use eight conditioning points and truncate the KLE expansion of the permeability field with 20 terms to maintain a sufficient accuracy. In Figure 4.6, we plot the acceptance rates for 6×6 and 10×10 coarse-scale models against different choices of σ_c^2. The acceptance rate for the direct (full) MCMC method is 0.0008, and it is designated by the dashed line. The acceptance rate is increased by more than 10 times in the preconditioned MCMC method when σ_c is slightly larger than σ_f (the vertical dotted line marks the choice $\sigma_c = \sigma_f$). We also observe a higher acceptance

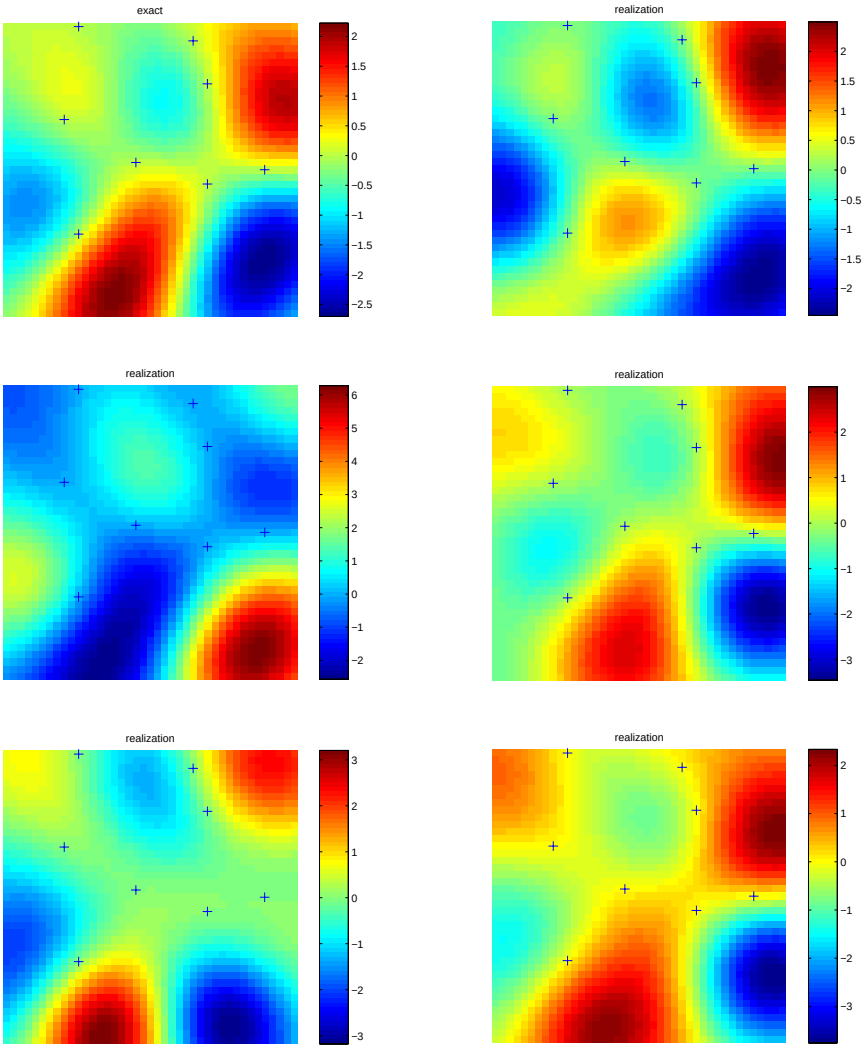


FIG. 4.5. *The last five accepted realizations of the log permeability field. The "+" sign marks the locations of the hard data.*

rate for the 10×10 coarse-scale model than for the 6×6 coarse-scale model. This is because the 10×10 coarse-scale model provides more accurate predictions of the fine-scale results compared to the 6×6 coarse-scale model. As in the previous cases, when the σ_c is slightly larger than σ_f, the preconditioned MCMC method can accept the same number of samples as the underlying full MCMC method but performs only 10% of the fine-scale simulations. Moreover, we have observed that both the direct (full) MCMC method and the preconditioned MCMC method converge to the steady state within 20 accepted iterations, which indicates that both chains have similar convergence properties. In Figure 4.7, we plot the last five accepted samples of the permeability field generated by the preconditioned MCMC method using the 6×6 coarse-scale model. Some of these samples closely resemble the reference (true)

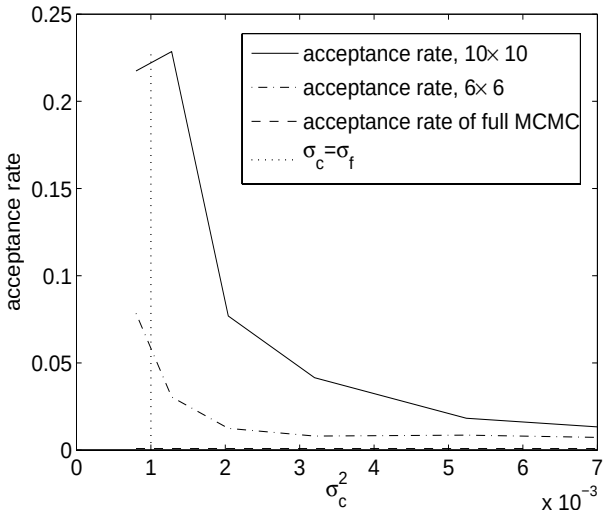


FIG. 4.6. *Acceptance rate versus different coarse-scale precisions of the MCMC method using* 6×6 *and* 10×10 *coarse-scale models. Anisotropic single-phase flow and* $\sigma_f^2 = 0.001$.

permeability field.

Our next set of numerical experiments are for the two-phase flow simulations. We have observed very similar results for two-phase flow simulations, and thus restrict our numerical results to only a few examples. We consider $\mu_w/\mu_o = 5$ and $k_{rw}(S) = S^2$, $k_{ro}(S) = (1-S)^2$. Typically, one observes similar behavior between the upscaling errors for single- and two-phase flows. We consider a 40×40 fine-scale log-normal permeability field with $L_1 = L_2 = 0.2$ and 10×10 coarse-scale models. In Figure 4.8, the acceptance rate for $\sigma_f^2 = 0.001$ is plotted. As in the case of the single-phase flow simulations, we observe more than 10 times an increase in the acceptance rate. The preconditioned MCMC method accepts the same amount of samples as in the full MCMC method with less than 10% of the fine-scale runs. To study the relative convergence of the preconditioned MCMC method, we plot the fractional flow error for both full and preconditioned MCMC simulations in Figure 4.9. It can be seen from this figure that both the full and preconditioned MCMC methods reach the steady state within 20 accepted iterations. This indicates that the direct and preconditioned MCMC methods have similar convergence properties. The typical samples for the two-phase flow are very similar to those for the single-phase flow, and we do not present them here.

Next we present some numerical results using the random walk sampler (4.6) as the instrumental proposal distribution. The random walk sampler proposes new permeability fields in a neighborhood of the previously accepted permeability field. This improves the acceptance rate in general, though the random walk sampler can get stuck in the neighborhood of the local maxima of the distribution. As a result, the MCMC method will accept a large number of realizations, but it takes a long time for the Markov chain to go from one local maxima to another local maxima. We consider 60×60 fine-scale permeability fields, with $L_1 = 0.4$, $L_2 = 0.05$, and $\sigma^2 = 2$ for the covariance function (4.4). In the preconditioning step, 10×10 and 6×6 coarse-scale models are used. In Figure 4.10, we present the acceptance rates for

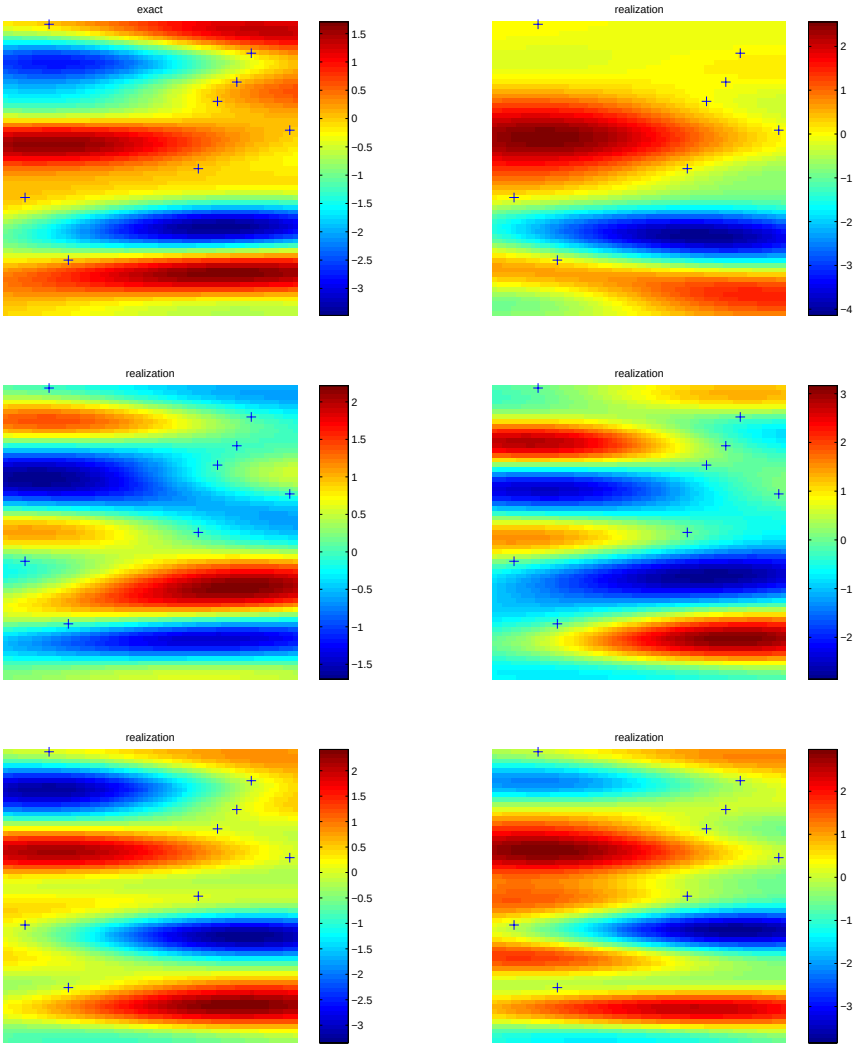


FIG. 4.7. *The last five accepted realizations of the log permeability field for the anisotropic case. The "+" sign marks the locations of the hard data.*

both coarse-scale models when the side-to-side boundary condition is used. In both cases, the acceptance rates are increased several times. In particular, the acceptance rate reaches its peak for σ_c close to σ_f and decreases as σ_c increases. We find that the generated chain k_n has a long correlation length and the nearby accepted permeability realizations are similar to each other. This indicates that the permeability realizations are sampled from a neighborhood of a local maxima, and consequently many proposals are required for a proper sampling. Next we study the convergence of the direct (full) and preconditioned MCMC methods using the random walk sampler (4.6). Figure 4.11 is the plot of the fractional flow errors against accepted iterations. As we can see from this figure, both the full MCMC method and the preconditioned MCMC method converge within 20 accepted iterations.

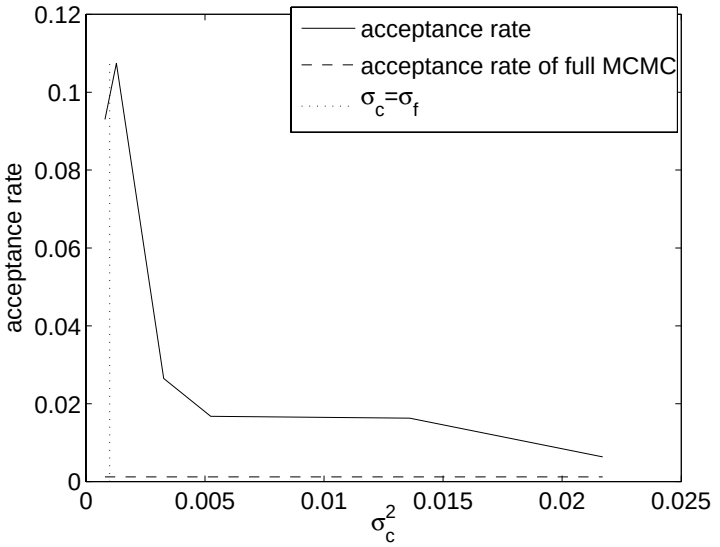


FIG. 4.8. *Acceptance rate versus coarse-scale precision of the MCMC method. Two-phase flow and* $\sigma_f^2 = 0.001$.

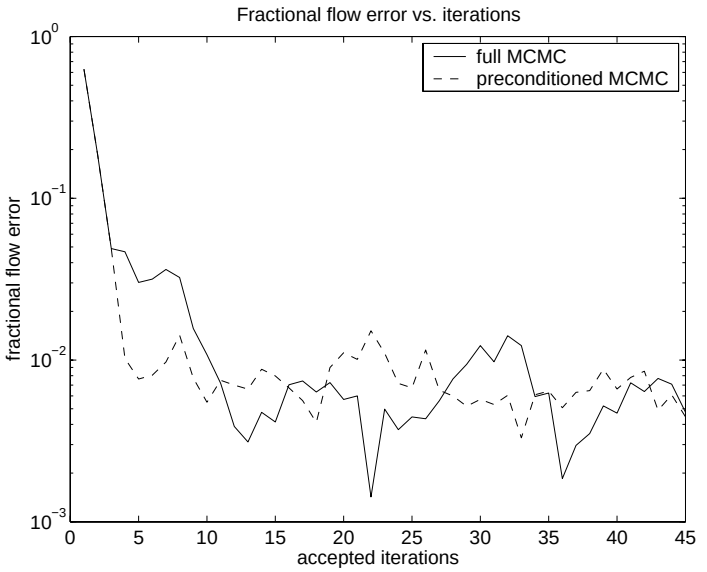


FIG. 4.9. *Fractional flow errors vresus accepted iterations in two phase-flow.*

Finally, we test the preconditioned MCMC method when different boundary conditions are used. In Figure 4.12, we compare the acceptance rates using 10×10 and 6×6 coarse-scale models for the side-to-side and the corner-to-corner boundary conditions. We obtain similar increases of the acceptance rates in the preconditioned MCMC method for the different boundary conditions. We have tested the preconditioned MCMC algorithm with more complicated boundary conditions involving mul-

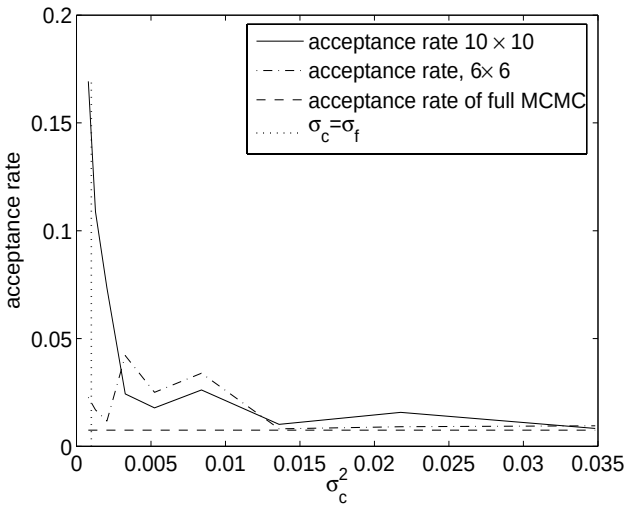


FIG. 4.10. *Acceptance rate versus coarse-scale precision of the MCMC method using* 6×6 *and* 10×10 *coarse-scale models. Single-phase flow and* $\sigma_f^2 = 0.001$. *The random walk sampler is used as the proposal distribution.*

tiple wells (source terms) that arise in petroleum applications (see [5]). In these numerical tests, the single-phase flow upscaling is used (as in [4]) since the multiscale finite volume methods require additional modifications to take into account the well information. The resulting preconditioned MCMC method can increase the acceptance rates by several times. In general, we have found the multiscale finite volume methods to be more accurate for coarse-scale simulations, and they can be further used for efficient and robust fine-scale simulations.

As we mentioned earlier, the full MCMC method and the preconditioned MCMC method accept approximately the same amount of samples for a fixed number of tested proposals. Denote N as the total number of tested proposals; then the direct MCMC method requires exactly N number of fine-scale simulations. Suppose $M < N$ is the number of fine-scale simulations conducted in the preconditioned MCMC method. Denote t_f and t_c as the CPU times for a single fine-scale and coarse-scale forward simulation. Then the computational costs for the direct MCMC method and the preconditioned MCMC method would be Nt_f and $Nt_c + Mt_f$, respectively. Therefore, the CPU cost for the preconditioned MCMC method is only $\frac{t_c}{t_f} + \frac{M}{N}$ that of the direct MCMC method. The coarse-scale computational cost t_c is usually much smaller than the fine-scale computational cost t_f. Suppose the fine-scale model is upscaled 5 times in each direction. Then solving the pressure equation at each time step is about 25 times faster on the coarse grid than on the fine grid. Moreover, the saturation equation is also solved on the [illegible] and with [illegible]. This makes the overall coarse-scale computations of the two-phase flow equation at least 25 times faster than the fine-scale computations, i.e., $t_c \approx 0.04 t_f$. If the acceptance rate is increased by more than 10 times in the preconditioned MCMC method, as in our numerical experiments, then $\frac{M}{N} < 0.1$, and the overall CPU cost of the preconditioned MCMC method would be only 10% of the CPU costs of the direct MCMC method. Note that using very coarse-scale models (fewer coarse blocks) reduces t_c but increases the

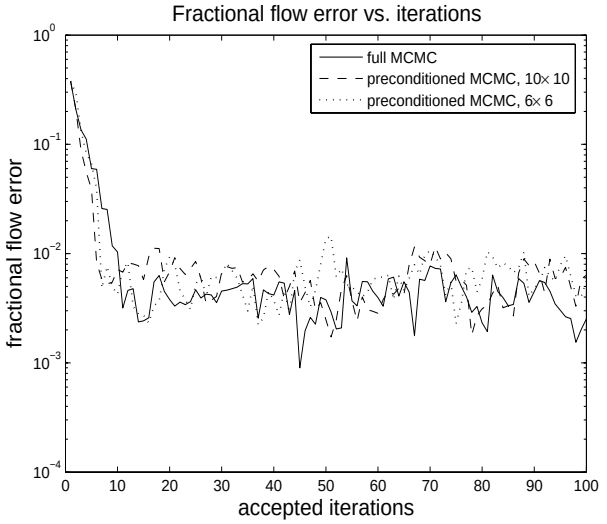


FIG. 4.11. *Fractional flow errors versus accepted iterations for* 6×6 *and* 10×10 *coarse-scale models. Single-phase flow and* $\sigma_f^2 = 0.001$. *The random walk sampler is used as the proposal distribution.*

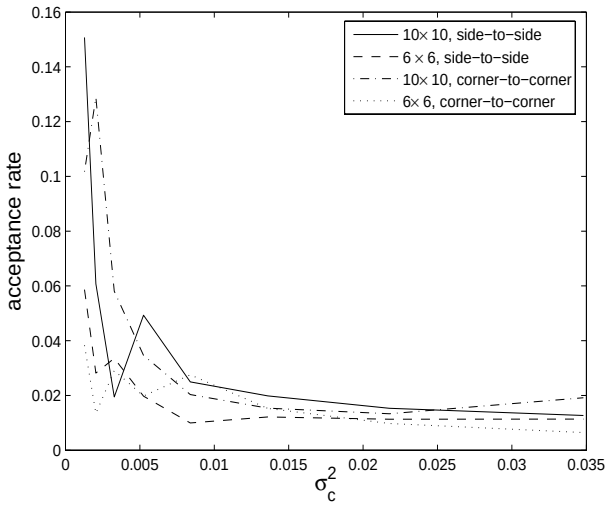


FIG. 4.12. *Acceptance rates of the preconditioned MCMC method using* 6×6 *and* 10×10 *coarse-scale models for side-to-side and corner-to-corner boundary conditions. Single-phase flow and* $\sigma_f^2 = 0.001$. *The random walk sampler is used as the proposal distribution.*

fine-scale run ratio $\frac{M}{N}$. On the other hand, using finer coarse-scale models reduces the ratio $\frac{M}{N}$ but increases t_c. Consequently, a somewhat moderate coarsening (5–10 times coarsening in each direction for large-scale fine models) can provide an optimal choice in the preconditioning of the MCMC simulations.

One can use cruder approximation methods instead of physics-based upscaling methods in preconditioning the MCMC simulations. Next we discuss applying simple

averaging methods in the preconditioned MCMC method. Suppose that the proposal $k(\boldsymbol{x})$ can be represented by the KLE $\log(k(\boldsymbol{x})) = \sum_{k=1}^{n} c_k \phi_k(\boldsymbol{x})$. Denote $\phi_k^*(\boldsymbol{x})$ as the spatial average of $\phi_k(\boldsymbol{x})$ on the coarse grid

$$\phi_k^*(\boldsymbol{x}) = \sum_i \left(\frac{1}{|\Omega_i|} \int_{\Omega_i} \phi_k(\boldsymbol{x})\, d\boldsymbol{x} \right) 1_{\Omega_i}(\boldsymbol{x}),$$

where Ω_i are the coarse blocks. Then $k^*(\boldsymbol{x}) = \exp(\sum_{k=1}^{n} c_k \phi_k^*(\boldsymbol{x}))$ is a coarse-scale approximation of $k(\boldsymbol{x})$. We can use $k^*(\boldsymbol{x})$ in the coarse-scale simulations to determine whether or not to run the fine-scale simulations. We would like to note that this type of averaging is less expensive compared to the upscaling method used in the paper because it involves only volume average and it is performed only once. However, in general this type of averaging does not represent the correct average flow properties, and consequently the strong correlation between the fine-scale and coarse-scale quantities is not guaranteed. Our numerical results show that using simple averaging methods, such as the one presented here, can give an incorrect sampling. We have observed that averaging the KLE eigenfunctions leads to more uniform permeability fields. Consequently, the first stage of the preconditioned MCMC method restricts the proposal permeability to the more uniform fields and leads to incorrect sampling of the multimodal target distribution.

Finally, we would like to point out that the coarse-scale approximation techniques can also be efficiently used for other instrumental distributions. In our recent work [3], we have used coarse-scale approximations based on the multiscale finite volume methods in Langevin MCMC algorithms. In the Langevin MCMC algorithms, the gradient of the posterior distribution is used in the instrumental proposal distribution. The computation of the gradient of the posterior distribution is very expensive. We have employed the coarse-scale model in approximating the gradient and used a two-stage MCMC method in filtering these proposals. We have shown that one can achieve the acceptance rate comparable to the fine-scale Langevin MCMC with much less CPU time.

5. Conclusion. In this paper, we study the preconditioning of MCMC simulations using inexpensive coarse-scale runs in inverse problems related to subsurface characterization. For each MCMC proposal, a coarse-scale simulation is performed to decide whether or not to run the fine-scale simulations. The coarse scale simulation, which is based on the multiscale finite volume methods, filters unlikely acceptable proposals and avoids expensive fine-scale simulations for them. The filtering process takes into account the coarse-scale information of the problem and modifies the Markov chain generated by the MCMC method. We formulate the conditions which guarantee that the modified chain will converge to the correct posterior distribution. We also discuss the applicability of these conditions to the commonly used upscaling methods. Numerical examples show that we can achieve more than 10 times an increase in the acceptance rate if the MCMC simulations are preconditioned using coarse-scale models. The sampled realizations of the permeability field can be used in uncertainty quantification.

Appendix A. Coarse-scale models using multiscale finite volume methods. In this appendix, we discuss the details of the upscaled model used in the paper. The key idea of the method is the construction of the finite element basis functions on the coarse grids, such that these basis functions capture the small scale information on each coarse block. The method we will use follows its finite element counterpart

presented in [11]. The basis functions are constructed from the solution of the leading order homogeneous elliptic equation on each coarse element with carefully chosen boundary conditions. For a coarse element K with d vertices, the local basis functions $\phi^i, i = 1, \ldots, d$, satisfy the following elliptic problem:

$$\begin{aligned} -\nabla \cdot (k \cdot \nabla \phi^i) &= 0 \quad \text{in } K, \\ \phi^i &= g^i \quad \text{on } \partial K \end{aligned} \tag{A.1}$$

for some functions g^i defined on the boundary of the coarse element K. Hou and Wu [11] have demonstrated that a careful choice of the boundary condition would guarantee that the basis functions capture the local information of the solution, and hence improve the accuracy of the method. The function g^i for each i varies linearly along ∂K. Thus, ϕ^i will reduce to a standard linear/bilinear basis function for a constant diagonal tensor. Note that as usual we require $\phi^i(\xi_j) = \delta_{ij}$. Finally, a nodal basis function associated with each vertex ξ is constructed from the combination of the local basis functions that share this ξ. These nodal basis functions are denoted by $\{\psi_\xi\}_{\xi \in Z_h^0}$.

Denote by V^h the space for the approximate pressure solution which is spanned by the basis functions $\{\psi_\xi\}_{\xi \in Z_h^0}$. Based on (2.2), a statement of mass conservation is formed on each control volume V_ξ, where the approximate solution is expressed as a linear combination of the basis functions. Assembly of this mass conservation statement on all control volumes would give rise to a linear system of equations that can be solved accordingly. The resulting linear system has incorporated the fine-scale information through the involvement of the nodal basis functions on the approximate solution. To be more specific, the problem now is to seek $p^h \in V^h$ with $p^h = \sum_{\xi \in Z_h^0} p_\xi \psi_\xi$ such that

$$\int_{\partial V_\xi} \lambda(S) k \cdot \nabla p^h \cdot \vec{n}\, dl = \int_{V_\xi} f\, dA \tag{A.2}$$

for every control volume $V_\xi \subset \Omega$. Here $\vec{n}$ denotes the unit normal vector on the boundary ∂V_ξ of the control volume, and S is the fine-scale saturation field at this point. We note that concerning the basis functions, a vertex-centered finite volume difference is used to solve (A.1).

Once the pressure solution is available, it can be used to compute the total velocity field at the coarse-scale level, denoted by $\overline{\boldsymbol{v}} = (\overline{v}_x, \overline{v}_z)$ via (2.5). In general, the following formulas are used to compute the velocities in the horizontal and vertical directions, respectively:

$$\overline{v}_x = -\frac{1}{h_z} \sum_{\xi \in Z_h^0} p_\xi \left(\int_E \lambda(S) k_x \frac{\partial \psi_\xi}{\partial x} dz \right), \tag{A.3}$$

$$\overline{v}_z = -\frac{1}{h_x} \sum_{\xi \in Z_h^0} p_\xi \left(\int_E \lambda(S) k_z \frac{\partial \psi_\xi}{\partial z} dx \right), \tag{A.4}$$

where E is the edge of V_ξ. Furthermore, for the control volumes V_ξ adjacent to the Dirichlet boundary (which are half control volumes), we can derive the velocity approximation using the conservation statement derived from (2.2) on V_ξ. One of the terms involved is the integration along part of the Dirichlet boundary, while the other

three terms are known from the adjacent internal control volumes calculations. The analysis of the two-scale finite volume method can be found in [7].

As for the upscaling of the saturation equation, we use the coarse-scale velocity to update the saturation field on the coarse grid, i.e.,

$$\frac{\partial \overline{S}}{\partial t} + \overline{\boldsymbol{v}} \cdot \nabla f(\overline{S}) = 0, \tag{A.5}$$

where $\overline{S}$ denotes the saturation on the coarse grid. In this case the upscaling of the saturation equation does not take into account the subgrid effects. As we mentioned above, one can reconstruct the velocity field and solve the saturation equation on the fine grid. The latter, though more expensive, provides an accurate approximation of the production data.

Acknowledgments. The authors would like to thank the referees for valuable comments and suggestions and Victor Ginting for his help in preparing this manuscript.

REFERENCES

[1] J. E. Aarnes, *On the use of a mixed multiscale finite element method for greater flexibility and increased speed or improved accuracy in reservoir simulation*, Multiscale Model. Simul., 2 (2004), pp. 421–439.
[2] J. A. Christen and C. Fox, *Markov Chain Monte Carlo using an approximation*, J. Comput. Graph. Statist., 14 (2005), pp. 795–810.
[3] P. Dostert, Y. Efendiev, T. Hou, and W. Luo, *Coarse-gradient Langevin algorithms for dynamic data integration and uncertainty quantification*, J. Comput. Phys., to appear.
[4] L. J. Durlofsky, *Numerical calculation of equivalent grid block permeability tensors for heterogeneous porous media*, Water Resour. Res., 27 (1991), pp. 699–708.
[5] Y. Efendiev, A. Datta-Gupta, V. Ginting, X. Ma, and B. Mallick, *An efficient two-stage Markov chain Monte Carlo method for dynamic data integration*, Water Resour. Res., W12423.
[6] Y. Efendiev, V. Ginting, T. Hou, and R. Ewing, *Accurate multiscale finite element methods for two-phase flow simulations*, J. Comput. Phys., to appear.
[7] V. Ginting, *Analysis of two-scale finite volume element method for elliptic problem*, J. Numer. Math., 12 (2004), pp. 119–142.
[8] P. Frauenfelder, C. Schwab, and R. A. Todor, *Finite elements for elliptic problems with stochastic coefficients*, Comput. Methods Appl. Mech. Engrg., 194 (2005), pp. 205–228.
[9] J. Glimm and D. H. Sharp, *Prediction and the quantification of uncertainty*, Phys. D, 133 (1999), pp. 152–170.
[10] D. Higdon, H. Lee, and Z. Bi, *A Bayesian approach to characterizing uncertainty in inverse problems using coarse and fine-scale information*, IEEE Trans. Signal Process., 50 (2002), pp. 388-399.
[11] T. Y. Hou and X. H. Wu, *A multiscale finite element method for elliptic problems in composite materials and porous media*, J. Comput. Phys., 134 (1997), pp. 169–189.
[12] P. Jenny, S. H. Lee, and H. Tchelepi, *Multi-scale finite volume method for elliptic problems in subsurface flow simulation*, J. Comput. Phys., 187 (2003), pp. 47–67.
[13] P. Jenny, S. H. Lee, and H. A. Tchelepi, *Adaptive multiscale finite-volume method for multiphase flow and transport in porous media*, Multiscale Model. Simul., 3 (2005), pp. 50–64.
[14] P. Kitanidis, *Quasi-linear geostatistical theory for inversing*, Water Resour. Res., 31 (1995), pp. 2411–2419.
[15] J. S. Liu, *Monte Carlo Strategies in Scientific Computing*, Springer, New York, 2001.
[16] O. Lodoen, H. Omre, L. Durlofsky, and Y. Chen, *Assessment of uncertainty in reservoir production forecasts using upscaled models*, in Proceedings of the Seventh International Geostatistics Congress, Banff, AB, Canada, 2004.
[17] M. Loeve, *Probability Theory*, 4th ed., Springer, Berlin, 1977.
[18] S. P. Meyn and R. L. Tweedie, *Markov Chains and Stochastic Stability*, Springer, London, 1996.

[19] D. Oliver, L. Cunha, and A. Reynolds, *Markov chain Monte Carlo methods for conditioning a permeability field to pressure data*, Math. Geol., 29 (1997), pp. 61–91.
[20] D. Oliver, N. He, and A. Reynolds, *Conditioning permeability fields to pressure data*, in Proceedings of the 5th European Conference on the Mathematics of Oil Recovery, Leoben, Austria, 1996.
[21] H. Omre and O. P. Lodoen, *Improved production forecasts and history matching using approximate fluid-flow simulators*, SPE Journal, (2004), pp. 339–351.
[22] C. Robert and G. Casella, *Monte Carlo Statistical Methods*, Springer, New York, 1999.
[23] E. Wong, *Stochastic Processes in Information and Dynamical Systems*, McGraw–Hill, New York, 1971.

SIAM J. SCI. COMPUT.
Vol. 28, No. 3, pp. 805–831
© 2006 Society for Industrial and Applied Mathematics

RKDG FINITE ELEMENT METHOD COMBINED WITH BGK SCHEME FOR SOLVING FLUID DYNAMICS SYSTEM*

QINGFANG DAI† AND XIJUN YU†

Abstract. This paper presents a new type of second-order scheme for solving the system of Euler equations, which combines the Runge–Kutta discontinuous Galerkin (RKDG) finite element method and the Bhatnagar–Gross–Krook (BGK) scheme. We first discretize the Euler equations in space with the DG method, and then the resulting system will be discretized using an RK method. Finally, a second-order BGK method is used to construct the numerical flux. The proposed scheme keeps the main advantages of the DG finite element method, including its flexibility in handling irregular solution domains and in parallelization. Moreover, with the BGK approach the proposed schemes avoid using Riemann solvers. The efficiency and effectiveness of the proposed method are illustrated by several numerical examples in one- and two-dimensional spaces.

Key words. Runge–Kutta method, discontinuous Galerkin method, Bhatnagar–Gross–Krook method, Euler equations

AMS subject classifications. 65L06, 65M60, 82C40, 35L65

DOI. 10.1137/040618783

1. Introduction.

In the past few years, there has been much important progress in constructing and analyzing high-resolution schemes for solving the hydromechanical Euler equations. Several high-order nonoscillatory schemes have been proposed, including TVD schemes [3], ENO schemes [4], NND schemes [38], and MmB schemes [33]. In the existing schemes, the upwinding-type schemes are very popular, partly because their dissipation is lower than that of many other types of schemes and they are very robust. However, upwinding schemes require computing exact or approximate Riemann solvers when being applied for solving hyperbolic conservation laws, which increases the numerical complexity and computational time. To overcome this difficulty, several methods have been investigated, including the flux vector splitting (FVS) scheme which splits the macroscopic flux vector directly. The first FVS scheme based on the gas-kinetic theory is the beam scheme which was proposed by Sanders and Prendergast [30]. Since the construction of the beam scheme uses three delta functions to approximate the equilibrium Maxwellian distribution function, the numerical dissipation is still high. The kinetic flux vector splitting (KFVS) method, which is similar to the beam scheme, splits the macroscopic flux vector based on the transport velocity symbol of the microcosmic molecule or particle. The KFVS scheme has been extensively studied in recent years, including by Pullin [14], Reitz [29], Deshpande [31], Macrossan [26], Perthame [12], and Eppard and Grossman [32]. Since the KFVS method is based on the collisionless Boltzmann equation, which ignores the particle collisions, the numerical dissipation of this method is higher than that of those classical high-resolution schemes. To include the real particle collisions in reducing the overdissipation in the KFVS scheme, a more accurate physical model

*Received by the editors August 29, 2005; accepted for publication (in revised form) November 16, 2005; published electronically June 9, 2006. This research was supported by Natural Science Fund of China grant 10471011.

http://www.siam.org/journals/sisc/28-3/61878.html

†Laboratory of Computational Physics, Institute of Applied Physics and Computational Mathematics, P.O. Box 8009-26, Beijing 100088, People's Republic of China (qdai@math.ualberta.ca, yuxj@mail.iapcm.ac.cn).

to approximate the particle collisions is needed. A good candidate for this purpose is the Bhatnagar–Gross–Krook (BGK) model. In 1990 [17], the BGK method based on the BGK model was developed. The BGK model is a microcosmic dynamical model raised by the concept of collision interval τ. The collision interval theory assumes that during a time dt, a fraction of dt/τ of molecules in a given small volume undergoes collision, which alters the velocity-distribution function from f to g (Maxwellian distribution function g is corresponding to the local values of ρ, u and T). This is equivalent to assuming that the rate of changes of f due to collisions is $(g - f)/\tau$, so the Boltzmann equation becomes

$$f_t + v \cdot \nabla_x f = \frac{g - f}{\tau}, \tag{1.1}$$

where $v = (v_x, v_y, v_z)$ is particle velocity and $u = (u_x, u_y, u_z)$ is macroscopic flow velocity. The earliest results were published in papers [16, 18]. This scheme has been further studied and simplified [19, 20, 21]. At the same time, the BGK method has been extended to multicomponent flow calculation [22] and the Euler equations with heat transfer [24]. In recent years, the BGK method has found its way into many applications, including astrophysics [15], aerodynamics [13], hydraulic engineering [25], and physical science [1, 2]. It has been demonstrated that the BGK scheme is robust and accurate in simulating various unsteady compressible flows. Both KFVS and BGK methods do not require computing Riemann solvers.

The discontinuous Galerkin (DG) finite element method was first introduced by Reed and Hill [34] to solve the neutron transport equation. The DG method has several advantages, including its flexibility in handling irregular solution domain, the requirement of fewer adjacent gridding points, and its highly self-adaptive and parallelization properties. Therefore, the DG finite element method has been developed rapidly in the past two decades. In particular, Cockburn and Shu successfully used the Runge–Kutta discontinuous Galerkin (RKDG) method and nonlinear limiters in solving hyperbolic conversation laws and the convection-dominated problems [5, 6, 7, 8, 9, 10, 11]. However, their schemes require using the Riemann solver, and it is difficult to distinguish the shock waves on the adjacent element interface. The main purpose of this work is to overcome these difficulties by improving the RKDG methods. To this end, we will apply the BGK flux constructing techniques based on the Boltzmann equation with collision term to RKDG methods. This work is to continue an earlier effort of the authors in [28, 35], where the RKDG methods were combined with the KFVS and the first-order BGK flux constructing techniques to solve the Euler equations. This work will make the scheme second-order accurate, which will be used to solve one-dimensional (1D) and two-dimensional (2D) Euler equations.

The organization of this paper is as follows. A class of second-order schemes obtained by combining the RKDG method with BGK numerical flux will be designed for 1D Euler equations in section 2 and 2D equations in section 3. Section 4 is devoted to the numerical experiments, which demonstrate the efficiency and effectiveness of the proposed schemes. The final section gives some concluding remarks.

2. Second-order scheme in one dimension. We consider the initial value problem for 1D hyperbolic conservation laws [37]:

$$\begin{cases} \dfrac{\partial U}{\partial t} + \dfrac{\partial F(U)}{\partial x} = 0 & \forall (x,t) \in R \times (0,T), \\ U(x,0) = U_0(x) & \forall x \in R, \end{cases} \tag{2.1}$$

where

$$U = (\rho, \rho u, E)^T, \quad F(U) = (\rho u, \rho u^2 + p, u(E + p))^T,$$

ρ is the density, u is the velocity, E is the total energy, and p is the pressure.

2.1. The DG finite element space discretization. For a partition of the interval R, $R = \cup_j^{\infty} I_j$, we set $I_j = (x_{j-1/2}, x_{j+1/2})$ and $\Delta x_j = x_{j+1/2} - x_{j-1/2}$ and use the piecewise interpolation polynomials to approximate U on I_j. The space of interpolated functions is

$$V_h = V_h^k = \{v_h \in L^{\infty}(R) : v_h \in P^k(I_j)\ \forall I_j\},$$

where $P^k(I_j)$ is a polynomial space of degree not exceeding k on I_j. Choose

$$P^k(I_j) = span\{v_i^{(j)}(x)\}_{i=0}^k,$$

where $v_i^{(j)}(x)$ is the Legendre polynomial over I_j,

$$v_0^{(j)}(x) = 1, v_1^{(j)}(x) = x - x_j, v_2^{(j)}(x) = (x - x_j)^2 - \frac{1}{12}\Delta x_j^2, \ldots. \tag{2.2}$$

Then the approximate solution of (2.1), $U_h \in V_h$, is defined by

$$U_h(x,t) = \sum_{l=0}^{k} a_l U_j^{(l)}(t) v_l^{(j)}(x)\ \forall x \in I_j, \tag{2.3}$$

where the coefficients a_l are given by

$$a_l = \frac{\Delta x_j^{l+1}}{\int_{I_j} (v_l^j(x))^2 dx}. \tag{2.4}$$

In particular, the first three coefficients are given by

$$a_0 = 1, \quad a_1 = 12/\Delta x_j, \quad a_2 = 180/\Delta x_j^2.$$

The degree of freedom in (2.3), $U_j^{(l)}(t)$, is defined by

$$U_j^{(l)} = U_j^{(l)}(t) = \frac{1}{\Delta x^{l+1}} \int_{I_j} U(x,t) v_l^{(j)}(x) dx, \quad l = 0, 1, \ldots, k. \tag{2.5}$$

To determine $U_j^{(l)}(t)$, we multiply (2.1) by $v_h \in V_h^k$ and integrate the resulting equation over I_j. Using integration by parts gives

$$\begin{aligned} &\frac{d}{dt} \int_{I_j} U(x,t) v_h(x) dx + [\Delta_+(v_h(x_{j-1/2}) F(U(x_{j-1/2}, t)))] \\ &\quad - \int_{I_j} F(U(x,t)) \frac{d}{dx} v_h(x) dx = 0\ \forall v_h \in V_h^k,\ \forall j \geq 1, \end{aligned} \tag{2.6}$$

where $\Delta_{\pm} a_j = \pm(a_{j\pm1} - a_j)$. In (2.6), the exact solution U will be replaced by its approximation, U_h, and $F(U(x_{j+1/2}, t))$ will be approximated by some monotone

numerical fluxes, $H_{j+1/2} = H(U_j, U_{j+1})$, where U_j is the average value of U_h over I_j. Then we obtain

$$\begin{aligned}(2.7)\qquad &\frac{d}{dt}U_j^{(l)}(t) + \frac{1}{\Delta x_j^{l+1}}[\Delta_+(v_l^{(j)}(x_{j-1/2})H_{j-1/2})] \\ &- \frac{1}{\Delta x_j^{l+1}}\int_{I_j} F(U_h(x,t))\frac{d}{dx}v_l^{(j)}(x)dx = 0, \quad l = 0, 1, \ldots, k, \quad j = 1, 2, \ldots.\end{aligned}$$

Note that the choice of numerical flux has no relation to the choice of finite element space. Therefore, if we choose $k = 0$, then the finite element space is a piecewise constant space. In this case, (2.7) can be rewritten as follows:

$$(2.8)\qquad \frac{d}{dt}U_j^{(0)}(t) + (H_{j+1/2} - H_{j-1/2})/\Delta x_j = 0 \ \forall j \geq 1.$$

This indicates that the first-order method above is a finite volume-type method and can also be regarded as a finite difference method when a uniform mesh is used.

It is known that properly constructing numerical fluxes is a key to the numerical methods for hyperbolic conservation laws. If (2.8) is a monotone scheme, then the numerical flux $H(a, b)$ has to satisfy the following conditions:

(1) $H(a, b)$ is locally Lipschitz continuous and consistent with the flux F, i.e.,

$$H(U, U) = F(U).$$

(2) $H(a, b)$ is a nondecreasing function of its first argument and a nonincreasing function of its second argument.

2.2. The RK time discretization. It is noted that (2.7) leads to an ODE system

$$(2.9)\qquad \frac{d}{dt}U_h = L_h(U_h, t).$$

In order to obtain high-order accuracy in time, we use the RK formula to solve the ODE system (2.9). The $(k + 1)$th order accuracy in space can be achieved in smooth regions when the kth order polynomials are chosen (see, e.g., [5, 7]). Thus, to unify the accuracy in space and time, we use an $(k + 1)$th order time discretization. Similar to the finite difference theory, for the RKDG method the numerical solution at each time step also needs to be limited (only for the polynomials of degree ≥ 1) to enhance the nonlinear stability of the numerical method.

Let $\{t^n\}_0^N$ be a partition for $[0, T]$ and $\Delta t^n := t^{n+1} - t^n, 0 \leq n \leq N - 1$, be the time step. Below we briefly describe the $(k + 1)$th order RK time discretization for (2.9).

1. $U_h^0 = P(U_0)$.
2. For $n = 0, \ldots, N - 1$ compute U_h^{n+1} as follows:
   a. set $U_h^{(0)} = U_h^n$.
   b. for $i = 1, \ldots, k + 1$ compute the intermediate functions:

$$U_h^{(i)} = \sum_{l=0}^{i-1} [\alpha_{il}U_h^{(l)} + \beta_{il}\Delta t^n L_h(U_h^{(l)}, t^n + d_l\Delta t^n)].$$

3. Set $U_h^{n+1} = U_h^{(k+1)}$.

After a local slope limiter operator $\Lambda\Pi_h$ is introduced, the above algorithm becomes the following.

1. $U_h^0 = \Lambda\Pi_h P(U_0)$.
2. For $n = 0, \ldots, N-1$ compute U_h^{n+1} as follows:
   a. set $U_h^{(0)} = U_h^n$.
   b. for $i = 1, \ldots, k+1$ compute the intermediate functions:

$$U_h^{(i)} = \Lambda\Pi_h \left\{ \sum_{l=0}^{i-1} [\alpha_{il} U_h^{(l)} + \beta_{il} \Delta t^n L_h(U_h^{(l)}, t^n + d_l \Delta t^n)] \right\}. \tag{2.10}$$

3. Set $U_h^{n+1} = U_h^{(k+1)}$.

Some detailed analysis for the limiter $\Lambda\Pi_h$ can be found in [5]. The parameters in the above formula are as follows.

• Second-order accuracy (i.e., $k+1=2$):

$$\alpha_{10} = \beta_{10} = 1; \quad \alpha_{20} = \alpha_{21} = \beta_{21} = \tfrac{1}{2}$$
$$\beta_{20} = 0; \quad d_0 = 0; \quad d_1 = 1.$$

• Third-order accuracy (i.e., $k+1=3$):

$$\alpha_{10} = \beta_{10} = 1; \quad \alpha_{20} = \tfrac{3}{4}; \quad \beta_{20} = 0;$$
$$\alpha_{21} = \beta_{21} = \tfrac{1}{4}; \quad \alpha_{30} = \tfrac{1}{3}; \quad \beta_{30} = \alpha_{31} = \beta_{31} = 0;$$
$$\alpha_{32} = \beta_{32} = \tfrac{2}{3}; \quad d_0 = 0; \quad d_1 = 1, \quad d_2 = \tfrac{1}{2}.$$

The numerical scheme (2.7) with (2.10) is the RKDG finite scheme for solving the 1D hyperbolic equation (2.1).

2.3. Second-order BGK numerical flux. For scheme (2.7) with (2.10), if the linear finite element method is used, then the accuracy is of second order on the spatial direction. In this case, for the numerical flux in (2.7), we need to solve Riemann problems when it is across the cell interface. To preserve the second-order accuracy in space and avoid using the Riemann solvers, we will construct second-order accurate numerical fluxes $H_{j+1/2}$ by using the relation between the Euler equations and the Boltzmann equation. In particular, we will construct a second-order BGK numerical flux.

In Boltzmann equation (1.1), due to the mass, momentum, and energy conservations in the process of particle collisions, the collision term satisfies the following compatibility condition:

$$\iint \frac{g-f}{\tau} \psi dv d\xi = 0. \tag{2.11}$$

Here the molecular equilibrium velocity distribution function g is taken for the Maxwell–Boltzmann distribution

$$g = \rho \left(\frac{\lambda}{\pi}\right)^{\frac{k+3}{2}} e^{-\lambda[(v-u)^2+\xi^2]}$$

and the moment vector function ψ,

$$\psi = (1, v_x, v_y, v_z, \tfrac{1}{2}(v^2+\xi^2))^T,$$

where $(v-u)^2=(v_x-u_x)^2+(v_y-u_y)^2+(v_z-u_z)^2$, $\xi^2=\xi_1^2+\cdots+\xi_K^2$, $\lambda=\rho/2p$, $K=2/(\gamma-1)-d$ is the internal freedom degree of particle, and d is the dimension of space. Equations (1.1) and (2.11) are the famous BGK model.

To construct a second-order BGK numerical flux, we need to reconstruct the initial data. In one dimension, for uniform partition, the conservation variables in $I_j=(x_{j-1/2},x_{j+1/2})$ and $U_j=(\rho_j,\rho_j u_j,E_j)$ can be reconstructed as follows:

$$\bar{U}_j(x)=U_j+L(s_{j+},s_{j-})(x-x_j)$$

by using the Van Leer limiter

$$L(s_{j+},s_{j-})=S(s_{j+},s_{j-})\frac{|s_{j+}||s_{j-}|}{|s_{j+}|+|s_{j-}|}$$

or the MUSCL limiter

$$L(s_{j+},s_{j-})=S(s_{j+},s_{j-})\min\left(\frac{1}{2}|s_{j+}+s_{j-}|,2|s_{j+}|,2|s_{j-}|\right), \tag{2.12}$$

where

$$S(s_{j+},s_{j-})=sign(s_{j+})+sign(s_{j-}),$$

$$s_{j+}=\frac{(U_{j+1}-U_j)}{\Delta x},\quad s_{j-}=\frac{(U_j-U_{j-1})}{\Delta x},$$

$sign(\cdot)$ being the sign function. In our numerical experiments (section 4), the Van Leer limiter was used.

From the above formula, the flow variables $\bar{U}_j(x)$ at the cell interface $x_{j-1/2}$ and $x_{j+1/2}$ of I_j are given by

$$\bar{U}_j(x_{j-1/2})=U_j-\frac{\Delta x}{2}L(s_{j+},s_{j-}),\quad \bar{U}_j(x_{j+1/2})=U_j+\frac{\Delta x}{2}L(s_{j+},s_{j-})$$

and $\bar{U}_j(x_j)=U_j$ at the cell center x_j.

For the BGK model (1.1) and (2.11), the equivalent integral solution of f at the cell interface $x_{j+1/2}$ and time t is [23]

$$f(x_{j+1/2},t,v,\xi)=\frac{1}{\tau}\int_0^t g(x',t',v,\xi)e^{\frac{-(t-t')}{\tau}}dt'+e^{\frac{-t}{\tau}}f_0(x_{j+1/2}-vt), \tag{2.13}$$

where $x'=x_{j+1/2}-v(t-t')$ is the trajectory of a particle motion, f_0 is the initial distribution function at the beginning of each time step, and g is the corresponding equilibrium function to the point $(x_{j+1/2},t^n)$. To obtain the solution f, both f_0 and g must be specified in the above equation.

For the second-order scheme, f_0 and g around the cell interface $x_{j+1/2}$ are given by

$$f_0=\begin{cases} g_l[1+a_l(x-x_{j+1/2})], & x<x_{j+1/2},\\ g_r[1+a_r(x-x_{j+1/2})], & x>x_{j+1/2}, \end{cases} \tag{2.14}$$

and

$$g=g_0[1+(1-H(x-x_{j+\frac{1}{2}}))\bar{a}_l(x-x_{j+\frac{1}{2}})+H(x-x_{j+\frac{1}{2}})\bar{a}_r(x-x_{j+\frac{1}{2}})+\bar{A}t], \tag{2.15}$$

where g_l and g_r are local Maxwellian distribution functions located to the left and to the right, g_0 is the local Maxwellian distribution function at the center of a cell interface, and $H(\mathrm{x})$ is a Heaviside function,

$$H(x) = \begin{cases} 0, & x < x_{j+1/2}, \\ 1, & x \geq x_{j+1/2}, \end{cases}$$

and $a_l, a_r, \bar{a}_l, \bar{a}_r$, and $\bar{A}$ depend on the particle velocity v and the internal variable ξ. Using the second-order Taylor expansion gives

$$\begin{aligned} a_l &= a_{1l} + a_{2l}v + a_{3l}\frac{1}{2}(v^2 + \xi^2), \\ a_r &= a_{1r} + a_{2r}v + a_{3r}\frac{1}{2}(v^2 + \xi^2), \\ \bar{a}_l &= \bar{a}_{1l} + \bar{a}_{2l}v + \bar{a}_{3l}\frac{1}{2}(v^2 + \xi^2), \\ \bar{a}_r &= \bar{a}_{1r} + \bar{a}_{2r}v + \bar{a}_{3r}\frac{1}{2}(v^2 + \xi^2), \\ \bar{A} &= \bar{A}_1 + \bar{A}_2 v + \bar{A}_3\frac{1}{2}(v^2 + \xi^2). \end{aligned}$$

Using the relationship between the macroscopic variables and the velocity distribution function

$$\int f_0 \psi dv d\xi = \bar{U}_j(x), \quad x_{j-1/2} \leq x \leq x_{j+1/2},$$

we have

$$\int \psi g_l dv d\xi = \begin{pmatrix} \bar{\rho}_j(x_{j+1/2}) \\ \bar{\rho}_j \bar{u}_j(x_{j+1/2}) \\ \bar{E}_j(x_{j+1/2}) \end{pmatrix}, \quad \int \psi g_r dv d\xi = \begin{pmatrix} \bar{\rho}_{j+1}(x_{j+1/2}) \\ \bar{\rho}_{j+1}\bar{u}_{j+1}(x_{j+1/2}) \\ \bar{E}_{j+1}(x_{j+1/2}) \end{pmatrix}, \tag{2.16}$$

$$\int \psi a_l g_l dv d\xi = \begin{pmatrix} \dfrac{\bar{\rho}_j(x_{j+1/2}) - \bar{\rho}_j(x_j)}{\Delta x^-} \\ \dfrac{\bar{\rho}_j \bar{u}_j(x_{j+1/2}) - \bar{\rho}_j \bar{u}_j(x_j)}{\Delta x^-} \\ \dfrac{\bar{E}_j(x_{j+1/2}) - \bar{E}_j(x_j)}{\Delta x^-} \end{pmatrix},$$

$$\int \psi a_r g_r dv d\xi = \begin{pmatrix} \dfrac{\bar{\rho}_{j+1}(x_{j+1}) - \bar{\rho}_{j+1}(x_{j+1/2})}{\Delta x^+} \\ \dfrac{\bar{\rho}_{j+1}\bar{u}_{j+1}(x_{j+1}) - \bar{\rho}_{j+1}\bar{u}_{j+1}(x_{j+1/2})}{\Delta x^+} \\ \dfrac{\bar{E}_{j+1}(x_{j+1}) - \bar{E}_{j+1}(x_{j+1/2})}{\Delta x^+} \end{pmatrix}, \tag{2.16$'$}$$

where $\Delta x^- = x_{j+1/2} - x_j$, $\Delta x^+ = x_{j+1} - x_{j+1/2}$. With the definition of the Maxwellian distribution functions

$$g_l = \rho_l \left(\frac{\lambda_l}{\pi}\right)^{\frac{K+1}{2}} e^{-\lambda_l[(v-u_l)^2+\xi^2]}, \quad g_r = \rho_r \left(\frac{\lambda_r}{\pi}\right)^{\frac{K+1}{2}} e^{-\lambda_r[(v-u_r)^2+\xi^2]}$$

and from equations (2.16), we obtain ρ, u, λ in both equilibrium states g_l and g_r, respectively:

$$\begin{pmatrix} \rho_l \\ u_l \\ \lambda_l \end{pmatrix} = \begin{pmatrix} \bar{\rho}_j(x_{j+1/2}) \\ \bar{u}_j(x_{j+1/2}) \\ \dfrac{(K+1)\bar{\rho}_j(x_{j+1/2})}{4(\bar{E}_j(x_{j+1/2}) - \frac{1}{2}\bar{\rho}_j\bar{u}_j^2(x_{j+1/2}))} \end{pmatrix},$$

$$\begin{pmatrix} \rho_r \\ u_r \\ \lambda_r \end{pmatrix} = \begin{pmatrix} \bar{\rho}_{j+1}(x_{j+1/2}) \\ \bar{u}_{j+1}(x_{j+1/2}) \\ \dfrac{(K+1)\bar{\rho}_{j+1}(x_{j+1/2})}{4(\bar{E}_{j+1}(x_{j+1/2}) - \frac{1}{2}\bar{\rho}_{j+1}\bar{u}_{j+1}^2(x_{j+1/2}))} \end{pmatrix}.$$

It follows from (2.16)′ that

$$\frac{1}{\rho_l}\begin{pmatrix} \dfrac{\bar{\rho}_j(x_{j+1/2}) - \bar{\rho}_j(x_j)}{\Delta x^-} \\ \dfrac{\bar{\rho}_j\bar{u}_j(x_{j+1/2}) - \bar{\rho}_j\bar{u}_j(x_j)}{\Delta x^-} \\ \dfrac{\bar{E}_j(x_{j+1/2}) - \bar{E}_j(x_j)}{\Delta x^-} \end{pmatrix} \equiv \frac{1}{\rho_l}\begin{pmatrix} \left(\dfrac{\partial \rho}{\partial x}\right)^l \\ \left(\dfrac{\partial \rho u}{\partial x}\right)^l \\ \left(\dfrac{\partial E}{\partial x}\right)^l \end{pmatrix} = M^l_{\alpha\beta}\begin{pmatrix} a_{1l} \\ a_{2l} \\ a_{3l} \end{pmatrix},$$

$$\frac{1}{\rho_r}\begin{pmatrix} \dfrac{\bar{\rho}_{j+1}(x_{j+1}) - \bar{\rho}_{j+1}(x_{j+1/2})}{\Delta x^+} \\ \dfrac{\bar{\rho}_{j+1}\bar{u}_{j+1}(x_{j+1}) - \bar{\rho}_{j+1}\bar{u}_{j+1}(x_{j+1/2})}{\Delta x^+} \\ \dfrac{\bar{E}_{j+1}(x_{j+1}) - \bar{E}_{j+1}(x_{j+1/2})}{\Delta x^+} \end{pmatrix} \equiv \frac{1}{\rho_r}\begin{pmatrix} \left(\dfrac{\partial \rho}{\partial x}\right)^r \\ \left(\dfrac{\partial \rho u}{\partial x}\right)^r \\ \left(\dfrac{\partial E}{\partial x}\right)^r \end{pmatrix} = M^r_{\alpha\beta}\begin{pmatrix} a_{1r} \\ a_{2r} \\ a_{3r} \end{pmatrix},$$

(2.17)

where the symmetric matrices $M^l_{\alpha\beta}$ and $M^r_{\alpha\beta}$ are given by

$$M^l_{\alpha\beta} = \begin{pmatrix} 1 & u_l & \dfrac{1}{2}\left(u_l^2 + \dfrac{K+1}{2\lambda_l}\right) \\ u_l & u_l^2 + \dfrac{1}{2\lambda_l} & \dfrac{1}{2}\left(u_l^3 + \dfrac{(K+3)u_l}{2\lambda_l}\right) \\ \dfrac{1}{2}\left(u_l^2 + \dfrac{K+1}{2\lambda_l}\right) & \dfrac{1}{2}\left(u_l^3 + \dfrac{(K+3)u_l}{2\lambda_l}\right) & \dfrac{1}{2}\left(u_l^4 + \dfrac{(K+3)u_l^2}{\lambda_l} + \dfrac{(K^2+4K+3)}{4\lambda_l^2}\right) \end{pmatrix},$$

$$M^r_{\alpha\beta} = \begin{pmatrix} 1 & u_r & \dfrac{1}{2}\left(u_r^2 + \dfrac{K+1}{2\lambda_r}\right) \\ u_r & u_r^2 + \dfrac{1}{2\lambda_r} & \dfrac{1}{2}\left(u_r^3 + \dfrac{(K+3)u_r}{2\lambda_r}\right) \\ \dfrac{1}{2}\left(u_r^2 + \dfrac{K+1}{2\lambda_r}\right) & \dfrac{1}{2}\left(u_r^3 + \dfrac{(K+3)u_r}{2\lambda_r}\right) & \dfrac{1}{2}\left(u_r^4 + \dfrac{(K+3)u_r^2}{\lambda_r} + \dfrac{(K^2+4K+3)}{4\lambda_r^2}\right) \end{pmatrix}.$$

(2.18)

Using (2.17) and (2.18) gives

$$
\begin{aligned}
a_{3m} &= \frac{4\lambda_m^2}{K+1}\left(2\left(\frac{\partial \varepsilon}{\partial x}\right)^m - 2u_m\left(\frac{\partial u}{\partial x}\right)^m\right),\\
a_{2m} &= 2\lambda_m\left(\left(\frac{\partial u}{\partial x}\right)^m - \frac{u_m}{2\lambda_m}a_{3m}\right),\\
a_{1m} &= \frac{1}{\rho_m}\left(\frac{\partial \rho}{\partial x}\right)^m - a_{2m}u_m - a_{3m}\left(\frac{u_m^2}{2} + \frac{K+1}{4\lambda_m}\right),
\end{aligned}
\tag{2.19}
$$

where

$$
\begin{aligned}
\left(\frac{\partial u}{\partial x}\right)^m &= \frac{1}{\rho_m}\left(\left(\frac{\partial(\rho u)}{\partial x}\right)^m - u_m\left(\frac{\partial \rho}{\partial x}\right)^m\right),\\
\left(\frac{\partial \varepsilon}{\partial x}\right)^m &= \frac{1}{\rho_m}\left(\left(\frac{\partial E}{\partial x}\right)^m - \frac{1}{2}\left(u_m^2 + \frac{K+1}{2\lambda_m}\right)\left(\frac{\partial \rho}{\partial x}\right)^m\right), \quad m = l, r.
\end{aligned}
$$

Thus the initial distribution function after reconstruction is obtained. To determine g_0, we take the limit $t \to 0$ in (2.13) and substitute the result into (2.11) to have

$$
\iint g_0 \psi dv d\xi = \iint_{v>0} g_l \psi dv d\xi + \iint_{v<0} g_r \psi dv d\xi. \tag{2.20}
$$

Since

$$
\lambda_0 = \frac{(K+1)\rho_0}{4(E_0 - \frac{1}{2}\rho_0 u_0^2)},
$$

we need only to know $(\rho_0, \rho_0 u_0, E_0)^T$, which can be expressed as moments of g_l and g_r. From (2.20), we have

$$
\begin{pmatrix} \rho_0 \\ \rho_0 u_0 \\ E_0 \end{pmatrix} = \begin{pmatrix} \rho_l \langle v^0 \rangle_{v>0} + \rho_r \langle v^0 \rangle_{v<0} \\ \rho_l \langle v^1 \rangle_{v>0} + \rho_r \langle v^1 \rangle_{v<0} \\ \frac{1}{2}\rho_l \langle v^2 + \xi^2 \rangle_{v>0} + \frac{1}{2}\rho_r \langle v^2 + \xi^2 \rangle_{v<0} \end{pmatrix},
$$

where

$$
\rho_l \langle \cdots \rangle_{v>0} = \iint_{v>0} (\cdots) g_l dv d\xi,
$$

$$
\rho_r \langle \cdots \rangle_{v<0} = \iint_{v<0} (\cdots) g_r dv d\xi.
$$

After obtaining g_0, $\bar{a}_l$ and $\bar{a}_r$ in (2.15) can be computed through the relation

$$
\frac{1}{\rho_0}\begin{pmatrix} \dfrac{\bar{\rho}_{j+1}(x_{j+1}) - \rho_0}{\Delta x^+} \\ \dfrac{\bar{\rho}_{j+1}\bar{u}_{j+1}(x_{j+1}) - \rho_0 u_0}{\Delta x^+} \\ \dfrac{\bar{E}_{j+1}(x_{j+1}) - E_0}{\Delta x^+} \end{pmatrix} = \bar{M}^0_{\alpha\beta}\begin{pmatrix} \bar{a}_{1r} \\ \bar{a}_{2r} \\ \bar{a}_{3r} \end{pmatrix} \tag{2.21}
$$

and

$$\frac{1}{\rho_0}\begin{pmatrix} \dfrac{\rho_0-\bar{\rho}_j(x_j)}{\Delta x^-} \\ \dfrac{\rho_0 u_0-\bar{\rho}_j\bar{u}_j(x_j)}{\Delta x^-} \\ \dfrac{E_0-\bar{E}_j(x_j)}{\Delta x^-} \end{pmatrix} = \bar{M}^0_{\alpha\beta}\begin{pmatrix} \bar{a}_{1l} \\ \bar{a}_{2l} \\ \bar{a}_{3l} \end{pmatrix}, \tag{2.21}'$$

where the matrix $\bar{M}^0_{\alpha\beta}=\frac{1}{\rho_0}\int g_0\psi_\alpha\psi_\beta dvd\xi$ has the same form as $M^r_{\alpha\beta}$, that is,

$$\bar{M}^0_{\alpha\beta}=\begin{pmatrix} 1 & u_0 & \frac{1}{2}\left(u_0^2+\frac{K+1}{2\lambda_0}\right) \\ u_0 & u_0^2+\frac{1}{2\lambda_0} & \frac{1}{2}\left(u_0^3+\frac{(K+3)u_0}{2\lambda_0}\right) \\ \frac{1}{2}\left(u_0^2+\frac{K+1}{2\lambda_0}\right) & \frac{1}{2}\left(u_0^3+\frac{(K+3)u_0}{2\lambda_0}\right) & \frac{1}{2}\left(u_0^4+\frac{(K+3)u_0^2}{\lambda_0}+\frac{(K^2+4K+3)}{4\lambda_0^2}\right) \end{pmatrix}.$$

Therefore, $(\bar{a}_{1l},\bar{a}_{2l},\bar{a}_{3l})^T$ and $(\bar{a}_{1r},\bar{a}_{2r},\bar{a}_{3r})^T$ can be found through the procedure similar to (2.19).

Substituting (2.14) and (2.15) into (2.13), the final gas distribution function at a cell interface can be written as

$$\begin{aligned} f(x_{j+1/2},t,v,\xi) &= (1-e^{-t\tau})g_0 \\ &\quad +(\tau(-1+e^{-t/\tau})+te^{-t/\tau})(\bar{a}_lH(v)+\bar{a}_r(1-H(v)))vg_0 \\ &\quad +\tau(t/\tau-1+e^{-t/\tau})\bar{A}g_0 \\ &\quad +e^{-t/\tau}((1-vta_l)H(v)g_l+(1-vta_r)(1-H(v))g_r). \end{aligned} \tag{2.22}$$

To determine the only unknown term $\bar{A}$ in the above expression, using the relation (2.11) at the cell interface $x_{j+1/2}$ and integrating it over the whole time step Δt, we have

$$\int_0^{\Delta t}\int((g_0+\bar{A}g_0t)-f)\psi dvd\xi dt=0.$$

Substituting (2.22) into the above equation gives

$$\int_0^{\Delta t}\int(\tau-\tau e^{-t/\tau})\bar{A}g_0\psi dvd\xi dt=\int_0^{\Delta t}\int\eta\psi dvd\xi dt,$$

where

$$\begin{aligned} \eta &= -e^{-t/\tau}g_0 \\ &\quad +(\tau(-1+e^{-t/\tau})+te^{-t/\tau})(\bar{a}_lH(v)+\bar{a}_r(1-H(v)))vg_0 \\ &\quad +e^{-t/\tau}((1-vta_l)H(v)g_l+(1-vta_r)(1-H(v))g_r). \end{aligned}$$

With the integrating relations

$$\int_0^{\Delta t}(\tau-\tau e^{-t/\tau})dt=\tau(\Delta t-\tau(1-e^{-\Delta t/\tau})),$$
$$\int_0^{\Delta t}e^{-t/\tau}dt=\tau(1-e^{-\Delta t/\tau}),$$
$$\int_0^{\Delta t}(\tau(-1+e^{-t/\tau})+te^{-t/\tau})dt=-\tau\Delta t+2\tau^2(1-e^{-\Delta t/\tau})-\tau\Delta te^{-\Delta t/\tau},$$
$$\int_0^{\Delta t}te^{-t/\tau}dt=-\tau\Delta te^{-\Delta t/\tau}+\tau^2(1-e^{-\Delta t/\tau}),$$

we obtain

$$\begin{aligned}\int\bar{A}g_0\psi dvd\xi dt=\int&[\gamma_1 g_0+\gamma_2 v(\bar{a}_l H(v)+\bar{a}_r(1-H(v)))g_0\\&+\gamma_3(H(v)g_l+(1-H(v))g_r)\\&+\gamma_4 v(a_l H(v)g_l+a_r(1-H(v))g_r)]\psi dvd\xi,\end{aligned}\tag{2.23}$$

where

$$\begin{aligned}\gamma_0&=\Delta t-\tau(1-e^{-\Delta t/\tau}),\\\gamma_1&=-(1-e^{-\Delta t/\tau})/\gamma_0,\\\gamma_2&=(-\Delta t+2\tau(1-e^{-\Delta t/\tau})-\Delta te^{-\Delta t/\tau})/\gamma_0,\\\gamma_3&=(1-e^{-\Delta t/\tau})/\gamma_0,\\\gamma_4&=(\Delta te^{-\Delta t/\tau}-\tau(1-e^{-\Delta t/\tau}))/\gamma_0.\end{aligned}$$

Equation (2.23) gives $(\bar{A}_1,\bar{A}_2,\bar{A}_3)$ by following the same procedure as evaluating $\bar{a}_l$.

Up to this point, we have obtained the distribution function $f(x_{j+1/2},t,v,\xi)$ at the cell interface $x_{j+1/2}$, and the numerical fluxes can subsequently be obtained as follows:

$$\begin{pmatrix}H_\rho\\H_{\rho u}\\H_E\end{pmatrix}_{j+1/2}=\int v\begin{pmatrix}1\\v\\\frac{1}{2}(v^2+\xi^2)\end{pmatrix}f(x_{j+1/2},t,v,\xi)dvd\xi.\tag{2.24}$$

Combining formulas (2.7), (2.24), and (2.10), we have obtained a second-order RKDG finite element method which is Riemann solver-free.

3. Second-order scheme in two dimensions. We consider 2D hyperbolic conservation laws:

$$\frac{\partial U}{\partial t}+\frac{\partial F(U)}{\partial x}+\frac{\partial F(U)}{\partial y}=0\quad\forall(x,y)\in\Omega,\quad t\in(0,T),\tag{3.1}$$

where $U=(\rho,\rho u,\rho v,E)^T$, $F(U)=(\rho u,\rho u^2+p,\rho uv,u(E+p))^T$, $G(U)=(\rho v,\rho uv,\rho v^2+p,v(E+p))^T$, ρ is the flow density, u and v are velocity components in the x- and y-directions, respectively, E is the total energy, and p is the flow pressure.

3.1. The DG finite element method space discretization. Assume Γ_h is a partition of the domain Ω. Let $K \in \Gamma_h$ and $V(K)$ be a finite element space on K. In practice, $V(K)$ is often taken as mth ($m \geq 0$) order polynomial space. Given $t \in [0, T]$, we look for the approximate solution $U_h(t)$ in the finite element space

$$V_h = \{v_h \in L^\infty(\Omega) : v_h|_K \in V(K) \ \forall K \in \Gamma_h\}.$$

For any smooth function $v(x)$, we have

(3.2)
$$\frac{d}{dt}\int_K U(x,t)v(x)dx + \sum_{e\in\partial K}\int_e \vec{f}(U(x,t))\cdot n_{e,K}v(x)d\Gamma - \int_K \vec{f}(U(x,t))\cdot\nabla v(x)dx = 0,$$

where $n_{e,K}$ denotes the unit outward normal of the edge e, $\vec{f}(U) = (F(U), G(U))$, and $x = (x, y)$. The following numerical quadrature rules will be employed to approximate the integrals in (3.2):

$$\int_e \vec{f}(U(x,t))\cdot n_{e,K}v(x)d\Gamma \approx \sum_{l=1}^{L}\omega_l\vec{f}(U(x_{el},t))\cdot n_{e,K}v(x_{el})\,|e|, \tag{3.3}$$

$$\int_K \vec{f}(U(x,t))\cdot\nabla v(x)dx \approx \sum_{j=1}^{M}\underline{\omega}_j\vec{f}(U(x_{Kj},t))\cdot\nabla v(x_{Kj})\,|K|. \tag{3.4}$$

Replacing the exact solution U by U_h, the smooth function v by $v_h \in V(K)$, and the flux $\vec{f}(U(x,t))\cdot n_{e,K}$ in the boundary integrals by the numerical flux $H_{e,K}(x,t)$, we obtain

$$\begin{aligned}&\frac{d}{dt}\int_K U_h(x,t)v_h(x)dx + \sum_{e\in\partial K}\sum_{l=1}^{L}\omega_l H_{e,K}(x_{el},t)v_h(x_{el})|e|\\ &-\sum_{j=1}^{M}\underline{\omega}\vec{f}(U_h(x_{kj},t))\cdot\nabla v_h(x_{kj})|K| \ \forall v_h \in V(K),\ \forall K \in \Gamma_h,\end{aligned} \tag{3.5}$$

$$\int_K U_h(x,0)v_h dx = \int_K U_0(x)v_h dx,$$

where

$$H_{e,K} = H_{e,K}(U_h(x^{int(K)},t), U_h(x^{ext(K)},t)),$$

$$U_h(x^{int(K)},t)) = \lim_{y\to x, y\in K} U_h(y,t),$$

$$U_h(x^{ext(K)},t) = \begin{cases}\gamma_h(x,t), & x \in \partial\Omega,\\ \lim\limits_{y\to x, y\notin K} U_h(y,t) & \text{otherwise.}\end{cases}$$

Above, $\gamma_h(x,t)$ is the trace function and $H_{e,K}$ satisfies the following conditions:

(1) $H_{e,K}(U,U)$ is locally Lipschitz continuous and is consistent with $\vec{f}(U)\cdot n_{e,K}$ in the following sense:

$$H_{e,K}(U,U)=\vec{f}(U)\cdot n_{e,K}.$$

(2) $H_{e,K}(U,U)$ is a nondecreasing function of its first argument and a nonincreasing function of its second argument.

(3) $H_{e,K}(U,U)$ satisfies the conservation laws in the following sense:

$$H_{e,K}(U_h(x^{int(K)},t),U_h(x^{ext(K)},t))+H_{e,K}(U_h(x^{int(K')},t),U_h(x^{ext(K')},t))=0, \tag{3.6}$$

$$K\cap K'=e.$$

3.2. The RK time discretization. The semidiscrete scheme (3.5) yields the following ODE system:

$$\frac{d}{dt}U_h=L_h(U_h,\gamma_h), \tag{3.7}$$

which is solved by using the RK method similar to that described in the 1D case; see also [7]. Therefore, the RKDG finite element scheme is obtained by combining (3.5) with (2.10).

3.3. Second-order BGK numerical flux. Again, to avoid solving Riemann problems, we use the second-order BGK method to construct the numerical fluxes $H_{i+1/2,j}$ and $H_{i,j+1/2}$ in (3.5). More precisely, consider the 2D element $K=[x_{i-1/2}, x_{i+1/2}]\times[y_{j-1/2},y_{j+1/2}]$, where (i,j) denotes the cell K with the boundary $i\pm 1/2$ and $j\pm 1/2$, $(x_{i+1/2,j},y_{i+1/2,j})$ denotes the center of the cell boundary between cells (i,j) and $(i+1,j)$. For simplicity, $(x_{i+1/2,j}=0,y_{i+1/2,j}=0)$ is assumed.

Consider the 2D BGK model

$$f_t+uf_x+vf_y=\frac{g-f}{\tau}, \tag{3.8}$$

$$\iint\frac{g-f}{\tau}\psi_\alpha dudvd\xi=0,\quad \alpha=1,2,3,4, \tag{3.9}$$

where $\psi=(\psi_1,\psi_2,\psi_3,\psi_4)=(1,u,v,\frac{1}{2}(u^2+v^2+\xi^2))^T$, which gives the integral solution of f at $(x_{i+1/2,j},y_{i+1/2,j})$:

$$\begin{aligned}f(x_{i+1/2,j},y_{i+1/2,j},t,u,v,\xi)=&\frac{1}{\tau}\int_0^t g(x',y',t',u,\xi)e^{\frac{-(t-t')}{\tau}}dt'\\&+e^{\frac{-t}{\tau}}f_0(x_{i+1/2,j}-ut,y_{i+1/2,j}-vt),\end{aligned} \tag{3.10}$$

where $x'=x_{i+1/2,j}-u(t-t')$ and $y'=y_{i+1/2,j}-v(t-t')$ are the trajectory of a particle motion, f_0 is the initial distribution function, and g is the equilibrium state around the cell interface $(x_{i+1/2,j},y_{i+1/2,j})$. With the definition $(x_{i+1/2,j}=0,y_{i+1/2,j}=0)$, f goes to

$$f(0,0,t,u,v,\xi)=\frac{1}{\tau}\int_0^t g(x',y',t',u,\xi)e^{\frac{-(t-t')}{\tau}}dt'+e^{\frac{-t}{\tau}}f_0(-ut,-vt). \tag{3.11}$$

To obtain the solution f, both f_0 and g must be specified in (3.11).

Similar to the 1D case, the reconstructed initial data in the cell (i,j) at the beginning of each time step can be expressed as

$$\bar{U}_{i,j}(x,y) = U_{i,j} + L_x(s^+_{i,j}, s^-_{i,j})(x - x_{i,j}) + L_y(\delta^+_{i,j}, \delta^-_{i,j})(y - y_{i,j}),$$

where $U_{i,j}$ is the cell averaged initial data and $L_x(s_1, s_2)$, $L_y(\delta_1, \delta_2)$ are the nonlinear limiters,

$$s^+_{i,j} = (U_{i+1,j} - U_{i,j})/(x_{i+1,j} - x_{i,j}), \quad s^-_{i,j} = (U_{i,j} - U_{i-1,j})/(x_{i,j} - x_{i-1,j}),$$
$$\delta^+_{i,j} = (U_{i,j+1} - U_{i,j})/(x_{i,j+1} - x_{i,j}), \quad \delta^-_{i,j} = (U_{i,j} - U_{i,j-1})/(x_{i,j} - x_{i,j-1}).$$

For the second-order scheme, f_0 and g after the initial data reconstruction are given by

$$\begin{aligned} f_0(x,y,0) &= \begin{cases} g_l[1 + a_l x + b_l y], & x < 0, \\ g_r[1 + a_r x + b_r y], & x > 0, \end{cases} \\ &= g_l(1 + a_l x + b_l y)(1 - H(x)) + g_r(1 + a_r x + b_r y)H(x), \end{aligned} \tag{3.12}$$

and

$$g(x,y,t) = g_0[1 + (1 - H(x))\bar{a}_l x + H(x)\bar{a}_r x + \bar{b}y + \bar{A}t], \tag{3.13}$$

where

$$\begin{aligned} g_l &= \rho_l\left(\tfrac{\lambda_l}{\pi}\right)^{\frac{k+2}{2}} e^{-\lambda_l[(u-v_l)^2 + (v-v_l)^2 + \xi^2]}, \\ g_r &= \rho_r\left(\tfrac{\lambda_r}{\pi}\right)^{\frac{k+2}{2}} e^{-\lambda_r[(u-v_r)^2 + (v-v_r)^2 + \xi^2]}, \\ g_0 &= \rho_0\left(\tfrac{\lambda_0}{\pi}\right)^{\frac{k+2}{2}} e^{-\lambda_0[(u-v_0)^2 + (v-v_0)^2 + \xi^2]}, \end{aligned}$$

$$H(x) = \begin{cases} 0, & x < 0, \\ 1, & x \geq 0. \end{cases} \tag{3.14}$$

The parameters a_l, a_r, b_l, and b_r in (3.12) and $\bar{a}_l$, $\bar{a}_r$, $\bar{b}$, and $\bar{A}$ in (3.13) depend on the particle velocity (u,v) and the internal variable ξ. By the second-order Taylor expansion, we have

$$\begin{aligned} a_l &= a_{1l} + a_{2l}u + a_{3l}v + a_{4l}\tfrac{1}{2}(u^2 + v^2 + \xi^2), \\ a_r &= a_{1r} + a_{2r}u + a_{3r}v + a_{4r}\tfrac{1}{2}(u^2 + v^2 + \xi^2), \\ b_l &= b_{1l} + b_{2l}u + b_{3l}v + b_{4l}\tfrac{1}{2}(u^2 + v^2 + \xi^2), \\ b_r &= b_{1r} + b_{2r}u + b_{3r}v + b_{4r}\tfrac{1}{2}(u^2 + v^2 + \xi^2), \\ \bar{a}_l &= \bar{a}_{1l} + \bar{a}_{2l}u + \bar{a}_{3l}v + \bar{a}_{4l}\tfrac{1}{2}(u^2 + v^2 + \xi^2), \\ \bar{a}_r &= \bar{a}_{1r} + \bar{a}_{2r}u + \bar{a}_{3r}v + \bar{a}_{4r}\tfrac{1}{2}(u^2 + v^2 + \xi^2), \\ \bar{b} &= \bar{b}_1 + \bar{b}_2 u + \bar{b}_3 v + \bar{b}_4\tfrac{1}{2}(u^2 + v^2 + \xi^2), \\ \bar{A} &= \bar{A}_1 + \bar{A}_2 u + \bar{A}_3 v + \bar{A}_4\tfrac{1}{2}(u^2 + v^2 + \xi^2). \end{aligned} \tag{3.15}$$

Similar to (2.19), a_l, a_r, b_l, b_r can be computed. Thus the initial distribution function f_0 is obtained. To determine g_0, we take the limit $t \to 0$ in (3.10); then g_0 can be obtained from g_l and g_r of f_0 by using the compatibility condition

$$\int g_0 \psi du dv d\xi = \iint_{u>0} \psi g_l du dv d\xi + \iint_{u<0} \psi g_r du dv d\xi.$$

More specifically, we have

$$\begin{aligned} U_0 = \begin{pmatrix} \rho_0 \\ \rho_0 u_0 \\ \rho_0 v_0 \\ E_0 \end{pmatrix} &= \rho_l \begin{pmatrix} \langle u^0 v^0 \rangle_{u>0} \\ \langle u^1 v^0 \rangle_{u>0} \\ \langle u^0 v^1 \rangle_{u>0} \\ \frac{1}{2}(\langle u^2 v^0 \rangle_{u>0} + \langle u^0 v^2 \rangle_{u>0} + \langle u^0 v^0 \xi^2 \rangle_{u>0}) \end{pmatrix} \\ &+ \rho_r \begin{pmatrix} \langle u^0 v^0 \rangle_{u<0} \\ \langle u^1 v^0 \rangle_{u<0} \\ \langle u^0 v^1 \rangle_{u<0} \\ \frac{1}{2}(\langle u^2 v^0 \rangle_{u<0} + \langle u^0 v^2 \rangle_{u<0} + \langle u^0 v^0 \xi^2 \rangle_{u<0}) \end{pmatrix}, \end{aligned} \tag{3.16}$$

$$\lambda_0 = \frac{(K+2)\rho_0}{4(E_0 - \frac{1}{2}\rho_0(u_0^2 + v_0^2))}. \tag{3.17}$$

Up to this point, g_0 is totally determined. $\bar{a}_l$ and $\bar{a}_r$ terms in (3.13) can be obtained by using the similar techniques (2.21) and (2.21)′. The term $\bar{b}$ in (3.13) can be obtained from the following compatibility condition:

$$\frac{\partial}{\partial y} \int (g - f_0) \psi du dv d\xi = 0,$$

which gives

$$\frac{1}{\rho_0} \int_{-\infty}^{\infty} \psi \bar{b} g_0 du dv d\xi = M^0_{\alpha\beta} \begin{pmatrix} \bar{b}_1 \\ \bar{b}_2 \\ \bar{b}_3 \\ \bar{b}_4 \end{pmatrix} = \frac{1}{\rho_0} \Big(\iint_{u>0} \psi b_l g_l du dv d\xi + \iint_{u<0} \psi b_r g_r du dv d\xi \Big), \tag{3.18}$$

where the matrix

$$M^0_{\alpha\beta} = \frac{1}{\rho_0} \int \psi_\alpha \psi_\beta g_0 du dv d\xi$$

has the same form as the matrix in section 2. So, $(\bar{b}_1, \bar{b}_2, \bar{b}_3, \bar{b}_4)$ can be obtained by using similar methods of obtaining (a_{1m}, a_{2m}, a_{3m}) in (2.19).

Substituting both the expression (3.12) of f_0 and the expression (3.13) of g into the integral solution (3.11) of the BGK model, we have

$$\begin{aligned} f(0,0,t,u,v,\xi) &= (1-e^{-t/\tau})g_0 \\ &\quad + (\tau(-1+e^{-t/\tau}) + te^{-t/\tau})((\bar{a}_l H(u) + \bar{a}_r(1-H(u)))u + \bar{b}v)g_0 \\ &\quad + \tau(t/\tau - 1 + e^{-t/\tau})\bar{A}g_0 \\ &\quad + e^{-t/\tau}((1-uta_l - vtb_l)H(u)g_l + (1-uta_r - vtb_r)(1-H(u))g_r). \end{aligned} \tag{3.19}$$

To determine the only unknown term $\bar{A}$ in the above expression, using the compatibility condition at the point $(x=0, y=0)$ and integrating it over the whole time step Δt, we have

$$\int_0^{\Delta t} \int \begin{pmatrix} 1 \\ u \\ v \\ \frac{1}{2}(u^2+v^2+\xi^2) \end{pmatrix} (g(0,0,t) - f(0,0,t,u,v,\xi))dudvd\xi dt = 0, \tag{3.20}$$

which gives

$$\begin{aligned} M^0_{\alpha\beta}(\bar{A}_1,\bar{A}_2,\bar{A}_3,\bar{A}_4)^T &= \frac{1}{\rho_0}\int [\gamma_1 g_0 + \gamma_2(u(\bar{a}_l H(u) + \bar{a}_r(1-H(u))) + \bar{b}v)g_0 \\ &\quad + \gamma_3(H(u)g_l + (1-H(u))g_r) \\ &\quad + \gamma_4((a_l u + b_l v)H(u)g_l + (a_r u + b_r v)(1-H(u))g_r)]\psi dudvd\xi, \end{aligned} \tag{3.21}$$

where the parameters $\gamma_1, \gamma_2, \gamma_3, \gamma_4$ have the same values as those in the 1D case. Then $\bar{A} = (\bar{A}_1, \bar{A}_2, \bar{A}_3, \bar{A}_4)$ can be obtained from (3.21).

Up to this point, we have obtained the explicit distribution function $f(0,0,t,u,v,\xi)$ in (3.19), and the numerical fluxes across the cell interface between cells (i,j) and $(i+1,j)$ can subsequently be obtained as

$$H_{i+1/2,j} = \begin{pmatrix} H_\rho \\ H_{\rho u} \\ H_{\rho v} \\ H_E \end{pmatrix}_{i+1/2,j} = \int u \begin{pmatrix} 1 \\ u \\ v \\ \frac{1}{2}(u^2+v^2+\xi^2) \end{pmatrix} f(0,0,t,u,v,\xi)dudvd\xi. \tag{3.22}$$

Similar techniques can be developed to construct the numerical fluxes $H_{i,j+1/2}$ across the cell interface between cells (i,j) and $(i,j+1)$. Replacing the numerical fluxes in (3.5) by formula (3.22) and combining the resulting equation with (2.10), we obtained a second-order RKDG finite element method which is Riemann solver-free.

4. Numerical experiment. In this section, we apply the methods developed in section 2 and section 3 to some benchmark problems. Among them Examples 1–4 are 1D problems and Examples 5–6 are 2D ones.

4.1. 1D examples. For 1D problems, the domain $[-0.5, 0.5]$ is divided by 100 cells and 200 cells, respectively, and the CFL number is taken to be 0.1.

Example 1. The first example is the standard Sod test problem, which uses the following initial conditions:

$$(\rho, u, p)_L = (1.0, 0.0, 1.0), \quad (\rho, u, p)_R = (0.125, 0.0, 0.1).$$

The exact and approximate solutions are plotted in Figure 1. The problem was computed in [6] using the TVB RKDG method. It is found that our 100-cell results are more accurate than those obtained by using the componentwise limiter (Figures 1–3 in [6]) and are almost the same as the results obtained by using the characteristicwise limiter (Figures 7–9 in [6]). There is some instability in the back of the shock wave of the velocity. But, this instability does not spread when the meshes are refined.

Example 2. The second example is the Lax problem which uses the following initial conditions:

$$(\rho, u, p)_L = (0.445, 0.698, 3.528), \quad (\rho, u, p)_R = (0.5, 0.0, 0.571).$$

The results are displayed in Figure 2. Compared to the results in [6], 100-cell results are almost the same as those with second-order accuracy and more accurate than those with third-order accuracy obtained by using characteristicwise limiter.

Example 3. The third example uses the following initial conditions:

$$(\rho, u, p)_L = (5.99924, 19.5975, 460.894), \qquad (\rho, u, p)_R = (5.99242, -6.19633, 46.0950).$$

The results are plotted in Figure 3. We can find that there are some instabilities in the density, velocity, and pressure. It may be due to the BGK assumption made in section 1 that during a time dt, a fraction of dt/τ of molecules in a given small volume undergoes collision, which alters the velocity-distribution function from f to g. If the collision interval τ taken is not good, the instabilities appear in the back of the shock wave. Example 4 is similar.

Example 4. The fourth example uses the following initial conditions:

$$(\rho, u, p)_L = (1, 0, 1.0), \quad (\rho, u, p)_R = (0.125, 0, 0.2).$$

The results are displayed in Figure 4.

4.2. 2D examples. Two well-known 2D test problems, namely the double-Mach reflection problem and the forward-facing step problem, will be used to test the method proposed in section 3. Compared with [27], the numerical results showed the favorable shock resolution and computational accuracy of our methods.

Example 5 (The double-Mach reflection problem). The problem involves a Mach 10 shock that initially makes a 60° angle with a reflecting wall. The state ahead of the shock is $(\rho, u, v, p) = (1.4, 0, 0, 1)$. The initial value for numerical calculation is taken to be the preshock condition and the postshock condition, where the postshock condition is obtained by the following Hoguniot formula of the positive shock:

$$\frac{u - u_0}{C_0} = \frac{2}{\gamma + 1}\left(M - \frac{1}{M}\right), \quad \frac{p}{p_0} = \frac{2\gamma}{\gamma + 1}\left(M^2 - \frac{\gamma - 1}{\gamma + 1}\right), \quad \frac{\rho}{\rho_0} = \frac{(\gamma + 1)M^2}{(\gamma - 1)M^2 + 2},$$

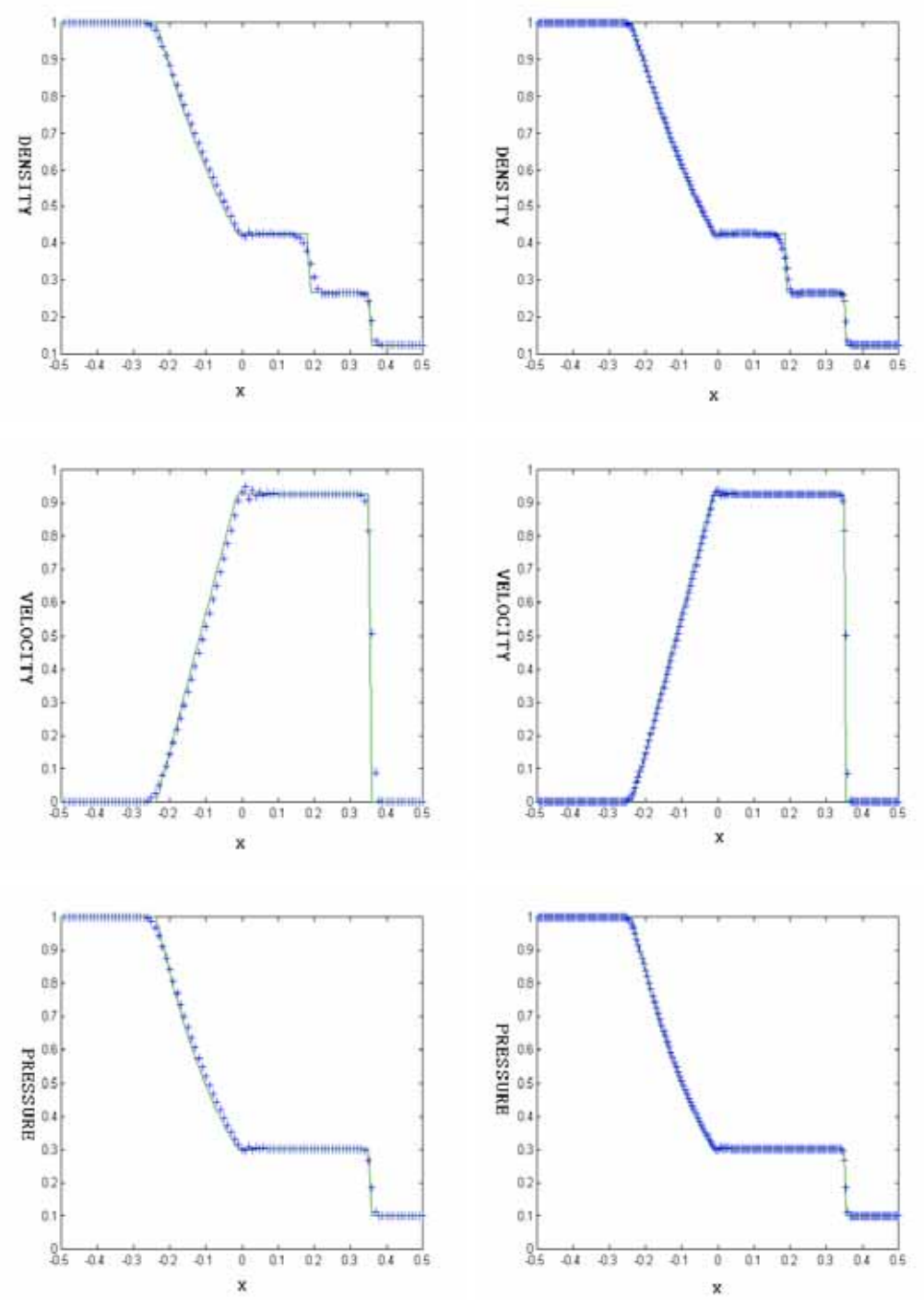


FIG. 1. *Example* 1. *Comparison between numerical solutions* (+) *and exact solutions* (*solid line*)*:* 100 *cells* (*left*) *and* 200 *cells* (*right*) *at* $t = 0.2$.

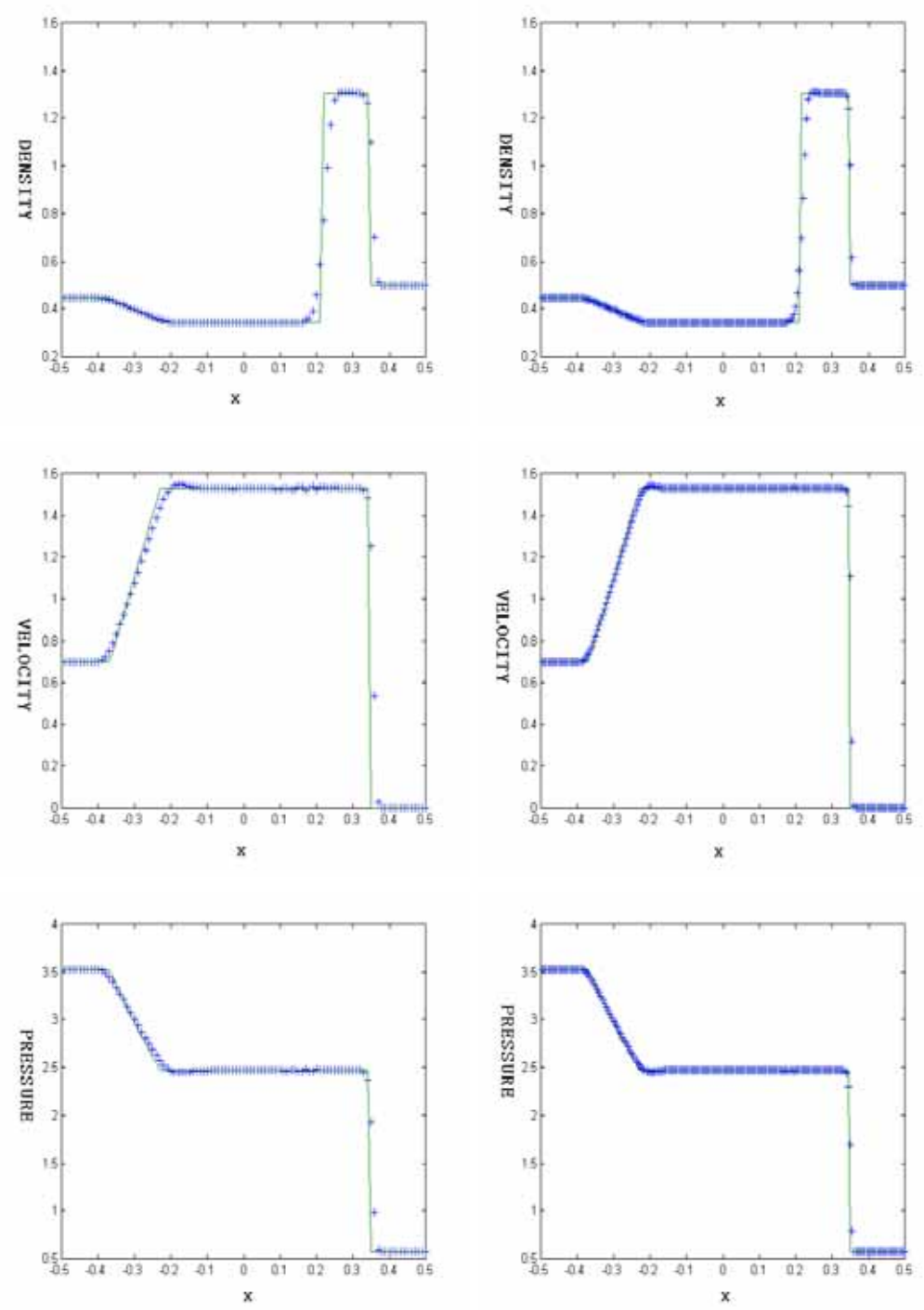


FIG. 2. *Example* 2. *Comparison between numerical solutions* (+) *and exact solutions* (*solid line*)*:* 100 *cells* (*left*) *and* 200 *cells* (*right*) *at* $t = 0.14$.

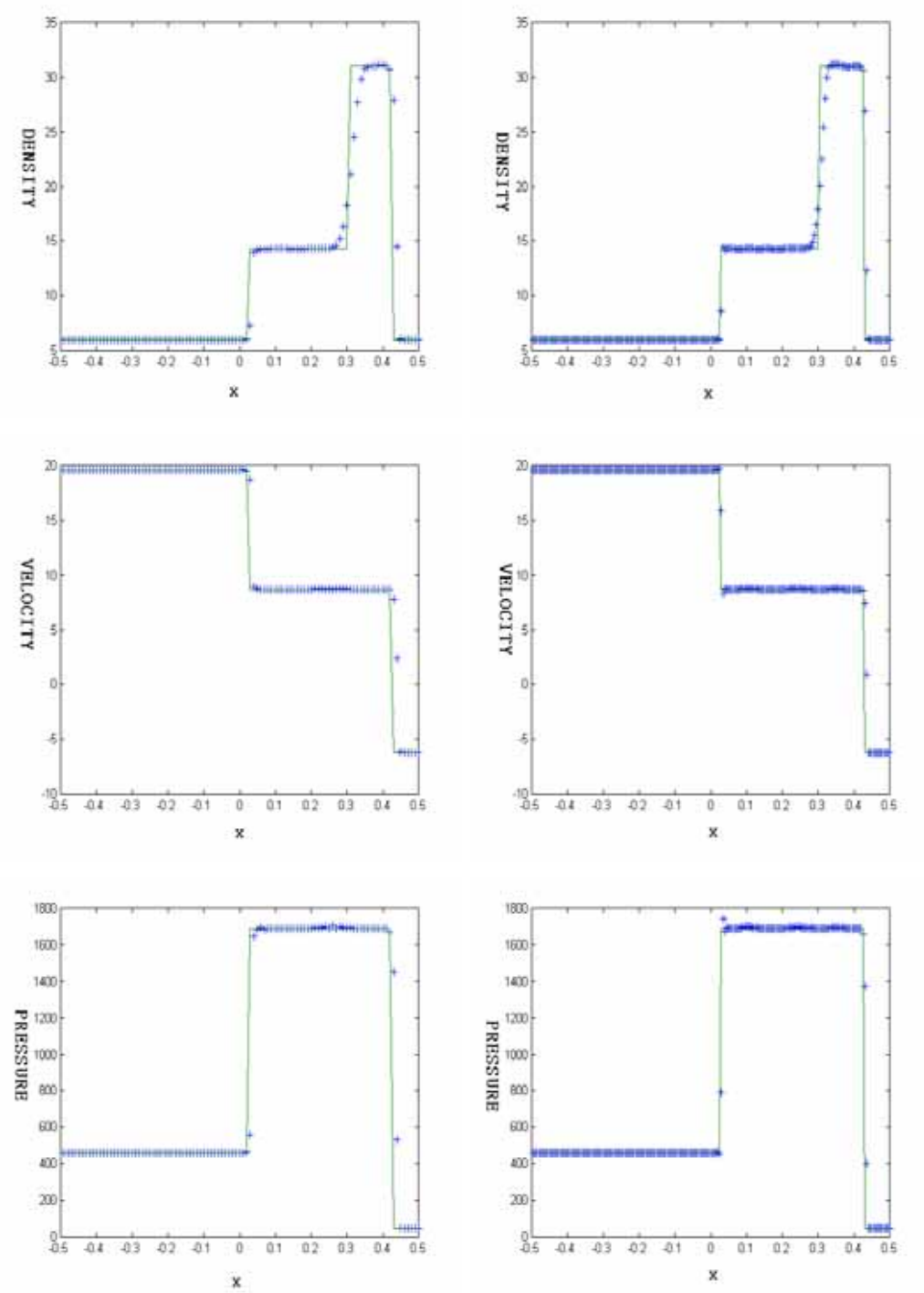


FIG. 3. *Example* 3. *Comparison between numerical solutions* (+) *and exact solutions* (*solid line*)*:* 100 *cells* (*left*) *and* 200 *cells* (*right*) *at* $t = 0.035$.

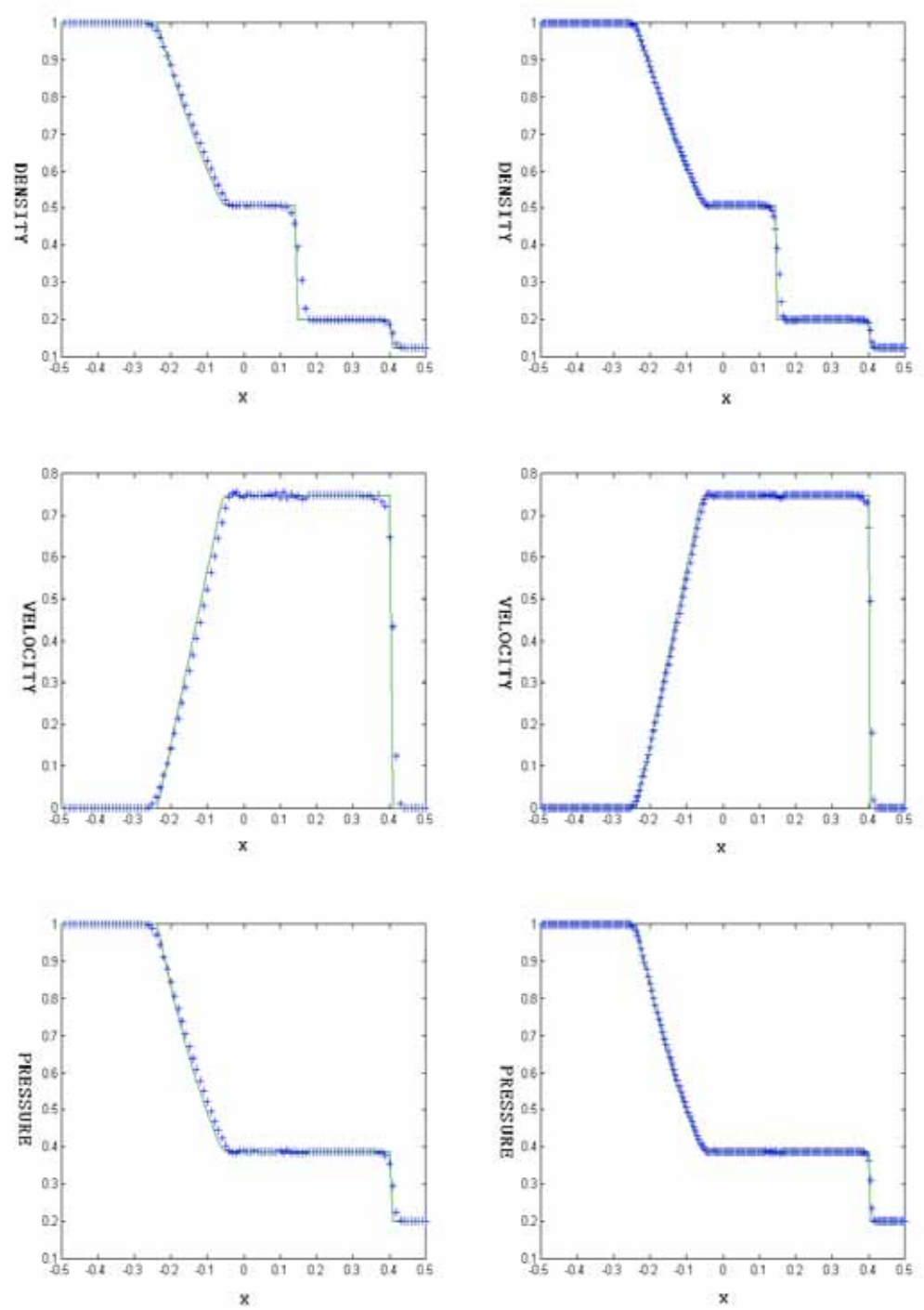


FIG. 4. *Example* 4. *Comparison between numerical solutions* (+) *and exact solutions* (*solid line*)*:* 100 *cells* (*left*) *and* 200 *cells* (*right*) *at* $t = 0.2$.

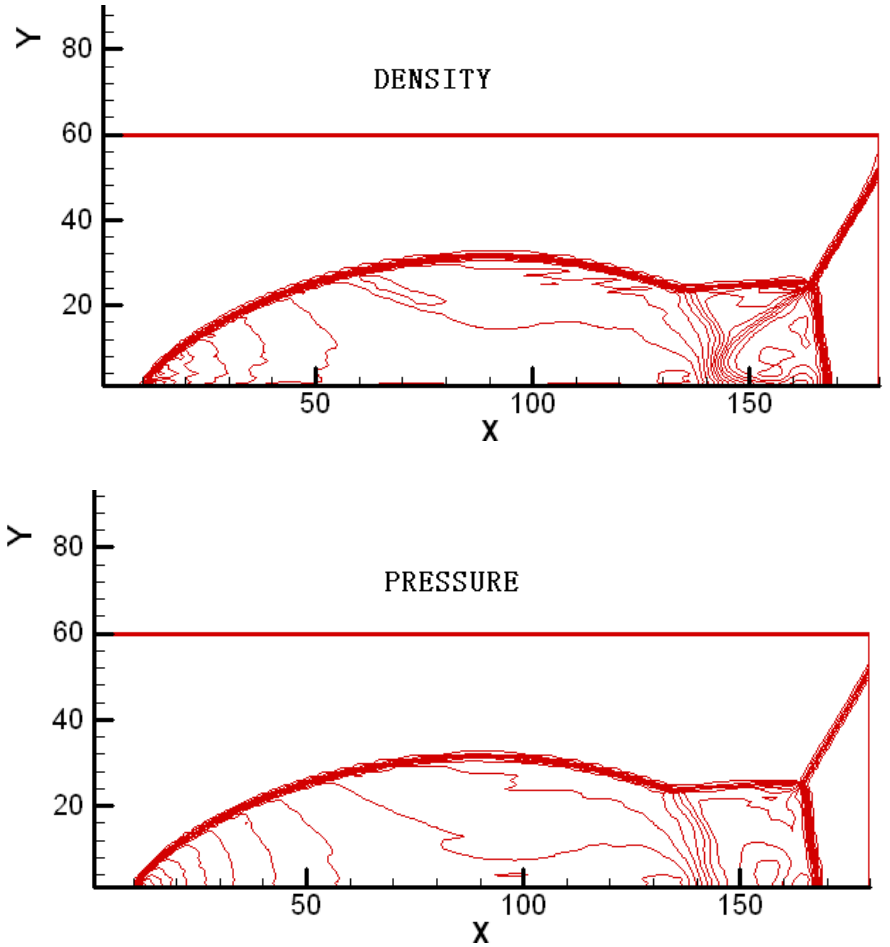


FIG. 5. *Example* 5. *Density and pressure distributions at* $t = 0.2$ *with a* 240×60 *mesh.*

where $M = (D - u_0)/C_0$, D is the Mach number of the shock, u_0 is a composition velocity consisting of the two velocity components in the normal direction of the preshock, and the variables with subscript 0 denote the preshock physical quantities.

This numerical example was studied extensively by Woodward and Colella [27], which is a standard test model in designing high-resolution schemes. In this numerical experiment, the computational domain is taken to be $[0, 4] \times [0, 1]$ and divided by the rectangular elements. The CFL number used is 0.18. The left and right boundaries use inflow and outflow boundary conditions, respectively, and the top boundary uses the exact boundary condition. The left side of the shock takes the postshock condition and the right side takes the preshock condition. For the bottom boundary, the exact postshock condition is imposed for the part $0 \le x \le 1/6$ and the rest uses the reflection boundary condition.

Figures 5 and 6 display the results in $[0, 3] \times [0, 1]$. Compared to the results in [8, 27, 35], our numerical results are found to be acceptable with good approximation accuracy.

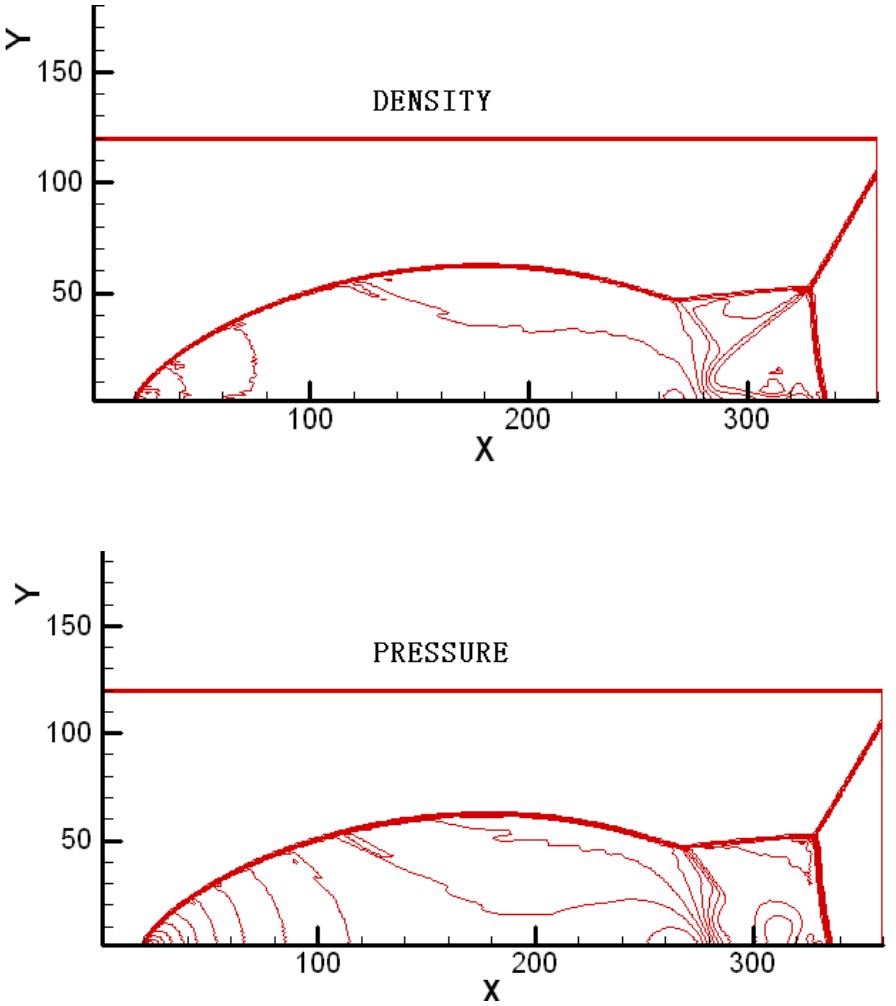


FIG. 6. *Example* 5. *Density and pressure distributions at* $t = 0.2$ *with a* 480×120 *mesh.*

6 (forward-facing step problem). In this example, the top and bottom boundaries are the reflective boundary condition. The left boundary uses an inflow boundary condition. For the right boundary, the reflective boundary condition is imposed for the part $x = 0.6$ and $0 \le y \le 0.2$, and the rest uses an outflow boundary condition.

The computational domain is taken to be $[0, 3] \times [0, 1]$ and divided by the rectangular elements. The CFL number used is 0.95. Observe that the corner of the face step is a singular point. In [27], a special corner treatment is made which improved the numerical results substantially. In [8], a local refining is used near the corner to enhance the overall numerical accuracy. In this work, we did not make any special efforts to handle the corner singularity.

Figures 7 and 8 display the computational results for Example 6. Compared to the results in [27], it is found that our numerical results are acceptable.

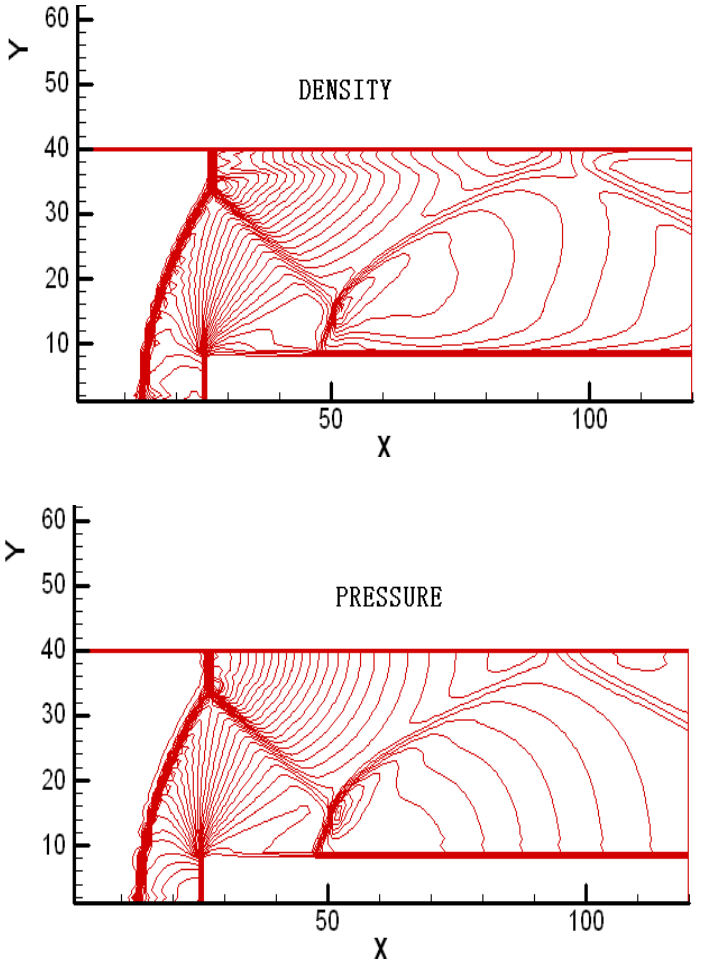


FIG. 7. *Example* 6. *Density and pressure distributions at* $t = 4.0$ *with a* 120×40 *mesh.*

In the above calculations, the collision time τ was taken to be

$$\tau = c_1 \Delta t + \Delta t \min(1, c), \tag{4.1}$$

where

$$c = c_2 \frac{|\rho_l/\lambda_l - \rho_r/\lambda_r|}{|\rho_l/\lambda_l + \rho_r/\lambda_r|},$$

and Δt is the CFL time step. c_1 and c_2 are positive constants. $c_1 = 0.01$ and $c_2 = 5$ are chosen in our numerical experiments.

We conclude this section by pointing out that the results presented in this section are more accurate than those obtained in our earlier work [35, 36].

5. Concluding remarks. In this paper, we have proposed a class of numerical schemes which combines the RKDG finite element scheme method and the second-order BGK scheme. The main advantage of this approach is that it is Riemann

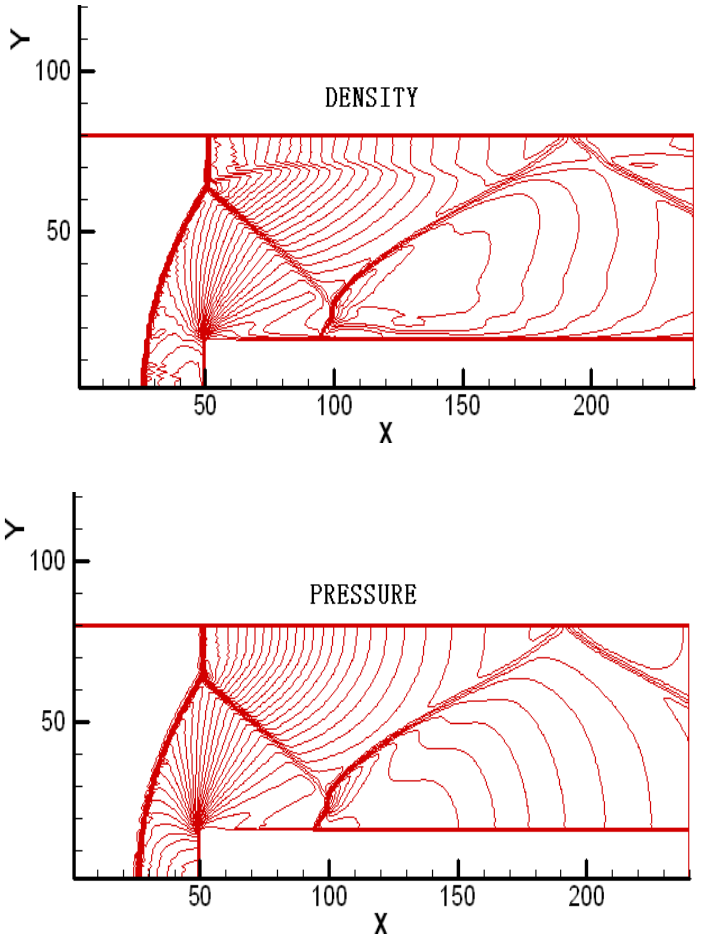


FIG. 8. *Example* 6. *Density and pressure distributions at* $t = 4.0$ *with a* 240×80 *mesh.*

solver-free when solving the Euler equations. The method also keeps the main advantages of the discontinuous Galerkin methods in solving the Euler equations. Further research of this scheme will be presented in future work.

REFERENCES

[1] A. D. KOTELNIKOV AND D. C. MONTGOMERY, *A kinetic method for computing inhomogeneous fluid behavior*, J. Comput. Phys., 134 (1997), pp. 364–388.

[2] A. D. KOTELNIKOV AND D. C. MONTGOMERY, *Shock induced turbulence in composite materials at moderate Reynolds numbers*, Phys. Fluids, 10 (1998), p. 2037.

[3] A. HARTEN, *High resolution schemes for hyperbolic conservation laws*, J. Comput. Phys., 49 (1983), pp. 357–393.

[4] A. HARTEN, B. ENGQUIST, S. OSHER, AND S. R. CHAKRAVARTHY, *Uniformly high order accurate essentially non-oscillatory schemes* III, J. Comput. Phys., 71 (1987), pp. 231–303.

[5] B. COCKBURN AND C.-W. SHU, *TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws* II: *General framework*, Math. Comp., 52 (1989), pp. 411–435.

[6] B. COCKBURN AND S.-Y. LIN, *TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws* III: *One-dimensional systems*, J. Comput. Phys., 84 (1989), pp. 90–113.

[7] B. Cockburn, S. Hou, and C.-W. Shu, *TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws* IV: *The multidimensional case*, Math. Comp., 54 (1990), pp. 541–581.

[8] B. Cockburn and C.-W. Shu, *The Runge-Kutta discontinuous Galerkin finite element method for conservation laws* V: *Multidimensional systems*, J. Comput. Phys., 141 (1998), pp. 199–224.

[9] B. Cockburn and C.-W. Shu, *The local discontinuous Galerkin method for time-dependent convection-diffusion systems*, SIAM J. Numer. Anal., 35 (1998), pp. 2440–2463.

[10] B. Cockburn, G. E. Karniadakis, and C. W. Shu, eds., *Discontinuous Galerkin methods. Theory, computation and applications*, Lect. Notes Comput. Sci. Eng. 11, Springer-Verlag, Berlin, 2000.

[11] B. Cockburn, G. Kanschat, I. Perugia, and D. Schötzau, *Superconvergence of the local discontinuous Galerkin method for elliptic problems on Cartesian grids*, SIAM J. Numer. Anal., 39 (2001), pp. 264–285.

[12] B. Perthame, *Second-order Boltzmann schemes for compressible Euler equations in one and two space dimensions*, SIAM J. Numer. Anal., 29 (1992), pp. 1–19.

[13] C. A. Kim and A. Jameson, *A robust and accurate LED-BGK solver on unstructured adaptive meshes*, J. Comput. Phys., 143 (1998), pp. 598–627.

[14] D. I. Pullin, *Direct simulation methods for compressible inviscid ideal gas flow*, J. Comput. Phys., 34 (1980), pp. 231–244.

[15] G. H. Xu, *Hydrodynamic and n-body schemes on an unstructured, adaptive mesh with application to cosmological simulations*, Monthly Notices Roy. Astronom. Soc. 288 (1997), pp. 903–919.

[16] K. H. Prendergast and K. Xu, *Numerical hydrodynamics from gas-kinetic theory*, J. Comput. Phys., 109 (1993), pp. 53–66.

[17] K. Xu, *Numerical Hydrodynamics from Gas-Kinetic Theory*, Ph.D. thesis, Columbia University, New York, NY, 1993.

[18] K. Xu and K. H. Prendergast, *Numerical Navier-Stokes solutions from gas kinetic theory*, J. Comput. Phys., 114 (1994), pp. 9–17.

[19] K. Xu, L. Martinelli, and A. Jameson, *Gas-kinetic finite volume methods, flux-vector splitting and artificial diffusion*, J. Comput. Phys., 120 (1995), pp. 48–65.

[20] K. Xu and A. Jameson, *Gas-kinetic relaxation (BGK-type) schemes for the compressible Euler equation*, in Proceedings of the 12th AIAA CFD Conference, San Diego, CA, 1995, paper 95-1736.

[21] K. Xu, C. Kim, L. Martinelli, and A. Jameson, *BGK-based schemes for the simulation of compressible flow*, Int. J. Comput. Fluid Dyn., 7 (1996), pp. 213–235.

[22] K. Xu, *BGK-based schemes for multicomponent flow calculation*, J. Comput. Phys., 134 (1997), pp. 122–133.

[23] K. Xu, *Gas-Kinetic Schemes for Unsteady Compressible Flow Simulations*, KVI report 1998-03, von Karman Institute for Fluid Dynamics, Rhode-Saint-Genese, Belgium, 1998.

[24] K. Xu, *A gas-kinetic scheme for the Euler equations with heat transfer*, SIAM J. Sci. Comput., 20 (1999), pp. 1317–1335.

[25] M. Su, K. Xu, and M. S. Ghidaoui, *Low speed flow simulation by the gas-kinetic scheme*, J. Comput. Phys., 150 (1998), pp. 17–39.

[26] M. N. Macrossan, *The equilibrium flux method for the calculation of flows with non-equilibrium chemical reactions*, J. Comput. Phys., 80 (1989), pp. 204–231.

[27] P. R. Woodward and P. Colella, *The numerical simulation of two-dimensional fluid flow with strong shocks*, J. Comput. Phys., 54 (1984), pp. 115–173.

[28] Q. Dai and X. Yu, *RKDG finite element schemes combined with gas-kinetic methods for* 1*D Euler compressible equations*, Math. Numer. Sin., 25 (2003), pp. 407–422.

[29] R. D. Reitz, *One-dimensional compressible gas dynamics calculations using the Boltzmann equations*, J. Comput. Phys., 42 (1981), pp. 108–123.

[30] R. H. Sanders and K. H. Prendergast, *The possible relation of the three-kiloparsec arm to explosions in the galactic nucleus*, Astrophys. J., 188 (1974), pp. 489–500.

[31] S. M. Deshpande, *A Second Order Accurate, Kinetic-Theory Based, Method for Inviscoid Compressible Flows*, NASA Langley Tech. paper 2613, NASA Langley Research Center, Hampton, VA, 1986.

[32] W. Eppard and B. Grossman, *A multidimensional kinetic-based upwind solver for the Euler equations*, in Proceedings of the 11th AIAA CFD Conference, Orlando, FL, 1993 .

[33] W. Huamo, *New accurate high resolution TVD schemes, analysis and applications*, in Proceedings of the International Symposium on Computational Fluid Dynamics, Nagoya, Japan, 1989.

[34] W. H. Reed and T. R. Hill, *Triangular Mesh Methods for the Neutron Transport Equation*, Los Alamos Scientific Laboratory Report LA-UR-73-479, Los Alamos Scientific Laboratory, Los Alamos, NM, 1973.

[35] X. Yu and Q. Dai, *RKDG finite element method combined with KFVS scheme for solving fluid dynamics system*, Numer. Methods Partial Differential Equations, to appear.

[36] X. Yu and Q. Dai, *RKDG finite element schemes combined with a gas-kinetic method for one-dimensional compressible Euler equations*, in Current Trends in Scientific Computing, Contemp. Math. 329, AMS, Providence, RI, 2003, pp. 355–372.

[37] X. Yu and Y. Wu, *A Taylor-Galerkin finite element method for one-dimensional hyperbolic conservation laws*, in Fluid Flow and Transport in Porous Media: Mathematical and Numerical Treatment, Contemp. Math. 295, AMS, Providence, RI, pp. 501–518.

[38] H. X. Zhang and F. G. Zhuang, *NND schemes and their applications to numerical simulation of two- and three-dimensional flows*, Adv. in Appl. Mech., 29 (1992), pp. 193–256.

SIAM J. SCI. COMPUT.
Vol. 28, No. 3, pp. 832–867
© 2006 Society for Industrial and Applied Mathematics

MODELING TEST RESPONSES BY MULTIVARIABLE POLYNOMIALS OF HIGHER DEGREES*

JIGUAN G. LIN†

Abstract. Engineering tests generally acquire a large set of response data at once while focusing on a small number of variables. Higher-order response models are better for capturing and displaying the characteristics such data can reveal, but response surface models traditionally are linear or quadratic (first or second order) only. If attempted for a higher order, the matrix X of the test inputs formed for fitting the test responses often immediately stops satisfying the full-rank assumption on X traditionally made for computing the regression coefficients. The resulting product matrix X^TX is singular and hence cannot be inverted, directly or indirectly, for obtaining the desired least-squares regression coefficients from the system of normal equations. Ill-conditioning of X when near rank deficiency, not unusual even in the traditional second-order response modeling, can also render the computed coefficients unreliable. This paper incorporates singular value decomposition (SVD) to handle the rank deficiency and ill-conditioning problems directly. The proposed SVD-based fitting algorithm enables response modeling to the desirable high-order reliably and efficiently with no difficulty despite rank deficiency in X. Two proposed trimming algorithms enable reduction of redundant terms or insignificant coefficients of the fitted polynomials on a sound mathematical base.

Key words. polynomial approximation, regression, number of terms, response surface, least squares, rank deficiency, singular value decomposition, higher-order response modeling, subset selection

AMS subject classifications. 41A10, 65D10, 65D17

DOI. 10.1137/040603954

1. Introduction.

Engineers have recently started to adopt the response surface methodology (RSM) [MyersM95, KhuriC96, HinkeK94] for various analysis, design, and optimization purposes (e.g., see [ChenGM01, SteffSY00, GiuntBH97, ChenAT96]). After making quantitative examination of their response surface model, the authors of [SteffSY00] analyzed the effects of six independent parameters on system performance using various three-dimensional response surface plots and wrote that "graphical interpretation was far more useful for exploring the results." Engineering test results usually become easier to understand or interpret when they are put into a graphic form. The graphics used in presenting the test results still are predominantly x-y curves with the response variable plotted against the test variable. They remain curves in a plane even when there are two or more test variables. In the case of two variables, for instance, one of them is treated as a parameter, and multiple parameterized curves are overlaid in a single x-y plane. Figure 1 is a typical traditional example, where 21 pressure measurements of an air-fuel-mixing stream by three rakes of seven probes are presented in three parameterized curves using rake position as the parameter [KamhaKT03]. Can one see as the authors of [KamhaKT03] claimed that near the top of the rakes the centerline pressure distribution shows separation of the flow, i.e., that some flow migrates to the outboard rakes? A single response surface in a three-dimensional space could make it easier than multiple overlaid parameterized curves in a plane to understand, interpret, or even visualize the same test results.

*Received by the editors February 7, 2004; accepted for publication (in revised form) November 21, 2005; published electronically June 9, 2006.
http://www.siam.org/journals/sisc/28-3/60395.html.
†Linsys, Lexington, MA 02421-5801 (linjiguan@email.com).

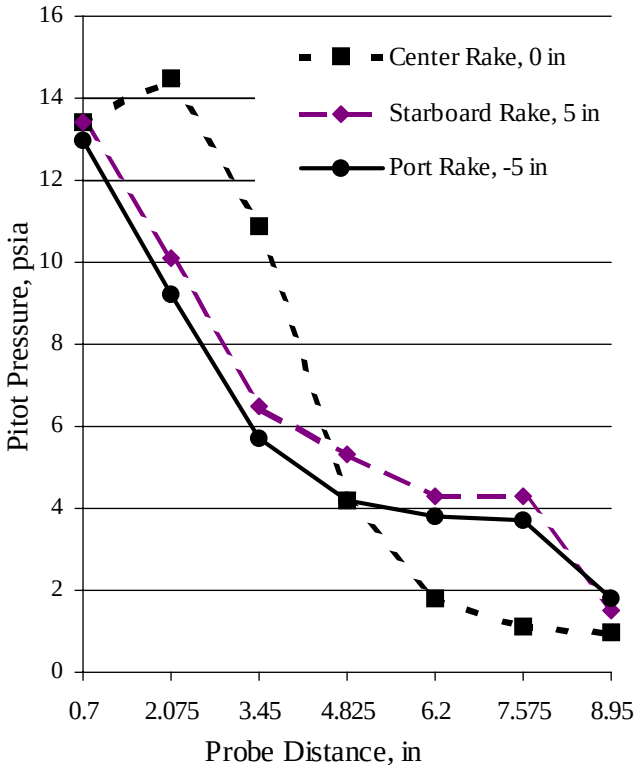


FIG. 1. *Pressure measurements by* 3 *rakes of* 7 *probes.*

A second-order response model obviously will not fit the pressure data of Figure 1 well, judging by the highly nonlinear sine-wave-like pressure profile along the center rake. We may consider building a higher-order response model. To build a model by RSM is to form a (test) matrix X first, using the specific values of the test variables. Then the resulting system of normal equations is solved for the unknown coefficients with the conventional assumption that the matrix X has full column rank [MyersM95, KhuriC96, Myers71], and hence the product matrix X^TX is nonsingular and invertible. We cannot build any higher-order response model for the pressure data this way because the matrix X stops having full column rank even for the third order.

We may seek D-optimal design of experiments [MyersM95, KhuriC96] for help, because it will maximize the determinant of the product matrix X^TX [BoxD71] and hence make X^TX most invertible. This would mean, for instance in using an algorithm such as proposed in [Mitch74], eliminating (or replacing) linearly dependent rows of matrix X and hence all the corresponding response data. Neglecting test results, especially those acquired with great effort and expense, such as from complex engineering or wind-tunnel tests or from flight experiments, however, would be impractical or unwise. Deleting rows from the matrix X for a third-order model of the pressure data actually cannot make the shrunk matrix X full column rank, 10, at all. Should we just give up and be content with a plainly flat second-order response surface?

Traditionally, response surface models are linear or quadratic (i.e., of first or second order). Recently, there have been some interesting advances to higher-order data or response fitting. The following are two examples.

(1) The problem of fitting scattered data is encountered frequently in many scientific disciplines. A variety of (quadratic) surface fitting methods are used in global methods, in which each interpolated value is influenced by all of the data, and in local methods, which are capable of treating much larger data sets. The most accurate local methods in earlier survey papers included a modified Shepard's method. Renka [Renka88] improved both efficiency and accuracy of this modified quadratic Shepard's method further by clever modification and enhancement.

Recently, Renka extended the modified quadratic Shepard method [Renka88] to a cubic Shepard method [Renka99] for constructing a smooth two-variable cubic function C that interpolates given data values y_i at given scattered points (nodes) in a two-dimensional plane for $i = 1, \ldots, N$, where N is a given large number. Accordingly, the accuracy is greatly improved further. The interpolating function C is defined as a weighted sum of N nodal functions C_j. At each node j, the nodal function C_j is a two-variable cubic polynomial that not only interpolates the data value y_j at node j but also fits the data values on a set of N_j nearby nodes in a weighted least-squares sense. The system of normal equations for each nodal function is solved for the regression coefficients. In both quadratic and cubic versions, when N_j nearby nodes have been incorporated for the least-squares fitting, the system is tested for ill-conditioning, and more nodes and data values are added if necessary, up to a limit that is arbitrarily set at $\min\{40, N-1\}$. In case the limit is exceeded, see [Renka88, Renka99] for more details.

(2) The drive for efficient methods of testing engineering design ideas and prototypes has led to a rapid increase in the field of numerical simulation using computers, which in turn has fueled the need for empirical models that interpolate data collected from these simulations. The data generated from computer code contain no measurement error, and models that are exact interpolators are desirable. Because engineering analysis, design, and optimization would be too slow if conducted on the simulator directly, these interpolators are used as fast approximations to the slow simulators. Traditional response surface methods are not sufficiently flexible to accurately model data from computer experiments. Bates, Giglio, and Wynn [BatesGW03] thus propose using a wider class of polynomial models and to take advantage of the lack of measurement error to develop a methodology for finding good interpolators. They propose a two-stage process. First, a search is conducted to find the smoothest (in the sense of minimum curvature) interpolating model from the set of "complete" models. Then backward stepwise regression is carried out on the chosen interpolator to reduce the number of terms in the polynomial according to a combination of model curvature and fitting residuals.

A model for which the number of terms in the polynomial is equal to the number of observations is an (exact) interpolator, provided that certain nonsingularity conditions on the (design) matrix X hold. Given a set of observations without error, in general there is no unique polynomial interpolator. When the number of variables and observations increases, the number of possible interpolators grows rapidly. Thus, in applications where a class of interpolators needs to be searched to find the best model for a given data set, the problem can become too complex. Instead, they consider the subclass C called "complete" models and present an algorithm that combines global and local search strategies within C for the efficient identification of interpolating

polynomials. A polynomial model is said to be complete if for each term in the model all of the terms that divide it are also included in the model. For example, with 2 variables (x_1, x_2) and 6 observations, there are four possible 6-term complete models:

$$\begin{aligned}
&\text{highest-order term is } x_1^5(\text{no } x_2) : 1, x_1, x_1^2, \ldots, x_1^5;\\
&\text{highest-order term is } x_2^5(\text{no } x_1) : 1, x_2, x_2^2, \ldots, x_2^5;\\
&\text{highest-order term is } x_1^2x_2 : 1, x_1, x_2, x_1^2, x_1x_2, x_1^2x_2;\\
&\text{highest-order term is } x_1x_2^2 : 1, x_1, x_2, x_1x_2, x_2^2, x_1x_2^2.
\end{aligned}$$

With 18 observations, there are six possible 18-term complete models, each having $x_1^{17}x_2^0, x_1^0x_2^{17}, x_1^8x_2, x_1x_2^8, x_1^5x_2^2$, or $x_1^2x_2^5$ as the highest-order term.

They must apply a matrix rank test to any interpolator considered during the search stage to prevent any model with singular design matrix X from being fitted. See [BatesGW03] for more details.

What if, in many other data or response fitting problems, adding more nodes or points with response data is hardly feasible? What if the matrix X^TX (or X if square) is nearly singular (i.e., ill-conditioned) because its computed determinant or one of its computed eigenvalues is nearly zero? How does one compute reliably for determining matrix rank, especially when the matrix is nearly rank deficient? In general, when the test or design matrix X, though still having full column rank, is near rank deficiency, the inverse of X^TX may have an excessively large magnitude erroneously, and any error in the response data will be magnified undesirably. What if one cannot simply give up such an ill-conditioned matrix?

Difficulties with solving ill-conditioned systems of linear equations have been actively addressed in the field of numerical algebra for a long time already, and reliable and efficient computational techniques have been developed (see, e.g., [GolubV83, LawsoH74, Stewa73, Wilki68]). In order to enable RSM to handle general test or design matrix X, for higher order or not, we propose that problems with rank deficiency and near rank deficiency in the matrix be dealt with directly by using the singular value decomposition (SVD) techniques and algorithms. Thus, the main effort in this paper is to incorporate SVD appropriately into conventional first/second-order and unconventional higher-order response modeling. Three associated issues are also addressed.

1. While developing and advocating SVD for solving general ill-conditioned systems of linear equations, Golub has persistently argued against the RSM-conventional normal-equation method of solving least-squares (LS) fitting problems. Should the normal-equation method be abandoned?

2. When the matrix X has full column rank, as in the traditional case, we may want to lop off some of the fitted coefficients if they are considered insignificant. Student's t-statistic and the F-statistic are commonly used for such a significance test. They are based on the assumption that the uncertainties in the response data are normally distributed. There are cases, such as numerical experiments using the computer, where normal distribution of the uncertainties or random errors is not justifiable and may be purely an arbitrary assumption for analytical convenience. What is a reasonable mathematical alternative to such statistical tests as a truncation criterion?

3. After having successfully modeled with a rank-deficient matrix X, there are more coefficients than the matrix rank, and we may want to reduce the redundant coefficients. The usual t- and F-statistical tests are not applicable if the matrix X does not have full column rank [Searl71]. What is a reasonable analytical way to truncate

such a fitted polynomial to a full-rank state without causing excessive increase in the residuals?

In section 2, first we formulate higher-order response modeling as a linear regression problem in the standard general form. Then we address major issues with solving the resulting, possibly ill-conditioned, system of normal equations. We also discuss whether we can simply turn the crank of a generalized inverse or pseudoinverse algorithm for the desired LS solution when the rank of the matrix X^TX (or X) is not known a priori. In section 3, we propose that the SVD of the matrix X be calculated first and then used for computing the desired solution. Key SVD properties, particularly the rank of a rank-deficient matrix, are introduced. We then derive the SVD-based solution from three different viewpoints and address the associated Issue 1. In section 4, we propose a specific computational formula for the desired SVD-based solution plus two new, intuitively meaningful fitness measures. Then Algorithm 1 is formulated for computing the LS fitting coefficients in the general case where the matrix X may or may not have full rank.

Now that Algorithm 1 handles rank deficiency effectively, we can model given responses with a higher order. An interesting question is, How high? In section 5, we derive some formulas for calculating the number of coefficients in higher-degree polynomials and discuss practical limitations on model order. On the other hand, after having modeled given responses with a desirable high order, we may want to reduce the model complexity by trimming off redundant terms or insignificant coefficients. Section 6 thus addresses the associated Issues 2 and 3 and develops two unconventional trimming algorithms. Finally, in section 7, demonstrations and applications are given. First, we use published results of two well-executed or well-illustrated examples of conventional RSM application for apples-to-apples comparison of the proposed SVD approach on the same conventional ground. Then we use two hypothetical examples to validate the proposed computational technique by demonstrating that if the response is actually a polynomial, the proposed technique can model it accurately whether the matrix X has full rank or has a large rank deficiency. Then an example of real hardware tests with a truly unknown response function is given to demonstrate the application of three proposed algorithms. There we show that the proposed SVD approach enables us to model the pressure responses of Figure 1 to a desirable higher order without difficulty. We can therefore obtain an interesting three-dimensional response surface plot that makes it rather simple to see and interpret the flow separation and the outward flow migration.

2. Higher-order regression. Given values of k independent test variables, $x_1, \ldots, x_k$, and the associated values of response variable, y, the general response modeling problem is to find a polynomial P_m of degree m that agrees with response y as well as possible at the given test points. To build a third-order response model, for instance, is to postulate a (full) cubic polynomial,

$$P_3(x_1, \ldots, x_k) = \beta_0 + \sum_{j_1=1}^{k} \beta_{j_1} x_{j_1} + \sum_{j_1=1}^{k} \sum_{j_2 \geq j_1} \beta_{j_1 j_2} x_{j_1} x_{j_2} + \sum_{j_1=1}^{k} \sum_{j_2 \geq j_1} \sum_{j_3 \geq j_2} \beta_{j_1 j_2 j_3} x_{j_1} x_{j_2} x_{j_3},$$

with undetermined coefficients (also called model parameters) $\beta_0, \beta_1, \ldots, \beta_k, \ldots, \beta_{kk}, \ldots, \beta_{kkk}$). We may occasionally see in design-of-experiment literature (e.g., [BoxD71])

that a polynomial basically of degree 2 includes a couple of two- or three-variable interactions. Here we start with full-order modeling since in general, the empirical models that are most sensible in practical situations are those that contain all terms up to a given order [BoxD71]. Suppose there are N observations or, equivalently, N test points at which responses are available. Let $x_{i1}, \ldots, x_{ik}$ represent specifically the inputs at test point i and y_i the corresponding response. Then the given test data can be expressed as

$$y_i = P_3(x_{i1}, \ldots, x_{ik}) + \varepsilon_i, \qquad i = 1, \ldots, N, \tag{1}$$

with ε_i representing the associated uncertainties. Denote by X the test matrix having

$$(1, x_{i1}, \ldots, x_{ik}; x_{i1}^2, x_{i1}x_{i2}, \ldots, x_{ik}^2; x_{i1}^3, x_{i1}^2 x_{i2}, \ldots, x_{ik}^3)$$

as the ith row and by $\boldsymbol{\beta}$ the coefficient vector

$$(\beta_0, \beta_1, \ldots, \beta_k; \beta_{11}, \beta_{12}, \ldots, \beta_{kk}; \beta_{111}, \beta_{112}, \ldots, \beta_{kkk}).$$

Also, denote by $\boldsymbol{y}$ the response vector $(y_1, \ldots, y_N)$ and by $\boldsymbol{\varepsilon}$ the uncertainty vector $(\varepsilon_1, \ldots, \varepsilon_N)$. Then (1) can be recast in the standard matrix form as

$$\boldsymbol{y} = X\boldsymbol{\beta} + \boldsymbol{\varepsilon}. \tag{2}$$

Similarly, to build a higher, mth-order response model is to postulate a polynomial of degree m and express the given test data in a form similar to (2), except that as a result of including higher-power products of the test inputs, the matrix X has additional columns and the vector $\boldsymbol{\beta}$ has additional components. In other words, X is an $N \times p$ matrix and $\boldsymbol{\beta}$ is a p-component vector, where p denotes the number of the terms in the polynomial.

In the following, consider the fitting of N response data by a generic mth-order model and use (2) as the general form with matrix X having p columns and vector $\boldsymbol{\beta}$ having p components. This general full-order formulation applies to all special cases where specific terms are omitted. (In such cases, delete the corresponding columns of the matrix X and the corresponding components of the vector $\boldsymbol{\beta}$.) To fit by a polynomial P_m is as usual to find a particular vector $\boldsymbol{\beta}$ that minimizes the sum of squared deviations

$$\sum_{i=1}^{N} [y_i - P_m(x_{i1}, \ldots, x_{ik})]^2 = (\boldsymbol{y} - X\boldsymbol{\beta})^T (\boldsymbol{y} - X\boldsymbol{\beta}).$$

Differentiating the right-hand side with respect to vector $\boldsymbol{\beta}$ and setting the result to zero, we readily obtain the following necessary and sufficient condition for vector $\boldsymbol{b}$ to be a LS solution:

$$X^T X\boldsymbol{b} = X^T \boldsymbol{y}. \tag{3}$$

It is the system of [illegible] in the matrix form. A common practice in the first- and second-order cases is to invert the product $X^T X$ and obtain a solution from (3) simply as

$$\boldsymbol{b} = (X^T X)^{-1} X^T \boldsymbol{y} \tag{4}$$

with the traditional assumption [Myers71, MyersM95, KhuriC96] that $N > p$ and the matrix X has full column rank, p. The matrix X, however, does not necessarily have full rank, and hence the product X^TX does not always have inverse, even when N is much larger than p. Solving (3) indirectly as usual by applying Cholesky factorization, Householder orthogonalization, Gram–Schmidt orthonormalization, or Givens rotations will lead to various numerical difficulties when X^TX, though nonsingular, is near rank deficiency.

An overdetermined system of linear equations is generally known to have no solution and an underdetermined or rank-deficient system to have infinitely many solutions. A rank-deficient system, even not overdetermined, such as the following, also may have no solution:

$$\begin{aligned} z_1 + 2z_2 &= 1, \\ 2z_1 + 3z_2 &= 2, \\ 3z_1 + 4z_2 &= 3, \\ 4z_3 + 5z_4 &= 4. \end{aligned}$$

Fortunately, we can assert, as it would likely have been taken for granted, that the normal equations always have solutions even when X is rank deficient.

THEOREM 1. For any vector $\boldsymbol{y}$, the system of normal equations (3) has a solution $\boldsymbol{b}$.

PROOF. All we need is to prove that the augmented matrix $[X^TX, X^T\boldsymbol{y}]$ has the same rank as X^TX. First of all, denote the rank of matrix X by r. Arrange the columns of X such that the first r columns are linearly independent. Denote by $X_R = [X_1, X_2, \ldots, X_r]$ the corresponding submatrix of matrix X, where X_j represents the jth column of X. Then any other column of X, represented by X_{r+1}, must be a linear combination of these r columns; that is, $X_{r+1} = X_R\boldsymbol{c}$ for some nonzero vector $\boldsymbol{c}$.

The first r rows of product X^TX are $X_1^TX, X_2^TX, \ldots, X_r^TX$. The corresponding submatrix of X^TX is X_R^TX. It has rank r. Let X_{r+1}^TX represent any other row of X^TX. Then it must be a linear combination of these rows because $X_{r+1}^TX = \boldsymbol{c}^TX_R^TX$.

The first r rows of the augmented matrix $[X^TX, X^T\boldsymbol{y}]$ are $[X_1^TX, X_1^T\boldsymbol{y}]$, $[X_2^TX, X_2^T\boldsymbol{y}], \ldots, [X_r^TX, X_r^T\boldsymbol{y}]$. The submatrix of $[X^TX, X^T\boldsymbol{y}]$ corresponding to these rows is $[X_R^TX, X_R^T\boldsymbol{y}]$. It has rank r. Likewise, let $[X_{r+1}^TX, X_{r+1}^T\boldsymbol{y}]$ represent any other row of the augmented matrix. Then it must be a linear combination of these r rows because $[X_{r+1}^TX, X_{r+1}^T\boldsymbol{y}] = [\boldsymbol{c}^TX_R^TX, \boldsymbol{c}^TX_R^T\boldsymbol{y}] = \boldsymbol{c}^T[X_R^TX, X_R^T\boldsymbol{y}]$. This means that product X^TX augmented by vector $X^T\boldsymbol{y}$ gets no increase in rank.

Consequently, given any vector $\boldsymbol{y}$, $X^T\boldsymbol{y}$ is a linear combination of the columns of X^TX. The combination coefficients thus provide a solution vector $\boldsymbol{b}$ to the system of normal equations. □

Theorem 1 implies that the system of normal equations is always consistent and that when X is rank deficient, the system may have multiple solutions. Theoretically, any consistent system of equations can be solved by using a generalized inverse of either the weakest kind, the weak or normalized kind, or the Moore–Penrose kind [Rohde66, Searl71]. In statistical analysis of unbalanced data [Searl71], the matrix X for first-order regression is a sparse incidence matrix, consisting only of 1's and 0's ("yes" or "no" it is a type considered), which rarely has full column rank, even though N is much greater than p. The product X^TX has no inverse, and a generalized inverse is computed. Given a matrix A, such as $A = X^TX$, a generalized inverse is any matrix G satisfying $AGA = A$. An algorithm for computing G is described essentially as follows [Searl71, pp. 5–6]. In matrix A, of rank r, find a nonsingular minor of

order r and bring it to the leading position. Let P and Q be the elementary row and column operations such that

$$PAQ = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} \quad \text{and} \quad \text{define } C = \begin{bmatrix} B_{11}^{-1} & 0 \\ 0 & 0 \end{bmatrix},$$

where B_{11} is nonsingular of order r. Then $G = QCP$ is a generalized inverse of A. Mentioned explicitly in [Searl71, pp. 6–7] are that the computation is based on knowing the rank of matrix A and that most methods involve the same kind of numerical problems as are incurred in calculating the regular inverse B_{11}^{-1} of a nonsingular matrix B_{11}. Numerical difficulties are frequently encountered in situations where near rank deficiency prevails; rounding errors and fuzzy data make rank determination a nontrivial exercise. One of the most valuable aspects of SVD is that it enables us to deal sensibly with the concept of matrix rank [GolubV83, p. 19].

The unique solution obtainable by using the pseudoinverse [Grevi59] is the smallest of all kinds of generalized inverses, having the least Euclidean norm. Computing the pseudoinverse of a matrix numerically has not been without complications and difficulties. It not only has four different conditions to satisfy but also has to assess the matrix rank from, for instance, extremely small computed eigenvalues, many of which are essentially zero in the finite-precision, floating-point computer operations. The development of reliable SVD calculation [GolubK69, GolubR70] started to greatly simplify it. Even then, no SVD subroutines for calculating pseudoinverses or solving LS solutions, such as the Golub–Reinsch algorithm [GolubR70] or Chan's improved algorithm [Chan82], produce a pseudoinverse automatically. They leave it to their user to decide the rank of the specific matrix after the subroutines have terminated before composing the desired pseudoinverse of X or X^TX. The SVD algorithms can compute all the singular values precisely and reliably, but the user has to decide how small a computed singular value is to be considered negligible (and thus treated as zero). This is important, since the pseudoinverse is essentially obtained from inverting each nonzero singular value deemed significant and retained. Small changes in matrix X or in vector $\boldsymbol{y}$ can induce undesirably large changes in solution $\boldsymbol{b}$ if an extremely small computed singular value is retained for inversion.

Therefore, instead of computing and applying pseudoinverses of X^TX formally, we propose that the following SVD-based solution of the normal equations be computed directly. It is simpler and more efficient, avoiding unnecessary matrix formation and multiplication.

3. Solution via SVD. We propose that the system of normal equations be solved for LS solution $\boldsymbol{b}$ by performing the SVD of the matrix X first. The following are some useful properties associated with SVD. See [GolubV83, pp. 16–20; KlemaL80, LawsoH74, Golub69].

LEMMA 1 (see [GolubV83, pp. 16–17; LawsoH74]). [illegible] $N \times p$, [illegible] X [illegible] [illegible] $N \times N$, [illegible] $U = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_N]$ [illegible] [illegible] $p \times p$, [illegible] $V = [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_p]$ [illegible]

$$U^TXV = S \quad \text{and hence} \quad X = USV^T, \tag{5}$$

[illegible] S [illegible] $N \times p$ [illegible] [illegible] [illegible]

square roots of the eigenvalues of X^TX or XX^T [LawsoH74; GolubV83, p. 285], but to compute them by first finding the eigenvalues of either product matrix is not a reliable approach (for an illustration, see [KlemaL80]). Columns $\boldsymbol{u}_j$ are the jth left singular vector and columns $\boldsymbol{v}_j$ the jth right singular vectors. Golub and Reinsch [GolubR70] have developed a very efficient and stable algorithm for calculating the SVD, fairly sophisticated and reliable implementation of which can be found in LINPACK or EISPACK; see, e.g., [KlemaL80, GolubV83] for more discussions. The Golub–Reinsch algorithm is made further efficient by Chan's improvement [Chan82, GolubV83].

1 (see [GolubV83, p. 18]). The rank of X is r if (a) all but the first r singular values of matrix X are zero, i.e., $s_1 \geq \cdots \geq s_r > s_{r+1} = \cdots = 0$, and $r < \min\{N, p\}$, or (b) all singular values of X are positive, i.e., $s_1 \geq \cdots \geq s_r > 0$, and $r = \min\{N, p\}$.

2 (see [GolubV83, p. 19]). Denote the rank of matrix X by r and the diagonal matrix of order r consisting of all r nonzero singular values of X by $S_r = \operatorname{diag}\{s_1, \ldots, s_r\}$. Partition the orthogonal matrix U into two parts, $U = [U_r, U_z]$, with $U_r = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_r]$ consisting of its first r columns. Also partition the orthogonal matrix V into two parts, $V = [V_r, V_z]$, with $V_r = [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_r]$ consisting of its first r columns. Then

$$U_r^T U_r = I_r, \quad V_r^T V_r = I_r, \quad X = U_r S_r V_r^T, \tag{6}$$

and I_r denotes the identity matrix of order r.

Now we are ready to derive the SVD-based solution by three approaches.

3.1. Solving the normal equations.

THEOREM 2. SVD X U, V S X r SVD X r (3)

$$V_r^T \boldsymbol{b} = S_r^{-1} U_r^T \boldsymbol{y} \tag{7}$$

$\boldsymbol{y}$ LS

$$\boldsymbol{b} = V_r S_r^{-1} U_r^T \boldsymbol{y}. \tag{8}$$

. It follows from (5) and orthogonality of U and V that $X^TX = VS^TSV^T$ and $X^T\boldsymbol{y} = VS^TU^T\boldsymbol{y}$. Then the necessary and sufficient condition, (3), is equivalent to $S^TSV^T\boldsymbol{b} = S^TU^T\boldsymbol{y}$ because V is nonsingular. Upon partitioning the matrices with respect to rank r of X as described above, the necessary and sufficient condition is equivalent to

$$\begin{bmatrix} S_r^2 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} V_r^T \\ V_z^T \end{bmatrix} \boldsymbol{b} = \begin{bmatrix} S_r & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} U_r^T \\ U_z^T \end{bmatrix} \boldsymbol{y},$$

which, in turn, is equivalent to (7) as claimed because S_r is nonsingular.

Now, given any $\boldsymbol{y}$, vector $\boldsymbol{b}$ defined by (8) is obviously an LS solution because by the second identity in (6), it satisfies an equivalent form of the normal equations, (7). Moreover, $V_rS_r^{-1}U_r^T\boldsymbol{y}$ is also the solution of the following minimum-norm problem constrained by this SVD form of the normal equations:

$$\begin{aligned} &\text{minimize} && \boldsymbol{\beta}^T\boldsymbol{\beta} \\ &\text{subject to} && V_r^T\boldsymbol{\beta} = S_r^{-1}U_r^T\boldsymbol{y}. \end{aligned}$$

To see that define Lagrangian $L = \boldsymbol{\beta}^T\boldsymbol{\beta} + \boldsymbol{\lambda}^T(V_r^T\boldsymbol{\beta} - S_r^{-1}U_r^T\boldsymbol{y})$ with vector $\boldsymbol{\lambda}$ of Lagrange multipliers. Differentiating it with respect to vector $\boldsymbol{\beta}$ and setting the result to zero gives $\boldsymbol{\beta} = -V_r\boldsymbol{\lambda}/2$. Substituting it into the equality constraint and using the second identity in (6) yields $-\boldsymbol{\lambda}/2 = S_r^{-1}U_r^T\boldsymbol{y}$. Thus, we have $\boldsymbol{\beta} = V_rS_r^{-1}U_r^T\boldsymbol{y}$ as the constrained minimum-norm solution. □

3.2. Pseudoinverting $\boldsymbol{X^TX}$. The pseudoinverse of a matrix A is a unique matrix Z that satisfies the following four Moore–Penrose conditions [Grevi59, LawsoH74, GolubV83]:

(i) $AZA = A$,
(ii) $ZAZ = Z$,
(iii) $(AZ)^T = AZ$, and
(iv) $(ZA)^T = ZA$.

In the case of product X^TX with SVD of X, we have a specific closed-form expression.

THEOREM 3. [illegible] X^TX, $V_rS_r^{-2}V_r^T$ [illegible]. Take $A = X^TX = V_rS_r^2V_r^T$ by the third identity in (6). Then $Z = V_rS_r^{-2}V_r^T$ satisfies all four Moore–Penrose conditions by the second identity in (6) as follows:

(a) $AZA = V_rS_r^2V_r^T(V_rS_r^{-2}V_r^T)V_rS_r^2V_r^T = V_rS_r^2(V_r^TV_r)S_r^{-2}(V_r^TV_r)S_r^2V_r^T = V_rS_r^2V_r^T = A$.
(b) $ZAZ = (V_rS_r^{-2}V_r^T)V_rS_r^2V_r^T(V_rS_r^{-2}V_r^T) = V_rS_r^{-2}(V_r^TV_r)S_r^2(V_r^TV_r)S_r^{-2}V_r^T = V_rS_r^{-2}V_r^T = Z$.
(c) $AZ = V_rS_r^2V_r^T(V_rS_r^{-2}V_r^T) = V_rV_r^T$ [illegible], $(AZ)^T = AZ$).
(d) $ZA = (V_rS_r^{-2}V_r^T)V_rS_r^2V_r^T = V_rV_r^T$ [illegible], $(ZA)^T = ZA$). □

It is interesting to see that $(V_rS_r^{-2}V_r^T)$ is an inverse of $X^TX = V_rS_r^2V_r^T$ in the sense of (squares of the nonzero singular values) S_r^2 being inverted inside the same wrap by V_r.

Now define $(X^TX)^+ = V_rS_r^{-2}V_r^T$ as the pseudoinverse of X^TX. Then a solution of normal equations can be conveniently written as

$$\boldsymbol{b} = (X^TX)^+X^T\boldsymbol{y}, \tag{8a}$$

just as (4). In fact, it is easy to prove that (8a) is actually the general form, taking (4) as the special case where pseudoinverse becomes regular inverse because of nonsingularity of X^TX. Also, writing it out in the SVD form, we have

$$\boldsymbol{b} = (X^TX)^+X^T\boldsymbol{y} = (V_rS_r^{-2}V_r^T)\ V_rS_rU_r^T\boldsymbol{y} = V_rS_r^{-1}U_r^T\boldsymbol{y},$$

which is the same as (8).

3.3. Pseudoinverting $\boldsymbol{X}$. Define S^+, a $p \times N$ matrix having reciprocals of the r nonzero singular values (i.e., $1/s_1, \ldots, 1/s_r$) in the leading diagonal positions but zeros elsewhere, as the pseudoinverse of the $N \times p$ singular-value matrix S. Define $X^+ = VS^+U^T$ as the pseudoinverse of X. Then, as stated in [GolubV83, p. 139],

$$\boldsymbol{b} = X^+\boldsymbol{y} \tag{8b}$$

is the minimum-norm LS solution of the system

$$X\boldsymbol{\beta} = \boldsymbol{y}. \tag{9}$$

Upon partitioning the SVD matrices according to rank r of matrix X, the pseudoinverse of X readily becomes $X^+ = VS^+U^T = V_rS_r^{-1}U_r^T$. Consequently, solution (8b) also becomes identical to solution (8).

The method of normal equations has traditionally been most commonly used to solve LS problems. Since X is conventionally presumed to have full column rank, the product matrix X^TX is both symmetric and positive definite and thus naturally suitable for applying Cholesky factorization algorithms, which not only run to completion but also are extremely stable [GolubV83, pp. 85, 89]. Nonetheless, Golub [Golub69, GolubV83] has persistently argued, with good reasons based on Wilkinson's extensive studies on Cholesky algorithms [Wilki68], against using Cholesky algorithms with the method of normal equations particularly when matrix X is nearly rank deficient. It is asserted in [GolubV83, p. 170] that the only fully reliable way to treat rank deficiency is to compute the SVD. What we have shown above is that if SVD is performed on X first, then solving the normal equations in the SVD form, (7), for LS solution (8) is the same, and equally reliable, as solving the system of equations, (9), for LS solution (8b) with the pseudoinverse of X. The associated Issue 1 is thus resolved.

4. Proposed computational technique. RSM has rarely made use of polynomials of degree higher than 2, and we suggest that one should try if it would fit and characterize the test results better. If one decides to try for a higher-order model and applies the method of normal equations as usual, one may likely encounter difficulties with rank deficiency or near rank deficiency of the resulting matrix X, difficulties that one generally need not deal with in the conventional first- or second-order modeling. Here we propose that SVD, a mathematical technique well-developed for treating ill-conditioned systems, be employed right away in the numerical computation so that one can easily succeed in the modeling whether or not the resulting matrix X has full rank. A major advantage of the proposed SVD approach is that the rank of any matrix X can be determined appropriately, reliably, and efficiently before computing the LS solution.

4.1. Computational formula. Lawson and Hanson [LawsoH74] pointed out that the inverse matrix is a very useful mathematical concept even though efficient and reliable methods for computing the solution of, say, system (9) when X is nonsingular do not involve explicit computation of an inverse matrix. Likewise, the pseudoinverse is also a very useful mathematical concept, but one would not usually compute it explicitly in computing a minimum-norm LS solution. Thus, (8) is more suitable for numerical computation than (8a) and (8b). Furthermore, it is more efficient to compute directly by the following expanded form, saving all the wasting operations of multiplying with the zero off-diagonal elements of the sparse S_r^{-1}:

$$\boldsymbol{b} = \sum_{j=1}^{r} (\boldsymbol{u}_j^T \boldsymbol{y}/s_j)\boldsymbol{v}_j. \tag{10}$$

4.2. Measures of fitness. The modeling errors minimized with LS solution $\boldsymbol{b}$ are the residuals: $\boldsymbol{e} = \boldsymbol{y} - X\boldsymbol{b}$. The sum of squared modeling errors, SSE $= \boldsymbol{e}^T\boldsymbol{e}$, is useful for assessing the accuracy, or measuring the fitness, of the model. Another measure is the spread of the uncertainties. A conventional convenient practice [MyersM95, KhuriC96] is to assume that all the uncertainties are independent and have zero mean and identical variance, σ^2. An estimate of the variance, hence the spread, is then derivable from the residuals.

THEOREM 4. [illegible] $E(\boldsymbol{\varepsilon}) = 0, E(\boldsymbol{\varepsilon}\boldsymbol{\varepsilon}^T) = \sigma^2 I_N$ [illegible] E [illegible] X [illegible] r [illegible] SVD [illegible]

[illegible] 2 [illegible]

$$E(\text{SSE}) = \sigma^2(N - r). \tag{11}$$

[illegible]. Relate the residuals to the uncertainties $\boldsymbol{\varepsilon}$ through (6) and (8) as follows:

$$\begin{aligned}\boldsymbol{e} &= \boldsymbol{y} - U_r S_r V_r^T V_r S_r^{-1} U_r^T \boldsymbol{y} \\ &= \boldsymbol{y} - U_r U_r^T \boldsymbol{y} \\ &= (I_N - U_r U_r^T) X\boldsymbol{\beta} + (I_N - U_r U_r^T)\boldsymbol{\varepsilon} \\ &= (I_N - U_r U_r^T)\boldsymbol{\varepsilon},\end{aligned}$$

where $(I_N - U_r U_r^T)X = X - U_r U_r^T (U_r S_r V_r^T) = X - U_r S_r V_r^T = 0$. Then

$$\text{SSE} = \boldsymbol{e}^T\boldsymbol{e} = \boldsymbol{\varepsilon}^T(I_N - U_r U_r^T)(I_N - U_r U_r^T)\boldsymbol{\varepsilon} = \boldsymbol{\varepsilon}^T(I_N - U_r U_r^T)\boldsymbol{\varepsilon} = tr[(I_N - U_r U_r^T)\boldsymbol{\varepsilon}\boldsymbol{\varepsilon}^T],$$

where tr denotes the trace of the matrix. Taking the expected value, we have

$$\begin{aligned}E(\text{SSE}) &= tr[(I_N - U_r U_r^T)E(\boldsymbol{\varepsilon}\boldsymbol{\varepsilon}^T)] = \sigma^2 tr(I_N - U_r U_r^T) \\ &= \sigma^2[N - tr(U_r^T U_r)] = \sigma^2(N - r). \qquad \square\end{aligned}$$

Therefore, for $N > r$, an unbiased estimator of variance σ^2 is

$$\hat{\sigma}^2 = \text{SSE}/(N - r). \tag{12}$$

Vector $\boldsymbol{e}$ represents the deviations from the responses by the fitted polynomial, and so does vector $\boldsymbol{\varepsilon}$ but by the underlying unknown polynomial. When would both deviations $\boldsymbol{e}$ and $\boldsymbol{\varepsilon}$ have the same spread, i.e., $E(\text{SSE}) = \sigma^2$? By (11), it would happen only if $N - r = 1$ when SSE $\neq 0$. In general, $E(\text{SSE}) \geq \sigma^2$; that is, for $N - r \geq 1$, we should not (and cannot) expect the residuals to be contained within the unknown uncertainties. Therefore, each can be a different fitness measure in its own right.

Define the root-mean-square (RMS) modeling error as RMSE $= \sqrt{(\text{SSE}/N)}$, evaluated over the N responses as usual in engineering (but not over the "degrees of freedom" in the statistical analysis of variance). We prefer the intuitive meaning of average individual size conveyed by RMS to the concept of root-sum-square, which increases with the group size (i.e., the number of residuals in the summation). Define the estimated standard deviation by (12) as StDev $= \sqrt{[\text{SSE}/(N - r)]}$, provided $N > r$.

Relative errors generally are more meaningful than absolute errors in accuracy assessment. Since both represent deviations from the same responses, we use the RMS value of the responses, RMSY $= \sqrt{(\text{SSY}/N)}$, as the common reference, where $SSY = \sum_{i=1}^{N} y_i^2$. Then define the relative RMS error (expressed in percentage) as

$$\text{RMSE}\% = 100(\text{RMSE}/\text{RMSY}). \tag{13}$$

Similarly, define the relative estimated standard deviation (also expressed in percentage) as

$$\text{StDev}\% = 100(\text{StDev}/\text{RMSY}). \tag{14}$$

Both RMSE% and StDev% indicate the goodness of the fit: the smaller the better. The former always decreases as the model order increases, but the latter may not. The latter is useful for helping one determine whether or not raising the order is necessary. Example 2 in section 7 shows that StDev% is a good alternative to the conventional statistical counterpart in discriminating fitness of models of different orders. Because they are normalized with respect to the response data, both RMSE% and StDev% are good to be universal fitness measures for general performance assessment when comparing across different sets of response data or even different response modeling situations.

These new definitions are compatible, respectively, with the R^2 and R^2_{adj} statistics used in the conventional response surface methodology [MyersM95, KhuriC96]. The square of the coefficient of correlation, also known as coefficient of multiple determination, $R^2 = 1 - \text{SSE}/\text{SST}$, is used as a measure of the amount of reduction in the variability of response obtained by regression [MyersM95], where $\text{SST} = \sum_{i=1}^{N}(y_i - \bar{y})^2$ is the total sum of squares (of responses $\boldsymbol{y}$) corrected for the mean $\bar{y} = \sum_{i=1}^{N} y_i$. The square of RMSE% essentially is a complement of R^2, since SST and SSY differ only by a constant, the mean $\bar{y}$.

Some model builders prefer the adjusted R^2 statistic,

$$R^2_{adj} = 1 - [\text{SSE}/(N-p)]/[\text{SST}/(N-1)],$$

because adding variables to the model will always increase R^2 regardless of whether or not the addition is statistically significant, whereas adding variables in general will not increase the adjusted R^2 statistic [MyersM95]. First, note that in analysis of variance, the SSE is considered to have $N-p$ "degrees of freedom" and the SST to have $N-1$ "degrees of freedom." Thus, $\text{SSE}/(N-p)$ and $\text{SST}/(N-1)$ are the so-called sums of squares in statistics. Next, note that this adjusted statistic is defined only for the conventional case of matrix X having full column rank p. Nevertheless, because of (12), we can extend it to the general case where X has rank r, full or not. Then the square of StDev% is essentially a complement of such a yet-to-be-extended R^2_{adj} statistic if it has to lose the mean $\bar{y}$ and one degree of freedom from its SSY. Moreover, StDev% is much closer to common sense, having a much more intuitively obvious meaning.

4.3. Computing steps. By SVD Property 1, the number of nonzero singular values determines the rank. It is not nearly as clear-cut in the context of computation on a digital computer, though, because zero singular values may actually appear as small nonzero numbers due to finite precision of the floating-point operations. (Bounded by finite precision, computation must be done in the presence of rounding or truncation error at each stage.) One approach, adapted from [GolubV83, p. 176; KlemaL80], is to have a parameter $\zeta > 0$ and a convention that the $N \times p$ matrix X has "numerical rank" r if the computed singular values are such that $s_1 \geq \cdots \geq s_r > \zeta \geq s_{r+1} \geq \cdots$ or has "numerical rank" $r = \min\{N, p\}$ if all the computed singular values are greater than ζ. The parameter ζ should be chosen consistent with both the precision of the specific machine and the numerical magnitude of the specific test matrix. After having gotten some experience, we have no difficulty setting such a parameter for automatic determination of the appropriate numerical rank.

In summary, we propose the following procedure for computing the LS fitting coefficients in the general case where the test matrix X may or may not have full rank.

ALGORITHM 1.

1. Form the matrix X as usual.
2. Perform SVD of X and obtain matrices U, V, and S.
3. Determine the numerical rank r of X based on the computed singular values.
4. Determine the first (largest) r computed singular values s_j, the first r columns $\boldsymbol{u}_j$ of U, and the first r columns $\boldsymbol{v}_j$ of V.
5. Compute the regression coefficients: $\boldsymbol{b} = \sum_{j=1}^{r} (\boldsymbol{u}_j^T \boldsymbol{y}/s_j)\boldsymbol{v}_j$.
6. Check if RMSE% and StDev% are small and acceptable. If not, repeat the procedure for a higher order.

Though not required by this algorithm, it is a good common RSM practice to convert natural variables ξ_j in natural units of measurement to "coded" variables—each being dimensionless with zero mean and same standard deviation—and use the coded variables as test variables x_j to form the matrix X. A standard procedure is to shift each natural variable ξ_j to the center between its maximum and minimum (in the N observations) and then scale it by half of the range between its maximum and minimum.

5. Number of coefficients. The (total) number of coefficients in a k-variable quadratic polynomial is well known to be $(k+1)(k+2)/2$. In this section, we derive similar general expressions for higher degrees and discuss how model order is limited by given numbers of test variables and test responses through the number of coefficients. That these formulas are useful to making D-optimal design of experiments is self-evident. First, we cite four known summation formulas.

LEMMA 2 (see [KornK68]).

(a) $\sum_{j=1}^{k} j = k(k+1)/2$.
(b) $\sum_{j=1}^{k} j^2 = k(k+1)(2k+1)/6$.
(c) $\sum_{j=1}^{k} j^3 = k^2(k+1)^2/4$.
(d) $\sum_{j=1}^{k} j^4 = k(k+1)(2k+1)(3k^2+3k-1)/30$.

Then we derive five new useful formulas.

THEOREM 5. Let p_m denote the number of coefficients in a k-variable polynomial of degree m. Then

(a) $p_3 = (k+1)(k^2+5k+6)/6$;
(b) $p_4 = (k+1)(k^3+9k^2+26k+24)/24$;
(c) $p_5 = (k+1)(k^4+14k^3+71k^2+154k+120)/120$.
(d) $p_m = (k+1)(k+2)\dots(k+m)/m!$
(e) $p_m = p_{m-1}(k+m)/m$.

PROOF. First of all, a polynomial of degree 2 can be reexpressed as follows:

$$\begin{aligned} P_2(x_1, \dots, x_k) &= \beta_0 + \sum_{j_1=1}^{k} \beta_{j_1} x_{j_1} + \sum_{j_1=1}^{k} \sum_{j_2 \le j_1} \beta_{j_1 j_2} x_{j_1} x_{j_2} \\ &= \beta_0 + \sum_{j_1=1}^{k} x_{j_1} \left[\beta_{j_1} + \sum_{j_2 \le j_1} \beta_{j_1 j_2} x_{j_2} \right] \\ &= \beta_0 + \sum_{j_1=1}^{k} x_{j_1} P_{1:j_1}(x_1, \dots, x_k), \end{aligned}$$

where $P_{1:j_1}$ denotes a polynomial of degree 1 having the first j_1 of the k variables.

This linear polynomial, $P_{1:j_1}$, obviously has $(1+j_1)$ terms. Therefore, the number of coefficients in polynomial P_2 is

$$p_2 = 1 + \sum_{j_1=1}^{k} (1+j_1) = 1 + k + k(k+1)/2 = (k+1)(k+2)/2,$$

as is well known.

A polynomial of degree 3 can be reexpressed in a similar manner:

$$\begin{aligned} P_3(x_1, \ldots, x_k) &= \beta_0 + \sum_{j_1=1}^{k} \beta_{j_1} x_{j_1} + \sum_{j_1=1}^{k} \sum_{j_2 \le j_1} \beta_{j_1 j_2} x_{j_1} x_{j_2} \\ &\quad + \sum_{j_1=1}^{k} \sum_{j_2 \le j_1} \sum_{j_3 \le j_2} \beta_{j_1 j_2 j_3} x_{j_1} x_{j_2} x_{j_3} \\ &= \beta_0 + \sum_{j_1=1}^{k} x_{j_1} \left[\beta_{j_1} + \sum_{j_2 \le j_1} \beta_{j_1 j_2} x_{j_2} + \sum_{j_2 \le j_1} \sum_{j_3 \le j_2} \beta_{j_1 j_2 j_3} x_{j_2} x_{j_3} \right] \\ &= \beta_0 + \sum_{j_1=1}^{k} x_{j_1} P_{2:j_1}(x_1, \ldots, x_k), \end{aligned}$$

where $P_{2:j_1}$ denotes a polynomial of degree 2 in the first j_1 of the k variables. By the quadratic formula, polynomial $P_{2:j_1}$ has $(j_1+1)(j_1+2)/2$ terms. Thus, the number of coefficients in polynomial P_3 is

$$p_3 = 1 + \sum_{j_1=1}^{k} (j_1+1)(j_1+2)/2 = 1 + \sum_{j_1=1}^{k} (j_1^2 + 3j_1 + 2)/2 = (k+1)(k^2+5k+6)/6,$$

where Lemma 2(a) and (b) were applied. This proves part (a) of this theorem.

Similarly, a polynomial of degree 4 can be expressed in terms of polynomials of degree 3:

$$\begin{aligned} P_4(x_1, \ldots, x_k) &= \beta_0 + \sum_{j_1=1}^{k} \beta_{j_1} x_{j_1} + \sum_{j_1=1}^{k} \sum_{j_2 \le j_1} \beta_{j_1 j_2} x_{j_1} x_{j_2} \\ &\quad + \sum_{j_1=1}^{k} \sum_{j_2 \le j_1} \sum_{j_3 \le j_2} \beta_{j_1 j_2 j_3} x_{j_1} x_{j_2} x_{j_3} \\ &\quad + \sum_{j_1=1}^{k} \sum_{j_2 \le j_1} \sum_{j_3 \le j_2} \sum_{j_4 \le j_3} \beta_{j_1 j_2 j_3 j_4} x_{j_1} x_{j_2} x_{j_3} x_{j_4} \\ &= \beta_0 + \sum_{j_1=1}^{k} x_{j_1} P_{3:j_1}(x_1, \ldots, x_k) \end{aligned}$$

with $P_{3:j_1}$ being a polynomial of degree 3 in first j_1 of the k variables. By the cubic

formula, the number of coefficients in polynomial P_4 is therefore

$$\begin{aligned} p_4 &= 1 + \sum_{j_1=1}^{k} (j_1 + 1)(j_1^2 + 5j_1 + 6)/6 \\ &= 1 + \sum_{j_1=1}^{k} (j_1^3 + 6j_1^2 + 11j_1 + 6)/6 \\ &= (k + 1)(k^3 + 9k^2 + 26k + 24)/24 \end{aligned}$$

after applying Lemma 2(a), (b), and (c). This proves part (b) of the theorem.

Reexpressing a polynomial of degree 5 in terms of polynomials of degree 4 likewise and using the quartic formula, we have its number of coefficients as

$$\begin{aligned} p_5 &= 1 + \sum_{j_1=1}^{k} (j_1 + 1)(j_1^3 + 9j_1^2 + 26j_1 + 24)/24 \\ &= 1 + \sum_{j_1=1}^{k} (j_1^4 + 10j_1^3 + 35j_1^2 + 50j_1 + 24)/24 \\ &= (k + 1)(k^4 + 14k^3 + 71k^2 + 154k + 120)/120 \end{aligned}$$

by using all parts of Lemma 2. Part (c) of the theorem is thus proved. Parts (d) and (e) are readily evident by induction after factorizing the expressions in parts (a) to (c). □

Flight experiments, wind-tunnel tests, and many complex or large-scale engineering experiments cannot afford frequent reruns, or modified reruns, because their setup and execution usually require commitment and consumption of a great deal of material and human resources. To be cost, time, and resource effective, sufficient and even redundant data often are obtained at once. Such tests tend to be thorough on certain few crucial variables so as to get as clear a picture of their impact as possible. When building regression models to fit those response data, higher-order models naturally are more appropriate for capturing or reflecting the characteristics better. How high order can a model be? It depends on both the number of responses and the number of test variables. As an extended conventional rule of thumb, the number of coefficients (i.e., model parameters) should be less than the number of responses. Now Theorem 5 indicates that the number of coefficients increases rapidly as the number of test variables, k, increases (see also Table 1) essentially at the rate of power k^m, given the order of the response model, m. Thus, given the number of test variables, the order of the response model is severely limited by the number of responses, N, given. With three test variables, for instance, the highest should be second order for 11 up to 20 responses, third order for 21 up to 35 responses, and so on; see Table 1. The test matrix X formed by specific values of the test variables may further dictate that the highest order be even lower to ensure it has full column rank.

Not all terms are equally needed for modeling a specific set of responses. Nevertheless, we can first build up to a desirable highest order and then trim off the redundant or insignificant terms afterward. Because Algorithm 1 handles rank deficiency effectively, a response model actually can be, unconventionally, of a few orders higher than both the number of responses and the rank of the test matrix would dictate (see Examples 4 and 6 in section 7). Suppose we want to use no more terms than the full rank of the test matrix while using important higher-order terms. We

TABLE 1
Number of coefficients.

k	p_2	p_3	p_4	p_5
1	3	4	5	6
2	6	10	15	21
3	*10*	*20*	*35*	*56*
4	15	35	70	126
5	21	56	126	252
6	28	84	210	462
7	36	120	330	792
8	45	165	495	1,287
9	55	220	715	2,002
10	66	286	1,001	3,003

can use Algorithm 2A (in section 6) to trim the terms of a rank-deficient higher-order model down to a full-rank state while keeping whatever higher-order terms that are highly important (see Example 6).

6. Trimming fitted polynomials. We often do not know what factors (powers or cross-products of the test variables) would actually contribute significantly in approximating the responses. Some model builders may toss in a few trial factors at a time; some others may simply include all possible factors for a given order. In any way, we may want to truncate the resulting polynomial afterward if we wish to use fewest possible terms, especially when some are insignificant or redundant. In the extended RSM, there are two trimming phases, full rank and rank deficient, each with a different intent. These are the associated Issues 2 and 3, respectively.

6.1. Trimming in rank-deficient or near rank-deficient phase. Suppose the matrix X is rank deficient or nearly rank deficient. Denote by r the empirical, numerical rank and by $s_r(X)$ the corresponding smallest computed singular value. (This means that there are possibly more than r computed singular values but that all that are smaller than $s_r(X)$ are regarded as being zero.) Rank deficiency implies redundancy among the terms that compose the approximating polynomial. If a coefficient is zero, the corresponding term obviously is redundant, but redundant terms may have nonzero coefficients. Nevertheless, we may not be interested in a polynomial that involves all of the redundant terms. We may instead seek a polynomial that has at most r nonzero coefficients. The position of the nonzero coefficients in the polynomial determines which columns of matrix X are to be used to approximate the responses.

A procedure proposed by Golub, Klema, and Stewart is, after the numerical rank of X is determined through SVD of X, as follows [GolubV83, p. 415]. (a) Calculate a permutation matrix P such that the columns of the $N \times r$ submatrix Z_1 in the partition $XP = [Z_1, Z_2]$ are sufficiently independent. (b) Approximate the responses with $Z_1\boldsymbol{\gamma}$, where $\boldsymbol{\gamma}$ is an r-vector of coefficients to be determined. Step (a) needs elaboration. The selection of r columns from matrix X should be best if they align with response vector $\boldsymbol{y}$ the closest possible because the LS criterion is to minimize deviations from the responses. Such a subset of columns, however, may be very weakly independent, if not linearly dependent. The resulting submatrix Z_1 may therefore be nearly rank deficient, or even rank deficient. Thus, there should be a trade-off between the independence of the chosen columns and the smallness of the residuals they render. Golub, Klema, and Stewart then proposed the following algorithm. Use QR factorization with column pivoting to compute $Q^T[V_{11}^T, V_{21}^T]P = [R_{11}, R_{12}]$, where V_{ij} are blocks of orthogonal matrix V. Then pick the first r columns of X

according to the resulting permutation matrix P as Z_1. Golub, Klema, and Stewart's procedure is particularly suitable for rank-deficient and nearly rank-deficient matrices. See [LawsoH74] for discussions on many other earlier methods of subset selection. The problem of subset selection is to delete some variables β_j from system (9), or select a subset of columns of matrix X, to find the smallest number of variables that can be used in the LS solution and still give acceptably small residuals.

The use of QR with column pivoting is to choose such a permutation P that will produce a matrix $\hat{V}_{11}$ (an upper principal block of product P^TV) as well conditioned as possible [GolubV83, p. 416] so that Z_1 would not be rank deficient (see related discussion shortly). Observe, however, that the choice of r columns of X this way clearly has no connection whatsoever to response vector $\boldsymbol{y}$ so as to lead to any such trade-off. Nevertheless, we can improve their algorithm for our trimming purpose as follows.

Suppose the responses have been fitted with $X\boldsymbol{b}$ and we wish to eliminate $p-r$ terms of the resulting approximating polynomial. Note that it is the whole term—the coefficient together with the self- or cross-product of the variables—that participates as a part of the polynomial in minimizing deviations from the responses. Thus, the columns X_j of matrix X alone do not adequately represent the terms in the approximating polynomial. Those columns of the matrix modified by the appropriate coefficients, namely, $b_1X_1, \ldots, b_jX_j, \ldots, b_pX_p$, do. For these particular, LS fitted, coefficients b_j connect the columns of X to the responses (through the normal equations, of course). Thus, we let diagonal matrix D of order r represent these coefficients, $D = \operatorname{diag}\{b_1, \ldots, b_j, \ldots, b_p\}$, and develop the following improved mathematical machinery.

THEOREM 6. $N \times r$ $\hat{Z}_1$ $XDP = [\hat{Z}_1, \hat{Z}_2]$ P r P^TDP $\hat{D}_1 = \operatorname{diag}\{\hat{b}_1, \ldots, \hat{b}_r\}$ $p \times p$ P^TV $\hat{V}_{11}$ r, P^TV $\begin{bmatrix} \hat{V}_{11} & \hat{V}_{12} \\ \hat{V}_{21} & \hat{V}_{22} \end{bmatrix}$ $\hat{V}_{11}$ $\hat{D}_1$ $s_r(\hat{Z}_1)$ $\hat{Z}_1$ $s_r(\hat{Z}_1) \geq s_r(X) \min\{|\hat{b}_1|, \ldots, |\hat{b}_r|\}/\|\hat{V}_{11}^{-1}\|$

. Let $\boldsymbol{w}$ be a unit r-vector such that $\|\hat{Z}_1\boldsymbol{w}\| = s_r(\hat{Z}_1)$. First, by definition of $\hat{Z}_1$ and $\hat{D}_1$,

$$\hat{Z}_1\boldsymbol{w} = [\hat{Z}_1, \hat{Z}_2] \begin{bmatrix} \boldsymbol{w} \\ 0 \end{bmatrix} = XDP \begin{bmatrix} \boldsymbol{w} \\ 0 \end{bmatrix} = XP(P^TDP) \begin{bmatrix} \boldsymbol{w} \\ 0 \end{bmatrix} = XP \begin{bmatrix} \hat{D}_1\boldsymbol{w} \\ 0 \end{bmatrix}.$$

Then, by (5) and the partitioning of P^TV,

$$\begin{aligned} \hat{Z}_1\boldsymbol{w} &= USV^TP \begin{bmatrix} \hat{D}_1\boldsymbol{w} \\ 0 \end{bmatrix} = US \begin{bmatrix} \hat{V}_{11}^T & \hat{V}_{21}^T \\ \hat{V}_{12}^T & \hat{V}_{22}^T \end{bmatrix} \begin{bmatrix} \hat{D}_1\boldsymbol{w} \\ 0 \end{bmatrix} \\ &= U \begin{bmatrix} S_r & 0 \\ 0 & S_z \end{bmatrix} \begin{bmatrix} \hat{V}_{11}^T\hat{D}_1\boldsymbol{w} \\ \hat{V}_{12}^T\hat{D}_1\boldsymbol{w} \end{bmatrix} = U \begin{bmatrix} S_r\hat{V}_{11}^T\hat{D}_1\boldsymbol{w} \\ S_z\hat{V}_{12}^T\hat{D}_1\boldsymbol{w} \end{bmatrix}, \end{aligned}$$

where S_z is the diagonal matrix of the remaining $p-r$ computed singular values. Because U is orthogonal,

$$\|\hat{Z}_1\boldsymbol{w}\|^2 = \|S_r\hat{V}_{11}^T\hat{D}_1\boldsymbol{w}\|^2 + \|S_z\hat{V}_{12}^T\hat{D}_1\boldsymbol{w}\|^2 \geq s_r(X)^2\|\hat{V}_{11}^T\hat{D}_1\boldsymbol{w}\|^2.$$

Now, because both $\hat{V}_{11}$ and $\hat{D}_1$ are nonsingular, first we can write

$$\boldsymbol{w} = (\hat{V}_{11}^T\hat{D}_1)^{-1}\hat{V}_{11}^T\hat{D}_1\boldsymbol{w}.$$

Taking the norm,

$$1 = \|\boldsymbol{w}\|^2 = \|(\hat{V}_{11}^T \hat{D}_1)^{-1}\, \hat{V}_{11}^T \hat{D}_1 \boldsymbol{w}\|^2 \le \|(\hat{V}_{11}^T \hat{D}_1)^{-1}\|^2\, \|\hat{V}_{11}^T \hat{D}_1 \boldsymbol{w}\|^2.$$

Since

$$\begin{aligned}\|(\hat{V}_{11}^T \hat{D}_1)^{-1}\| \le \|\hat{V}_{11}^{-1}\| \cdot \|\hat{D}_1^{-1}\| &\le \|\hat{V}_{11}^{-1}\| \max\{1/|\hat{b}_1|, \ldots, 1/|\hat{b}_r|\} \\ &= \|\hat{V}_{11}^{-1}\| / \min\{|\hat{b}_1|, \ldots, |\hat{b}_r|\},\end{aligned}$$

then we have

$$\|\hat{V}_{11}^T \hat{D}_1 \boldsymbol{w}\| \ge 1/\|(\hat{V}_{11}^T \hat{D}_1)^{-1}\| \ge \min\{|\hat{b}_1|, \ldots, |\hat{b}_r|\}/\|\hat{V}_{11}^{-1}\|.$$

Consequently,

$$s_r(\hat{Z}_1) = \|\hat{Z}_1 \boldsymbol{w}\| \ge s_r(X) \|\hat{V}_{11}^T \hat{D}_1 \boldsymbol{w}\| \ge s_r(X) \min\{|\hat{b}_1|, \ldots, |\hat{b}_r|\}/\|\hat{V}_{11}^{-1}\| \qquad \square$$

Theorem 6 implies that in order for the smallest singular value $s_r(\hat{Z}_1)$ to be bounded significantly above zero, the permutation matrix P needs not only to make $\hat{V}_{11}$ as well conditioned as possible (so that its inverse does not have a very large norm) but also to get each of the new leading r coefficients $\hat{b}_1, \ldots, \hat{b}_r$ as large as possible. Accordingly, we trade off some linear independence among the selected columns in some implicit favor (conveyed via the permuted coefficients) of participation in minimizing the deviations, $\boldsymbol{y} - \hat{Z}_1 \boldsymbol{\gamma}$, by incorporating the fitted coefficients into the choice for permutation P. Then those coefficients that correspond to the columns in $\hat{Z}_1$ chosen from according to P are retained. Others are lopped off (or set to zero).

It is a good idea to obtain an adjustment factor for each retained coefficient so that the loss of minimality in the residuals due to truncation can be made up as much as possible by the retained. Packing all above into a simple procedure, we propose the following trimming algorithm.

ALGORITHM 2A.

(a) Use QR factorization with column pivoting to compute $Q^T[V_{11}^T, V_{21}^T]DP = [R_{11}, R_{12}]$.
(b) Pick the first r columns of X according to the resulting permutation matrix P and form Z_1.
(c) Refit the responses with matrix Z_1 by applying Algorithm 1.

6.2. Trimming in full-rank phase. The conventional approach is to drop those factors whose coefficients are tested statistically insignificant. The tests—Student's t-test on individual coefficients and the F-test on groups of coefficients—are based on the assumption that the uncertainties are normally distributed. For illustrations and more details, see RSM texts [MyersM95, KhuriC96]. If we can hardly justify even the basic assumption that the uncertainties have zero mean and zero correlation, additional assumptions such as their normal distribution will be harder still to justify.

An alternative is to base the trimming on mathematical properties of the approximating polynomial. One obvious approach is to judge by the magnitude of the coefficients. This is not a good general idea, however, because it is not the coefficient alone but rather the whole term—the product of the coefficient with the corresponding factor—that participates in the polynomial approximation. Instead, we propose that those terms having low percentage of usage be trimmed off. Compute the norm of vector $b_j X_j$ (i.e., column j of matrix X multiplied by the corresponding coefficient

b_j) and then trim it off if the norm is less than a certain small percentage of the norm of the response vector $\boldsymbol{y}$.

Specifically, define the utility of term j by $u_j = 100\|b_j X_j\|/\|\boldsymbol{y}\|$. Assume that all terms with utility less than $u_t\%$ are trimmed off. Denote by $\boldsymbol{e}_t$ the residuals after the trimming. Then $\boldsymbol{e} = \boldsymbol{y} - X\boldsymbol{b} = \boldsymbol{e}_t - \sum_{u_j<u_t} b_j X_j$. Hence,

$$\|\boldsymbol{e}_t\| = \|\boldsymbol{e} + \sum_{u_j<u_t} b_j X_j\| \le \|\boldsymbol{e}\| + \left\| \sum_{u_j<u_t} b_j X_j \right\| \le \|\boldsymbol{e}\| + \sum_{u_j<u_t} \|b_j X_j\|.$$

Let $\mathrm{RMSE}_t\%$ represent the relative error corresponding to $\boldsymbol{e}_t$ by the same definition, (13). Then the increase in the relative modeling error due to truncation is

$$\begin{aligned} \mathrm{RMSE}_t\% - \mathrm{RMSE}\% &= 100(\|\boldsymbol{e}_t\|/\|\boldsymbol{y}\| - \|\boldsymbol{e}\|/\|\boldsymbol{y}\|) \\ &\le 100 \left\| \sum_{u_j<u_t} b_j X_j \right\| /\|\boldsymbol{y}\| \\ &\le \sum_{u_j<u_t} 100\|b_j X_j\|/\|\boldsymbol{y}\| = \sum_{u_j<u_t} u_j. \end{aligned}$$

It means that the increase is smaller than the sum of the utility of all the lopped-off terms. It also implies that the actual increase can even be substantially smaller since components of the dropped vectors $b_j X_j$ may have cancelled each other out, leaving a much smaller net effect than estimated by the sum of their individual norms.

Again, it is a good idea to refit the truncated polynomial to adjust the coefficients of the retained factors so that what has been trimmed off can be made up as much as possible. Therefore, we propose the following algorithm.

ALGORITHM 2B.

(a) Compute utility $u_j = 100\|b_j X_j\|/\|\boldsymbol{y}\|$ of each term j.
(b) Trim off all terms with utility less than $u_t\%$.
(c) Form submatrix Z_1 by deleting the corresponding columns of X.
(d) Refit the responses with matrix Z_1 by applying Algorithm 1.

7. Examples of application. We demonstrate by examples that the proposed SVD technique works as well as the current RSM while expanding and enabling the RSM to handle reliably and efficiently (a) any test or design matrix X that is rank deficient or nearly so and (b) unconventional higher-order response modeling. Examples 1 and 2 are "apples-to-apples" comparisons of the proposed technique on the conventional ground with existing results of other methods. Examples 2 and 3 also show how the new fitness measure StDev% is useful for judging whether adding higher-order terms is necessary, just as the adjusted R^2 statistic is in the conventional RSM applications. Two hypothetical cases (Examples 4 and 5) show that if the unknown response function is actually a polynomial, the proposed technique can determine accurately both the degree and the coefficients whether the matrix X has full column rank or a large rank deficiency. Example 6 is a real hardware test case with a truly unknown response function, the case that produced the pressure profiles of Figure 1. It shows that by using the proposed technique, we are able to model the pressure responses to a desirable higher order without difficulty and that the predictive capability is confirmed by three responses set aside. In addition, we get an interesting three-dimensional response surface representation that makes the flow separation and outward flow migration rather obvious.

Example 1. This is an apples-to-apples comparison with existing results of a conventional method to show that the proposed SVD technique produces computed coefficients and prediction results essentially the same as the conventional results even not using the same computer. It also shows that the coefficients trimmed by the proposed mathematical alternative criterion yield good results.

[SteffSY00] is a good documentary reference for its careful, comprehensive application of the conventional RSM to a complex engineering problem. In particular, it contains numerical details we can use directly for the comparison. They applied RSM designs and Navier–Stokes calculations to analyze the nozzle efficiency of rocket-based combined-cycle (RBCC) systems operating in an all-rocket mode over a broad range of parameters. They varied six independent parameters (four geometric and two flow field variables) simultaneously to study their effects upon system performance. A full parametric study of these six variables, each at three levels, would necessitate an enormous computational effort, a total of 729 possible computational fluid dynamics (CFD) simulation runs. Through a D-optimal design strategy, only 36 separate CFD solutions are required for developing a full response surface (RS) model of conventionally the highest (i.e., second) order.

Their RS model contained initially an intercept, six linear, 15 bilinear or interaction, and six square terms. Of the 28 coefficients, only 13 were retained in the final form because of statistical significance. Their final step in the design of experiments involved confirmation tests with eight new cases (37 to 44). All but case 37 were well within the approximate uncertainty band of $+/-1\%$ in the specific-impulse efficiency; case 37 results lay just outside the uncertainty band. Their D-optimal design construction and RSM model analyses were executed with the RS/1 software of Bolt, Beranek, and Newman, Inc.

The test points are the 36 cases of settings in their Table 3 for independent (natural) variables $x_1, \ldots, x_6$. The response variable y is the vacuum specific impulse efficiency, η_{Isp}, calculated with the S-A turbulence model in the CFD simulations. Its 36 response values are in the rightmost column of their Table 5. As in [SteffSY00], we convert each of the six independent variables to take on values only between -1 and +1 as a coded variable. Then we form the 36x28 matrix X for second-order regression, apply the SVD technique (Algorithm 1), and compute the 28 coefficients that fit the 36 responses.

(a) Conventional trimming. Not all terms in the full model are significant in fitting the responses, especially those with negligible coefficients. A conventional approach uses Student's t-statistic to test the hypothesis that an individual coefficient is zero. Accordingly, they determined with the RS/1 software that 15 of the 28 coefficients failed to reject the null-hypothesis with a certain level of confidence. They lopped off these 15 terms of their polynomial. First, we similarly lop off these 15 terms of ours by t-statistic. Then we drop those columns of the matrix X that correspond to the 15 lopped-off coefficients. We treat the thinned matrix as if it were a new 36x13 matrix X, apply the SVD technique again, and compute new 13 coefficients that fit the same 36 responses. In other words, our truncated polynomial is refitted to the 36 responses. The retained 13 coefficients of theirs and of ours are listed next to each other in Table 2. They are essentially identical to each other.

They used eight additional cases (37 to 44) for confirming the predicative capability of their truncated polynomial. We likewise subject our truncated polynomial to the same confirmation tests. The results are listed next to theirs in Table 3, where y^{CFD} stands for the vacuum specific impulse efficiency they calculated in the CFD

TABLE 2
Computed coefficients—Example 1.

	1	x_1	x_3	x_4	x_5	x_6	x_1x_3	x_2^2
Existent	89.3250	0.3503	-1.6150	-0.6976	1.6077	3.2062	0.3344	0
SVD (t-statistic)	89.330505	0.349373	-1.615697	-0.697912	1.607320	3.205373	0.333971	0
SVD (utility)	89.387012	0.324743	-1.701886	-0.833206	1.548222	3.190109	0	0.272434
	x_2x_3	x_2x_5	x_2x_6	x_3^2	x_3x_5	x_5^2	x_5x_6	x_6^2
Existent	0.3626	-0.4209	-0.7228	0	0.2830	0	-0.4061	-1.5860
SVD (t-statistic)	0.362972	-0.420155	-0.722275	0	0.283101	0	-0.406171	-1.590754
SVD (utility)	0	-0.493062	-0.591742	0.618907	0	-0.862915	-0.409170	-1.735099

TABLE 3
Prediction results—Example 1.

							Existent			SVD (t-statistic)		SVD (utility)	
Case	x_1	x_2	x_3	x_4	x_5	x_6	y^{CFD}	y^{RS}	Dif	y^P	Dif	y^P	Dif
37	750	0	120	3.5	2	12	89.94	91.35	-1.41	91.347	-1.407	90.827	-0.887
38	750	0	120	3.5	1	12	86.98	87.30	-0.32	87.289	-0.309	86.745	0.235
39	750	8	120	3.5	2	12	89.65	90.51	-0.86	90.512	-0.862	89.841	-0.191
40	750	8	120	3.5	1	12	87.58	88.14	-0.56	88.134	-0.554	87.731	-0.151
41	750	0	120	3.5	1.5	20	91.29	91.67	-0.38	91.655	-0.365	91.696	-0.406
42	750	0	120	3.5	1.5	4	83.46	83.81	-0.35	83.800	-0.340	84.132	-0.672
43	750	8	120	3.5	1.5	20	89.69	90.22	-0.53	90.215	-0.525	90.512	-0.822
44	750	8	120	3.5	1.5	4	85.39	85.26	0.13	85.249	0.141	85.315	0.075
								Dif%:	0.773		0.769		0.596

simulations and y^{RS} the response they predicted using their RS model. The Dif% at the bottom represents the RMS value of the eight prediction differences (Dif) relative to the RMS value of the eight CFD responses (y^{CFD}) and is expressed in percentage. Our prediction results are headed by SVD (t-statistic) with y^P standing for the predicted response by our SVD model. The comparison shows that the proposed SVD approach is at least as good as the conventional one, having the same good performance that all but one case are within the $+/-1\%$ uncertainty band.

Both our computed coefficients and our prediction results are essentially the same as theirs in [SteffSY00]. What this means is that the SVD approach is at least as good as the conventional on the same ground.

(b) Unconventional truncation. They made a conscious effort—an unusually good engineering practice—to control the sources of computational bias and defend their assumed uncertainty estimate, but still nowhere in their paper, not unusual in application papers (not their fault, either), does it show that it is valid to assume that the uncertainties are normally distributed. Instead of using t-statistic, which is based on such a normality assumption, an alternative is to truncate the polynomial according to the mathematical utility of the terms. We apply Algorithm 2B and truncate 28 coefficients also down to 13. These 13 coefficients also are listed in Table 2 but headed by SVD (utility). The main difference from the conventionally truncated counterpart is the replacement of three bilinear terms involving x_3 (i.e., x_1x_3, x_2x_3, x_3x_5) by three square terms (x_2^2, x_3^2, x_5^2).

The corresponding prediction results are listed in the last two columns of Table 3, headed by SVD (utility). Observe that such coefficients yield better results: All prediction differences (Dif) are within the $+/-1\%$ uncertainty band!

Example 2. A regression model need not be of a higher order to be better. The transistor example used throughout the textbook [MyersM95] is one that first order is superior to second order in fitting the transistor responses. The matrix X has full

column rank for both orders, and the conventional RS method is well demonstrated in the book. Here we apply the SVD technique for another apples-to-apples comparison.

The 14 responses (transistor gain in an integrated circuit device) to two variables (emitter drive-in time and emitter dose) are given in their Table 2.2. They formed the 14x3 matrix X for the first-order regression, multiplied the normal equations by the inverse of product X^TX, and thereby obtained the coefficients $\boldsymbol{b} = (1242.3057, 323.4366, -54.7691)$ and the estimated variance $\hat{\sigma}^2 = 1,220.1$. We form the same 14x3 matrix X but apply the proposed SVD technique and obtain the coefficients $\boldsymbol{b} = (1242.314755, 323.434737, -54.773595)$ and the estimated variance $\hat{\sigma}^2 = 1,220.101394$, which are virtually identical to what they obtained by the conventional method.

They added quadratic terms (x_1^2, x_1x_2, and x_2^2) and then fitted the same responses with a second-order model. The new estimated variance was larger: $\hat{\sigma}^2 = 1,559.975$. They regarded it as "suggesting that the first order model is superior in that there is less unexplained variability." Likewise, we fit the same responses with a second-order model by the SVD technique. Since the estimated variance we produced, $\hat{\sigma}^2 = 1,559.9785$, is also virtually identical to theirs, the conclusion is the same: the first-order model is superior to the second-order model.

They also demonstrated the above conclusion by using the adjusted R^2 statistic on both models. The statistic we computed with the SVD models is exactly the same as theirs when also rounded to 4 digits:

$$R^2_{adj} = 0.9762(\text{1st order}); \qquad R^2_{adj} = 0.9695(\text{2nd order}).$$

The decrease in R^2_{adj}, from first order to second order, is as stated in their text: "a strong indication that the quadratic terms are unnecessary."

The fitness measure StDev% of the SVD models is 0.735932 for first order but 0.832145 for second order. It is intuitively obvious that the increase in the spread (i.e., StDev%) indicates correctly that the quadratic terms are unnecessary.

Example 3. The curious reader may wonder if second order is superior to first order in the case of Example 1. Here we also form the 36x7 matrix X for first-order regression in addition to the previous 36x28 matrix X, apply the SVD technique, and obtain the following fitness measures and adjusted R^2 statistic on both orders:

[illegible]: RMSE% = 1.470005, StDev% = 0.272973, $R^2_{adj} = 0.893149$
[illegible]: RMSE% = 0.408067, StDev% = 0.144274, $R^2_{adj} = 0.970152$

The second-order model is clearly both necessary and superior because it essentially halves the spread (StDev%) of the first-order model to nearly perfect 0.14%, in addition to having a very impressive 0.41% RMS error, almost four times smaller than that of the first-order model. Correspondingly in the conventional front, the relatively large increase in the adjusted R^2 statistic from the first to the second order also indicates strongly that quadratic terms are necessary.

Example 4. This example illustrates how given responses are modeled from a full-rank first-order case through a rank-deficient fifth-order case of test matrix X. Suppose there are 18 test points in two variables with 7 levels in variable ξ_1, starting from $\xi_1 = 0.70$ and stepping up by 1.375, and 3 levels in variable ξ_2, starting from $\xi_2 = -5$ and stepping up by 5. (Three points, (2.075, 5), (3.45, −5), and (6.2, 5), are not included.) The natural variables are converted to coded variables and used as x_1

TABLE 4(a)
Modeling summary—Example 4.

Order	#Coef	Rank	Full?	RMSE%	StDev%
1	3	3	yes	22.500873	5.809700
2	6	6	yes	19.242467	5.554822
3	10	9	No	15.408708	5.136236
4	15	12	No	12.890470	5.262512
5	21	15	No	0.000002	0.000001

TABLE 4(b)
Computed singular values—Examples 4 *and* 6.

Order	Singular values									
1	4.264	3.296	2.935							
2	5.540	3.315	2.949	2.328	1.828	1.459				
3	5.604	4.941	4.302	2.344	1.830	1.459	1.301	1.214	0.6361	1.55E-16
4	6.836	4.996	4.337	3.823	2.563	2.094	1.303	1.215	0.7102	0.6446
	0.5253	0.2673	3.33E-16	1.38E-16	8.09E-17					
5	6.942	6.355	5.628	3.859	2.568	2.138	2.094	1.840	1.018	0.7101
	0.5332	0.3264	0.2966	0.1970	7.92E-02	3.81E-16	1.82E-16	6.05E-17		

and x_2 in the regression:

$$x_1 = (\xi_1 - 4.825)/4.125, \qquad x_2 = \xi_2/5.$$

Assume that the unknown response function is actually a two-variable polynomial of degree 5 with known coefficients:

$$\begin{aligned} P_5(x_1, x_2) = {} & 5.186855 - 13.933827x_1 + 0.135362x_2 + 7.685090x_1^2 - 0.019777x_1x_2 \\ & - 0.388437x_2^2 + 8.458755x_1^3 + 0.109929x_1^2x_2 + 5.363036x_1x_2^2 \\ & + 0.135362x_2^3 - 5.503517x_1^4 - 0.150153x_1^3x_2 + 0.741562x_1^2x_2^2 \\ & - 0.019777x_1x_2^3 - 0.388437x_2^4 - 0.681754x_1^5 - 0.593747x_1^4x_2 \\ & - 10.365834x_1^3x_2^2 + 0.109929x_1^2x_2^3 + 5.363036x_1x_2^4 + 0.135362x_2^5 \end{aligned} \tag{15}$$

Suppose 18 responses are thereby generated. Then we treat both the degree and the coefficients as if they were unknown and conduct the response modeling by the proposed SVD technique (Algorithm 1).

Table 4(a) summarizes the results with the fitness measures RMSE% and StDev%, where #Coef denotes the number of regression coefficients (i.e., p) and Rank the numerical rank of the matrix X (i.e., r). Table 4(b) lists the computed singular values of the matrix X for each order up to the fifth. First, observe that from the third order on, matrix X starts to have one or more nearly zero singular values, and the product X^TX therefore starts to have less than full rank. Note in particular that for the fifth order, $N < p$ (unconventionally) and the column-rank deficiency is large: $21 - 15 = 6$. Then observe that measure RMSE% decreases continuously until it suddenly becomes "zero" (0.000002) at the fifth order. In other words, the degree of the response polynomial is correctly indicated to be 5 despite rank deficiency in the corresponding matrix X.

Notice, however, that StDev% stops decreasing momentarily at fourth order, which might suggest that the fourth-order model is not really much better, but becomes "zero" (0.000001) at fifth order with a very large drop, which suggests (correctly) that it is worth continuing until fifth order. This also reflects (correctly) that an odd order fits these responses much better.

TABLE 4(c)
Computed coefficients—Example 4.

Order	1	x_1	x_2	x_1^2	x_1x_2	x_2^2	x_1^3	$x_1^2x_2$	$x_1x_2^2$	x_2^3	
1	6.549223	-6.176864	0.168112								
2	5.688723	-6.167538	0.157091	2.275374	-0.171268	-0.386195					
3	5.686959	-8.807759	0.071011	2.279343	-0.131571	-0.416629	1.254363	-0.030581	2.537936	0.071011	
4	5.029693	-9.156864	0.123258	9.001322	-0.420243	-0.356290	1.703212	-0.012769	2.351510	0.123258	
5	5.186855	-13.933827	0.135362	7.685090	-0.019777	-0.388437	8.458754	0.109929	5.363036	0.135362	
	x_1^4	$x_1^3x_2$	$x_1^2x_2^2$	$x_1x_2^3$	x_2^4	x_1^5	$x_1^4x_2$	$x_1^3x_2^2$	$x_1^2x_2^3$	$x_1x_2^4$	x_2^5
4	-6.741169	0.622442	0.803713	-0.420243	-0.356290						
5	-5.503518	-0.150153	0.741562	-0.019777	-0.388437	-0.681753	-0.593747	-10.365834	0.109929	5.363036	0.135362

Table 4(c) lists the computed coefficients for all these models. Notice that the computed coefficients for the fifth-order model are the same as assumed in (15) except some that are somewhat different in the sixth decimal place. (The SVD matrices computed and used in the third-order modeling are listed in the appendix.)

Example 5. A simple question of curiosity with regard to Example 4 is, What if there are more than 21 test points so that the resulting matrix X has full column rank for the fifth order? In this example, the number of data points is thus increased from 18 to 37. Before proceeding, it is worth mentioning that rank deficiency in matrix X is not unusual at all when increasing the 18 data points step by step. Near the end, after having additional two full columns of 7 points in the middle (i.e., at $\xi_2 = \pm 2.5$), the rank of the resulting 35x21 matrix X was 20, still one short of being full. Finally, after another two additional data points (rather arbitrarily at $\xi_1 = 0.70, \xi_2 = \pm 1.25$), the resulting 37x21 matrix X attained, marginally, its full column rank.

Now assume the same polynomial as in Example 4, (15), and suppose 37 responses are thereby generated. Again, we treat both the degree and the coefficients as if they were unknown and conduct the same response modeling as before. Table 5(a) summarizes the results with fitness measures RMSE% and StDev% for each model order up to fifth, whereas Table 5(b) lists the computed singular values of the matrix X for the fifth order. Both RMSE% and StDev% steadily decrease until suddenly dropping to "zero" (0.000002 and 0.000001, respectively) at the fifth order. Just as in Example 4, the computed coefficients for the fifth-order model (listed in Table 5(c)) are the same as the assumed except some that are slightly different in the sixth decimal place. Therefore, we conclude that if the unknown response is actually a polynomial of degree 5, the proposed SVD technique can model the responses as accurately when the corresponding matrix X is greatly rank deficient as when it has full column rank.

To summarize, this example shows that many more test points may be needed for the matrix X to have full column rank, but the full-rank results, when they fit the responses well, may have no noticeable difference from rank-deficient results for the same order.

TABLE 5(a)
Modeling summary—Example 5.

Order	#Coef	Rank	Full?	RMSE%	StDev%
1	3	3	yes	22.469425	3.853475
2	6	6	yes	19.202559	3.448881
3	10	10	yes	13.617129	2.620618
4	15	15	yes	11.727797	2.500375
5	21	21	yes	0.000002	0.000001

TABLE 5(b)
Computed singular values—Example 5 *(*5*th order).*

8.447	7.312	6.756	4.304	3.531	2.957	2.535	2.505	1.459	1.367	
0.8894	0.8500	0.5550	0.4511	0.4437	0.3929	0.3198	0.2533	0.2044	0.1506	3.66E-02

TABLE 5(c)
Computed coefficients—Example 5 *(*5*th order).*

1	x_1	x_2	x_1^2	x_1x_2	x_2^2	x_1^3	$x_1^2x_2$	$x_1x_2^2$	x_2^3	
5.186855	-13.933827	0.135361	7.685090	-0.019777	-0.38844	8.458753	0.109929	5.363038	0.13537	
x_1^4	$x_1^3x_2$	$x_1^2x_2^2$	$x_1x_2^3$	x_2^4	x_1^5	$x_1^4x_2$	$x_1^3x_2^2$	$x_1^2x_2^3$	$x_1x_2^4$	x_2^5
-5.503517	-0.150153	0.741562	-0.019777	-0.388436	-0.68175	-0.593747	-10.365834	0.109929	5.36303	0.135357

Example 6. Here we use a set of real hardware test results [KamhaKT03] as an example. NASA Glenn Research Center (GRC) has been developing a hydrogen-fueled single-thruster RBCC semiaxisymmetric propulsion system for single-stage-to-orbit vehicles [Trefn99]. For flight from lift-off to Mach 2.5, the propulsion system can operate in two different schemes: the simultaneous mixing and combustion (SMC) and the independently fueled ramjet stream. In the SMC scheme, the excess fuel from the rocket thruster mixes and burns with the unfueled air stream. Tests were conducted at the direct-connect facility of the GRC to determine the extent of mixing between the primary (rocket) stream and the secondary (unfueled air) stream to assess the feasibility of the SMC scheme in a short single-thruster flowpath.

Three vertical rakes (port, center, and starboard) with seven probes per rake were mounted at the end of the mixer-combustor for pressure measurements. The port and starboard rakes were 5 in. off the center rake. The seven probes are equally spaced 1.375 in. apart beginning with a probe 0.70 in. up from the mixer-combustor floor. There are totally 21 responses, each being the pitot-pressure reading of a probe. The three parametric curves in Figure 11 of [KamhaKT03] (see our Figure 1) represent the mixer-combustor exit pressure profiles of one of the tests with the rocket turned on. The profiles reveal a complex flow pattern. Near the floor, the rocket directly influences all rakes with similar high pressure; but near the cowl surface (i.e., top of the rakes), "the centerline distribution shows separation of the flow: some flow migrates to the outboard rakes" [KamhaKT03].

Generally speaking, it is desirable for systems evaluation and technology analysis of the RBCC engine, and eventually for design optimization of the space vehicle, if performance-related test results can be modeled analytically. More specifically, the engineers may like to visualize or get some idea of how the pressure might be distributed between the rakes if they would place more columns of such probes there. Since the test variables are spatial coordinates (height and side of the test location) on the exit plane, if the test results could be represented by a three-dimensional response surface over such physical two-dimensional domain, it would provide a realistic visualization and a particularly intuitive realization of the flow separation.

For this example, we take the probe distance as ξ_1 (starting from $\xi_1 = 0.70$) and the rake distance as ξ_2 (starting from $\xi_2 = -5$) and convert them to coded variables x_1 and x_2 just as described in Example 4. We read the parametric pressure-profile curves (Figure 11 in [KamhaKT03]) manually for the 21 pressure responses (subject to possible visual round-off errors) and "reproduced" the curves shown in Figure 1. (See Table 6(a) for the read response data.)

TABLE 6(a)
Pressures.

Probe	ξ_1	ξ_2	y
1	0.7	-5	13.0
2	2.075	-5	9.2
3	3.45	-5	5.7*
4	4.825	-5	4.2
5	6.2	-5	3.8
6	7.575	-5	3.7
7	8.95	-5	1.8
8	0.7	0	13.4
9	2.075	0	14.5
10	3.45	0	10.9
11	4.825	0	4.2
12	6.2	0	1.8
13	7.575	0	1.1
14	8.95	0	1.0
15	0.7	5	13.4
16	2.075	5	10.1*
17	3.45	5	6.5
18	4.825	5	5.3
19	6.2	5	4.3*
20	7.575	5	4.3
21	8.95	5	1.5

*Set aside for confirmation test

TABLE 6(b)
Modeling summary—Example 6.

Order	#Coef	Rank	Full?	RMSE%	StDev%
1	3	3	yes	22.930659	5.920671
2	6	6	yes	19.750396	5.701448
3	10	9	No	16.047028	5.349009
4	15	12	No	13.652458	5.573593
5	21	15	No	4.535100	2.618341

Before fitting the responses, we set aside three of them so as to have a token confirmation test on the predictive capability of the resulting model later. There are only 21 responses, and we cannot afford not to use as many as there are, because one of the parametric curves is highly nonlinear, indicating very clearly a strong need for a fairly high order if the responses are to be fitted with reasonably small residuals. We certainly would not ignore any of the responses along the center rake by setting aside for the confirmation purpose. We choose to set aside one point on the port rake ($\xi_1 = 3.45,\ \xi_2 = -5$) and two on the starboard rake ($\xi_1 = 2.075$ and $\xi_1 = 6.2$, both with $\xi_2 = 5$).

(a) Full-order models. We apply Algorithm 1 and model the pressure responses just as we did in Example 4, except that the response function is now truly unknown. Table 6(b) summarizes the results with the fitness measures. The computed singular values are exactly the same as in Example 4 (see Table 4(b)), but it is worth repeatedly mentioning here, because this example is a real test case, that the matrix X stops having full column rank at the third order.

As expected, the residuals (RMSE%) decrease as the order increases. Although zero was not reached at the fifth order, a sharp drop by more than 3 times from the fourth order was attained. It is a rather clear indication that fifth order is most desirable, even though 4.54% RMS error is still large.

TABLE 6(c)
Computed coefficients—Example 6.

Order	1	x_1	x_2	x_1^2	x_1x_2	x_2^2	x_1^3	$x_1^2x_2$	$x_1x_2^2$	x_2^3	
1	6.549223	-6.176863	0.168111								
2	5.688723	-6.167537	0.157090	2.275373	-0.171269	-0.386194					
3	5.686959	-8.807759	0.071011	2.279343	-0.131571	-0.416629	1.254363	-0.030581	2.537936	0.071011	
4	5.029693	-9.156863	0.123257	9.001322	-0.420243	-0.356290	1.703212	-0.012769	2.351510	0.123257	
5	5.186855	-13.933827	0.135362	7.685090	-0.019777	-0.388437	8.458755	0.109929	5.363036	0.135362	
	x_1^4	$x_1^3x_2$	$x_1^2x_2^2$	$x_1x_2^3$	x_2^4	x_1^5	$x_1^4x_2$	$x_1^3x_2^2$	$x_1^2x_2^3$	$x_1x_2^4$	x_2^5
4	-6.741168	0.622441	0.803712	-0.420243	-0.356290						
5	-5.503517	-0.150153	0.741562	-0.019777	-0.388437	-0.681754	-0.593747	-10.365834	0.109929	5.363036	0.135362

On the other hand, the estimated uncertainty spread (StDev%) stops decreasing momentarily at fourth order. Thus, it appears that third order might be a reasonable stop, if one is particularly concerned with increase in model complexity. Nevertheless, the uncertainty spread at the fifth order has a big drop from the fourth (and even from the third) order down to more than 2 times smaller. It means that there are definitely some significant, characteristic fifth- (or higher-) order terms that deserve attention if one does not mind the additional model complexity or the extra effort in seeking such particular fifth-order terms.

In light of Example 4, we can readily recognize that if fifth order were already higher enough for fitting the responses, both fitness measures should have dropped to zero at fifth order. Besides, judging by the still very large size of the residuals, 4.54% RMS, especially compared to the 2.62% estimated uncertainty spread, we hardly can regard this fifth-order model as overfitting (or mistaking noises as) the pressure responses.

Table 6(c) lists the computed coefficients for each of the five orders. Compare with Table 4(c) and observe that the coefficients for each order have essentially no difference between these two examples (only some differ at the sixth digit below the decimal point). The observant reader may have recognized that we use the fifth-order coefficients for (15) in Example 4 to investigate the rank-deficient response modeling and validate the SVD technique. The conclusion of Example 4 also has effectively ascertained the validity of the regression coefficients computed here for fitting the 18 pressure responses with a fifth-order model.

(b) Trimming. Full-order regression has the convenience of not having to guess first what the significant terms to include are, but if we also want to use as few terms as possible, generally we need to trim the polynomial afterward. If the matrix X is rank deficient or nearly rank deficient, trimming first to a full-rank state is desirable. [illegible]. Consider specifically the fifth-order model computed above. The polynomial has 21 coefficients, but the 18x21 matrix X is only of rank 15. We apply Algorithm 2A to trim it down to 15 coefficients while at the same time trying to minimize possible increase in the residuals due to trimming. First, define diagonal matrix D by the 21 computed coefficients. Next, use the 21x21 orthogonal matrix V in the SVD of the matrix X and apply QR factorization with column pivoting to compute $Q^T[V_{11}^T, V_{21}^T]DP = [R_{11}, R_{12}]$ for a permutation matrix P. Pick the first 15 columns of X according to the resulting permutation matrix P to form 18x15 submatrix Z_1. Then refit the 18 responses by repeating Algorithm 1 with Z_1 as if it were a new, 18x15 matrix X. (The refitting effectively adjusts the retained coefficients that correspond to the 15 chosen columns.) The results are shown in Tables 7(a) (summary, measures) and 7(b) (coefficients), all with #Coef "15." Observe that six out of originally 21 coefficients are zeroed but both RMSE% and StDev%

TABLE 7(a)
Refitting—Example 6 *(trimmed* 5*th).*

Order	#Coef	Rank	Full?	RMSE%	StDev%
5	15	15	yes	4.535100	2.618341
5	12	12	yes	4.684133	1.912289

TABLE 7(b)
Computed coefficients—Example 6 *(trimmed* 5*th).*

#Coef	1	x_1	x_2	x_1^2	x_1x_2	x_2^2	x_1^3	$x_1^2x_2$	$x_1x_2^2$	x_2^3	
15	5.186855	-13.933827	0	7.685090	-0.039555	-0.77687	8.458755	0	10.726072	0.40609	
12	5.204313	-14.016793	0	7.538879	0	-0.787260	8.807879	0	10.798500	0.41792	
	x_1^4	$x_1^3x_2$	$x_1^2x_2^2$	$x_1x_2^3$	x_2^4	x_1^5	$x_1^4x_2$	$x_1^3x_2^2$	$x_1^2x_2^3$	$x_1x_2^4$	x_2^5
15	-5.503517	-0.150153	0.741562	0	0	-0.68175	-0.593747	-10.365834	0.219858	0	0
12	-5.366035	0	0.738064	0	0	-0.95108	-0.386094	-10.433512	0	0	0

remain unchanged.

[illegible] We may trim the polynomial further depending on the utility of each term. We apply Algorithm 2B. First, compute and rank the utility. Then trim off three terms of utility less than, say, 1.5% and refit the 18 responses with 12 retained terms . The results also are shown in Tables 7(a) and 7(b) but with #Coef "12." Observe that eight out of 21 coefficients are now zeroed but RMSE% increases just a little and StDev% continues to decrease.

This trimmed fifth-order model has 2 fourth-order and 3 fifth-order terms and yet has a total of 12 terms, only 2 more than the total number of terms in a full third-order model.

(c) Confirmation. Response data not involved in fitting an approximating polynomial should not be expected to be predicted by the very polynomial with a given accuracy. Nevertheless, common intention of regression is to predict new responses by interpolating with the fitted approximating polynomial. Recall that we set aside responses at three test points to be used for confirmation. First, we predict these responses by using each of the four full-order models, second through fifth orders. The results are given in Table 8(a), where y^P stands for the predicted response and Dif% represents the RMS of the three prediction differences (Dif) relative to the RMS of the three responses and is expressed in percentage. (It is defined exactly in the same way as RMSE% in the fitting, except that only the three reserved responses are involved.) It is interesting to observe that prediction uncertainty (Dif%) exhibits the same trend as StDev% in the fitting (see Table 6(b)) and that the fifth-order model predicts the best even though the relative prediction "error" 3.46% is not as small as commonly acceptable. It is also interesting to notice that of all but the fourth-order models the prediction accuracy (Dif%) is better than the fitting accuracy (RMSE%).

Next, we predict the three responses by using the trimmed versions of the fifth-order model. Recall that we trimmed the fifth-order model first to a full-rank state (the 15-term version) by applying Algorithm 2A and then trimmed off three low-utility terms of the 15-term version by Algorithm 2B. The prediction results by these two trimmed versions are similarly given in Table 8(b). The prediction accuracy remains unchanged, just as the fitting accuracy, in the case of the 15-term version but improves slightly, unlike the fitting accuracy (see Table 7(a)), in the case of the 12-term version.

TABLE 8(a)
Prediction results—Example 6.

			Second order		Third order		Fourth order		Fifth order	
ξ_1	ξ_2	y	y^P	Dif	y^P	Dif	y^P	Dif	y^P	Dif
2.075	5	10.1	10.697	-0.597	10.308	-0.208	11.992	-1.892	10.319	-0.219
6.2	5	4.3	3.600	0.701	3.575	0.725	3.106	1.194	4.540	-0.240
3.45	-5	5.7	7.397	-1.697	7.385	-1.685	7.027	-1.327	5.979	-0.279
			Dif%:	15.607		14.923		21.030		3.462

TABLE 8(b)
Prediction results—Example 6 *(trimmed* 5*th).*

			15 terms		12 terms	
ξ_1	ξ_2	y	y^P	Dif	y^P	Dif
2.075	5	10.1	10.319	-0.219	10.130	-0.030
6.2	5	4.3	4.540	-0.240	4.547	-0.247
3.45	-5	5.7	5.979	-0.279	5.994	-0.294
			Dif%:	3.462		3.114

(d) Three-dimensional graphic representation. Having computed the coefficients for various model orders, we are ready to plot the pressure response surfaces. Although response surfaces in current engineering applications commonly mean those of the second-order models, the second-order response surface for the test results, such as shown in Figure 2, clearly is inadequate compared to Figure 1. It is inadequate not because it is an elementary graphic made simply by connecting the adjacent fitted responses with straight lines but because it is simply too flat to capture even an obvious characteristic of the centerline pressure distribution. (In Figure 2 and the subsequent response surface plots, imagine that we are standing inside the rocket on the combustor floor just like the rakes but with our back against the rakes, looking outward just as the flow presses out, and feeling the outward pressure of the air-fuel-exhaust flow on our back.) It is understandable because the second-order model has very large residuals (nearly 20% RMS), even though they were already minimized. Connecting the actual pressure responses likewise by straight lines will depict the pressure profiles like a shattered mirror (see Figure 3), but even such a simplistic graphic looks more inspiring, more revealing, or at least more expressive of the responses than Figure 2. (The second-order response surface remains equally plain, flat, and unexpressive even after including many interpolated responses between the test points in the connection.)

The third-order plot, Figure 4, is slightly better, but the fifth-order version, Figure 5, is clearly much better. First, residuals of the fifth order are more reasonable in size (4.54% versus 16.05% in RMS). Second, the centerline and the two edges of the response surface in Figure 5 resemble the three pressure profiles very well—capturing the basic characteristics shown in Figure 3, the simplistic three-dimensional graphic representation of the pressure responses.

After viewing Figure 5, one might wonder what the pressure distribution would look like if more columns of such probes were placed between the existent rakes. We can duly imagine it by interpolating the response since we have a reasonably well-fitted approximating polynomial in closed form. Suppose there were a 13x11 array of such

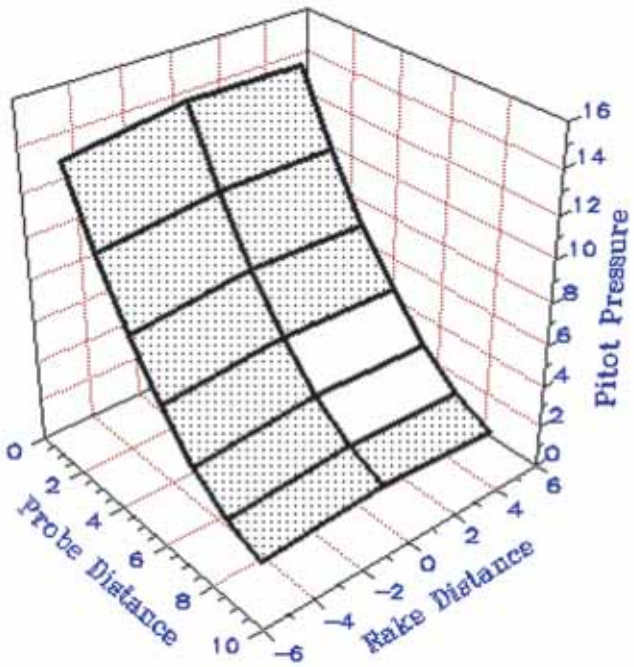


FIG. 2. *Second-order fitted responses connected with lines.*

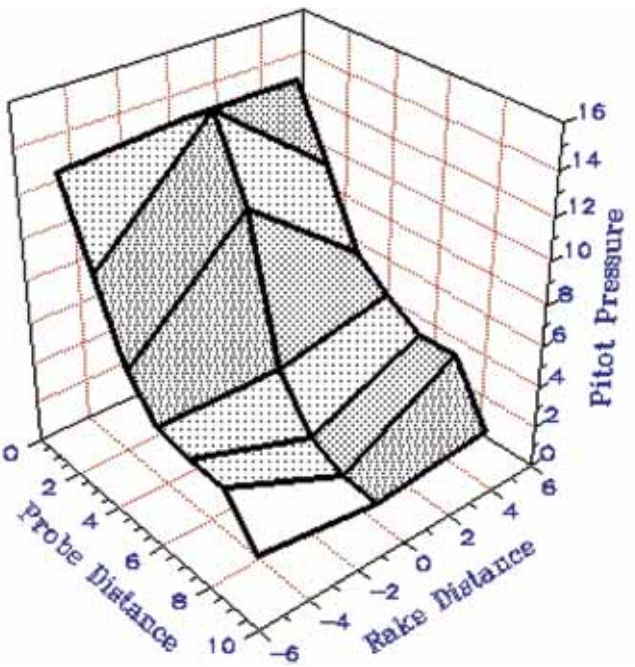


FIG. 3. *Pressure responses connected with lines.*

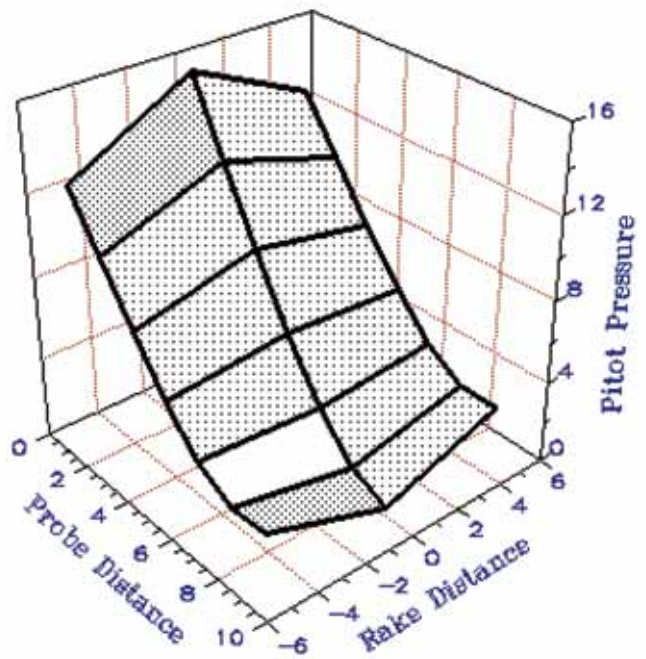


FIG. 4. *Third-order fitted responses connected with lines.*

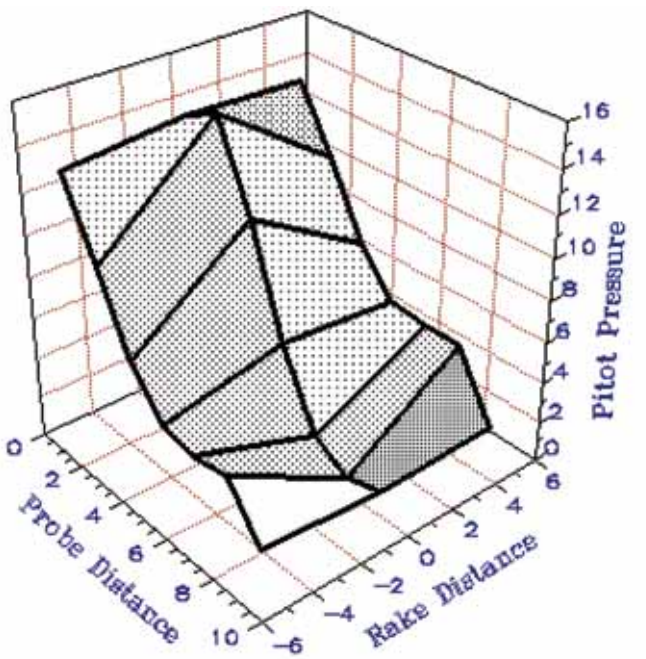


FIG. 5. *Fifth-order fitted responses connected with lines.*

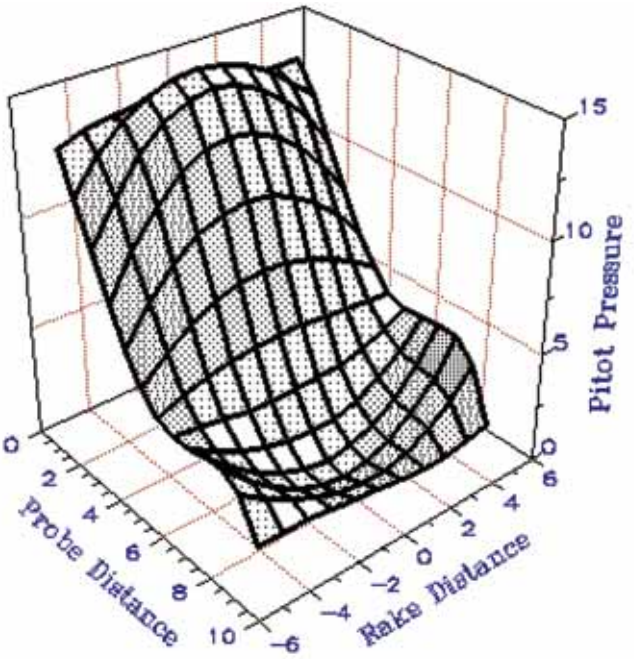


FIG. 6. *Fifth-order* 13x11 *interpolated responses connected with lines.*

probes evenly placed along and between the rakes. The swelling in Figure 6, big near the bottom center and relatively small on the upper port and starboard edges, may be a surprise to many, since conventional response surfaces would never show anything other than extremely flattened or forcibly smoothed. These small lifts on the edges suggest rather clearly that the flow does not mix uniformly but rather separates and some migrates to the outboard rakes. These lifts, in the semiaxisymmetric path of the entrained air stream, might help visualizing (and realizing) that a portion of the air (secondary) stream on the sides remains essentially separated from the rocket (primary) stream in the lower center. Without an expert's well-trained eyes, the flow separation and migration are not really easy to visualize or realize from just looking at the overlaid parameterized plane curves of the pressure profiles. After viewing the illuminating revelations by this response surface plot, Figure 6, it may become much easier to interpret or comprehend the parameterized pressure curves in Figure 1 as indicating flow separation and outward migration.

By the way, the 13x11 third-order response surface plot (Figure 7) is hardly adequate, let alone the corresponding second-order plot (not shown).

8. Conclusion. Instead of being unable to build a higher-order response model for the test results on hand because the product matrix X^TX is not invertible, we can now model the responses to a desirable higher order, despite rank deficiency in X, by using the proposed SVD technique, namely, Algorithm 1. RMSE% and StDev% provide new, intuitively meaningful, universal measures of fitness, whether or not matrix X has full rank. Algorithms 2A and 2B offer two good mathematical alternatives to statistics-based truncation of the fitted polynomials.

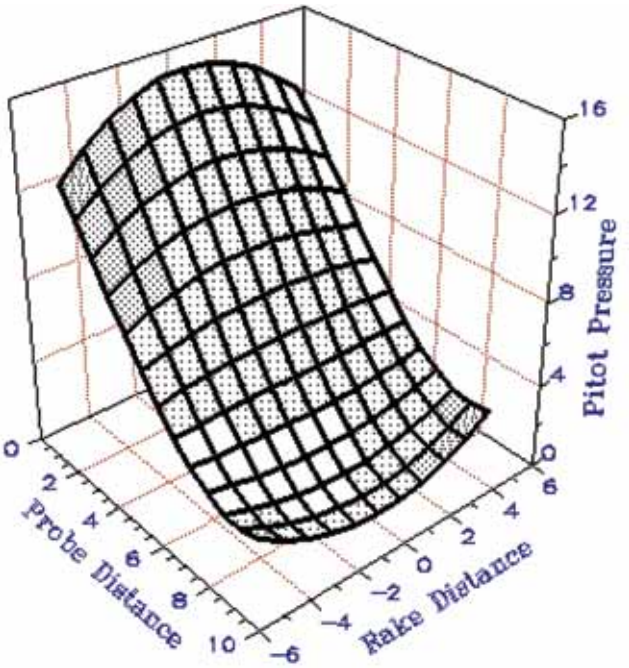


FIG. 7. *Third-order* 13x11 *interpolated responses connected with lines.*

9. Appendix. SVD matrices of a rank-deficient test matrix. A reviewer suggested that an interested reader (e.g., a discipline specialist seeking to apply the proposed SVD approach to physical test data) would be well served by seeing the full matrices (i.e., the U, S, and V) of the SVD result for the third-order model in Example 4. For computing LS solutions, all right singular vectors may be used (depending on the number of singular values accepted as being nonzero), but at most $\min\{N, p\}$ left singular vectors are useful. Since $N = 18$ and $p = 10$ for this model, only 10 columns of U (i.e., 10 left singular vectors) are listed below. To conserve space further, all numbers are rounded to the third decimal place.

10 left singular vectors in 18x18 Matrix U:

1	2	3	4	5	6	7	8	9	10
−0.346	−0.247	0.428	−0.433	−0.011	0.252	0.312	−0.092	0.253	−0.071
−0.311	−0.208	0.238	−0.292	−0.156	−0.022	0.079	0.209	−0.335	0.150
−0.291	−0.174	0.010	−0.006	−0.269	−0.223	−0.274	0.283	−0.123	−0.376
−0.309	−0.173	−0.085	0.128	−0.230	−0.153	−0.295	0.134	0.162	0.337
−0.347	−0.178	−0.208	0.247	−0.121	0.028	−0.145	−0.085	0.223	0.126
−0.408	−0.187	−0.386	0.346	0.062	0.317	0.225	−0.337	−0.196	−0.167
−0.173	0.046	0.321	0.028	0.418	0.117	−0.363	−0.420	0.115	−0.138
−0.145	0.045	0.170	0.018	0.276	−0.176	−0.162	−0.215	−0.353	0.240
−0.128	0.045	0.080	0.014	0.188	−0.352	−0.061	−0.090	−0.306	0.297
−0.125	0.047	0.023	0.011	0.159	−0.410	−0.011	−0.004	0.000	−0.498
−0.137	0.056	−0.031	0.003	0.194	−0.354	0.040	0.082	0.306	0.074
−0.165	0.073	−0.112	−0.017	0.295	−0.186	0.141	0.207	0.356	−0.216
−0.211	0.102	−0.247	−0.053	0.468	0.093	0.342	0.410	−0.107	0.240
−0.161	0.376	0.458	0.497	−0.001	0.331	−0.115	0.395	0.077	0.020
−0.139	0.328	0.150	0.192	−0.247	−0.159	0.361	−0.097	−0.240	−0.113
−0.141	0.329	0.059	0.048	−0.286	−0.226	0.318	−0.257	0.068	0.011
−0.177	0.400	−0.139	−0.252	−0.180	−0.013	−0.078	−0.208	0.330	0.300
−0.215	0.474	−0.305	−0.420	−0.026	0.263	−0.333	0.080	−0.231	−0.218

10 right singular vectors in 10x10 Matrix V:

1	2	3	4	5	6	7	8	9	10
−0.701	0.233	0.099	0.025	0.292	−0.599	−0.014	−0.004	0.000	0.000
−0.066	0.072	−0.659	−0.032	0.012	−0.005	0.166	0.289	0.666	0.000
0.210	0.622	0.053	0.031	−0.008	−0.001	0.193	−0.164	0.030	0.707
−0.375	0.132	0.060	−0.055	0.519	0.752	0.000	−0.002	0.002	0.000
0.011	0.047	0.054	−0.994	−0.044	−0.049	−0.042	−0.022	0.022	0.000
−0.508	0.150	0.051	0.024	−0.800	0.271	0.042	0.020	−0.018	0.000
−0.041	0.067	−0.561	−0.063	0.034	−0.013	0.292	0.215	−0.737	0.000
0.109	0.342	0.015	0.034	−0.020	0.011	−0.706	0.602	−0.094	0.000
−0.056	0.056	−0.475	0.014	−0.023	0.017	−0.557	−0.674	−0.050	0.000
0.210	0.622	0.053	0.031	−0.008	−0.001	0.193	−0.164	0.030	−0.707

Except the diagonal, zero is everywhere in the 18x10 "diagonal" S. On the diagonal $(S_{ii}, i = 1, \ldots, 10)$ are 10 computed singular values, the same as listed in the row of Table 4(b) for order "3."

Acknowledgment. The author wishes to thank the referees for many constructive comments and valuable suggestions.

REFERENCES

[BatesGW03] R. A. Bates, B. Giglio, and H. P. Wynn, *A global selection procedure for polynomial interpolators*, Technometrics, 45 (2003), pp. 246–255.

[BoxD71] M. J. Box and N. R. Draper, *Factorial designs, the |X'X| criterion, and some related matters*, Technometrics, 13 (1971), pp. 731–742.

[Chan82] T. F. Chan, *An improved algorithm for computing the singular value decomposition*, ACM Trans. Math. Software, 8 (1982), pp. 72–83.

[ChenAT96] W. Chen, J. K. Allen, K.-L. Tsui, and F. Mistree, *A procedure for robust design*, ASME J. Mech. Design, 118 (1996), pp. 478–485.

[ChenGM01] W. Chen, R. Garimella, and N. Michelena, *Robust design for improved vehicle handling under a range of maneuver conditions*, Engineering Optimization, 33 (2001), pp. 303–326.

[GiuntBH97] A. A. Giunta, V. Balabanov, D. Haim, B. Grossman, W. H. Mason, L. T. Watson, and R. T. Haftka, *Multidisciplinary optimisation of a supersonic transport using design of experiments theory and response surface modelling*, Aeronautical J., 101 (1997), pp. 347–356.

[Golub69] G. H. Golub, *Matrix decompositions and statistical calculations*, in Statistical Computation, R. C. Milton and J. A. Nelder, eds., Academic Press, New York, 1969, pp. 365–397.

[GolubK69] G. Golub and W. Kahan, *Calculating the singular values and pseudo-inverse of a matrix*, SIAM J. Numer. Anal., 2 (1965), pp. 205–224.

[GolubR70] G. H. Golub and C. Reinsch, *Singular value decomposition and least squares solutions*, Numer. Math., 14 (1970), pp. 403–420. Also in Handbook for Automatic Computation, Vol. 2, Linear Algebra, J. H. Wilkinson and C. Reinsch, eds., Springer-Verlag, New York, 1971.

[GolubV83] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Johns Hopkins University Press, Baltimore, MD, 1983.

[Grevi59] T. N. E. Greville, *The pseudoinverse of a rectangular or singular matrix and its application to the solution of systems of linear equations*, SIAM Rev., 1 (1959), pp. 38–43.

[HinkeK94] K. Hinkelman and O. Kempthorne, *Design and Analysis of Experiments*, John Wiley & Sons, New York, 1994.

[KamhaKT03] H. Kamhawi, T. M. Krivanek, S. R. Thomas, and J. F. Walker, *Direct-Connect Ejector Ramjet Combustor Experiment*, AIAA Paper 2003-0016, 41st Aerospace Sciences Meeting and Exhibit, Reno, NV, 2003.

[KlemaL80] V. C. Klema and A. J. Laub, *The singular value decomposition: Its computation and some applications*, IEEE Trans. Automat. Contr., 25 (1980), pp. 164–176.

[KornK68] G. A. Korn and T. M. Korn, *Mathematical Handbook for Scientists and Engineers*, 2nd ed., McGraw–Hill, New York, 1968.

[KhuriC96] A. I. Khuric and J. A. Cornell, *Response Surfaces: Designs and Analysis*, 2nd ed., Marcel Dekker, New York, 1996.

[LawsoH74] C. L. Lawson and R. J. Hanson, *Solving Least Squares Problems*, Prentice–Hall, Englewood Cliffs, NJ, 1974.

[Mitch74] T. J. Mitchell, *An algorithm for the construction of D-optimal experimental designs*, Technometrics, 16 (1974), pp. 203–210.

[Myers71] R. H. Myers, *Response Surface Methodology*, Allyn and Bacon, Boston, MA, 1971.

[MyersM95] R. H. Myers and D. C. Montgomery, *Response Surface Methodology: Process and Product Optimization Using Designed Experiments*, John Wiley, New York, 1995.

[Renka88] R. J. Renka, *Multivariate interpolation of large sets of scattered data*, ACM Trans. Math. Software, 14 (1988), pp. 139–148.

[Renka99] R. J. Renka, *Algorithm* 790: *CSHEP2D*: *Cubic Shepard method for bivariate interpolation of scattered data*, ACM Trans. Math. Software, 25 (1999), pp. 70–73.

[Rohde66] C. A. Rohde, *Some results on generalized inverses*, SIAM Rev., 8 (1966), pp. 201–205.

[Searl71] S. R. Searle, *Linear Models*, John Wiley, New York, 1971.

[SteffSY00] C. J. Steffen, Jr., T. D. Smith, S. Yungster, and D. J. Keller, *Computational analysis for rocket-based combined-cycle systems during all-rocket mode*, AIAA J. Propulsion & Power, 16 (2000), pp. 1030–1039.

[Stewa73] G. W. Stewart, *Introduction to Matrix Computations*, Academic Press, New York, 1973.

[Trefn99] C. Trefny, *An Air-Breathing Launch Vehicle Concept for Single-Stage-to-Orbit*, AIAA Paper 99-2730, 35th AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit, 1999.

[Wilki68] J. H. Wilkinson, *A priori error analysis of algebraic processes*, Proc. Intl. Cong. Math., Mir, Moscow, 1968, pp. 629–639.

SIAM J. SCI. COMPUT.
Vol. 28, No. 3, pp. 868–885
© 2006 Society for Industrial and Applied Mathematics

ON THE PRESERVATION OF INVARIANTS BY EXPLICIT RUNGE–KUTTA METHODS*

M. CALVO†, D. HERNÁNDEZ-ABREU‡, J. I. MONTIJANO†, AND L. RÁNDEZ†

Abstract. A new strategy to preserve invariants in the numerical integration of initial value problems with explicit Runge–Kutta methods is presented. It is proved that this technique retains the order of the original method, has an easy and cheap implementation, and can be used in adaptive Runge–Kutta codes. Some numerical experiments with the classical code of Dormand and Prince, DoPri5(4), based on a pair of embedded methods with orders 5 and 4, are presented to show the behavior of the new method for several problems which possess invariants.

Key words. initial value problems, explicit Runge–Kutta methods, numerical geometric integration, preservation of invariants, variable step-size codes

AMS subject classifications. 65L05, 65L06

DOI. 10.1137/04061979X

1. Introduction and basic definitions.

First integrals and invariants of (autonomous) differential systems

$$y' = f(y), \tag{1}$$

where $f : \mathcal{D} \subset \mathbb{R}^m \to \mathbb{R}^m$ is a sufficiently smooth function, play an important role both in qualitative and quantitative studies of the flow of (1). These invariants allow us to describe the geometry of the orbits and also to check the accuracy of numerical integrators of the corresponding differential equation.

A function $G = G(y) : \widehat{\mathcal{D}} \subset \mathbb{R}^m \to \mathbb{R}^l$ of class $\mathcal{C}^1(\widehat{\mathcal{D}})$, $\widehat{\mathcal{D}} \subset \mathcal{D}$, is called an l-invariant system of (1) in $\widehat{\mathcal{D}}$ (see [15, p. 61]) if

(i) there exists some $y^0 \in \widehat{\mathcal{D}}$ such that $G(y^0) = 0$,
(ii) for all solutions $y = y(t)$ of (1), $G(y(t)) = 0$ holds either for every t or for no t in the interval of definition of $y(t)$.

In such a case we may consider the flow defined by (1) restricted to the $(m-l)$-dimensional manifold $\mathcal{M} = \{y \in \widehat{\mathcal{D}}; G(y) = 0\}$ as a differential equation on $\mathcal{M}$. In the case $l = 1$, the invariant system is called an invariant relation. In connection with the above concept, it is worth remarking that for $l > 1$ the l scalar relations which constitute the invariant system need not be invariant relations. An l-invariant system is also called a weak invariant (see [5]), and it is not difficult to see that $G(y)$ is a weak invariant of (1) if and only if there exists some $y^0 \in \widehat{\mathcal{D}}$ such that $G(y^0) = 0$, and $\nabla G(y)^T f(y) = 0$ for all $y \in \mathcal{M}$.

On the other hand, a scalar function $F : \widehat{\mathcal{D}} \subset \mathbb{R}^m \to \mathbb{R}$ of class $C^1(\widehat{\mathcal{D}}), \widehat{\mathcal{D}} \subset \mathcal{D}$, is a first integral (or a conserved quantity) of (1) if $F(y(t))$ is a constant along any

*Received by the editors November 29, 2004; accepted for publication (in revised form) December 27, 2005; published electronically June 9, 2006.
http://www.siam.org/journals/sisc/28-3/61979.html

†Departamento Matemática Aplicada, Universidad de Zaragoza, 50009-Zaragoza, Spain (calvo@unizar.es, monti@unizar.es, randez@unizar.es). The work of these authors was supported by project MTM2004-06466-C02-01.

‡Departamento de Análisis Matemático, Universidad de La Laguna, 38271, La Laguna, Spain (dhabreu@ull.es). The work of this author was supported by project MTM2004-06466-C02-02 and grant AP2002-2761.

solution path $y = y(t)$ of (1). In such a case, since

$$0 = \frac{dF(y(t))}{dt} = \nabla F(y(t))^T \cdot f(y(t)) \qquad \text{for all } t,$$

it follows that $\nabla F(y)^T \cdot f(y) = 0$ for all $y \in \widehat{\mathcal{D}}$. Now, for all $y^0 \in \widehat{\mathcal{D}}$ the solution $y = y(t)$ of (1) such that $y(0) = y^0$ is contained in the hypersurface $\mathcal{M}_{y^0} = \{y \in \mathbb{R}^m \,;\, F(y) = F(y^0)\}$. First integrals are also called strong invariants (see [5, 7]) and quadratic integrals $G(y) = y^T S y$, where $S \in \mathbb{R}^{m\times m}$ is a constant symmetric matrix, are strong invariants of (1) if and only if $y^T\, S\, f(y) = 0$ for all $y \in \mathcal{D}$.

In order to simplify the presentation, we will use in the following the term invariant of the differential system (1) for either a first integral or an invariant system (or relation) of (1).

Since many general numerical integrators do not preserve first integrals or invariants of their corresponding differential systems, these functions have been widely used as a tool to measure the accuracy and the long-term behavior of these integrators. As a matter of fact, in the last two decades there has been an increasing interest in numerical integrators that preserve as far as possible the qualitative properties of the underlying differential systems. In particular, a natural requirement is to consider those numerical Runge–Kutta (RK) methods that preserve first integrals or invariants. It was proved by Cooper [3] that all RK methods conserve linear invariants. However, an irreducible RK method preserves all quadratic invariants $G(y) = y^T S y + \mu^T y + \nu$ with constants $S \in \mathbb{R}^{m\times m}, \mu \in \mathbb{R}^m, \nu \in \mathbb{R}$, if and only if their coefficients ($A \in \mathbb{R}^{s\times s}, b \in \mathbb{R}^s$) satisfy $m_{ij} \equiv b_i a_{ij} + b_j a_{ji} - b_i b_j = 0$, $1 \le i \le j \le s$. These conditions, which also imply that (A, b) is a symplectic method, pose strong requirements on the coefficients of the method which are satisfied only by some implicit RK methods and have been extensively studied in connection with symplectic methods. Moreover, it can be seen (see, e.g., [5, Thm. 3.3, p. 102]) that for $n \ge 3$ no RK method can conserve all polynomial invariants of degree n. Further studies on the preservation of invariants by numerical methods have been carried out in [7, 9, 11].

An alternative attempt to preserve invariants in numerical integrations is proposed by Schropp [10]. This author modifies the original flow of (1) by stabilizing the invariant set $\mathcal{M}_{y^0} = \{y \in \mathbb{R}^m \,;\, G(y) = G(y^0)\}$ and considers the modified problem

$$y' = f(y) - \nabla G(y)\; a(y)(G(y) - G(y^0)),$$

where $a(y)$ is a suitably chosen smooth function so that $\mathcal{M}_{y^0}$ is an exponentially attracting set for the modified problem. In view of this, the numerical solutions generated by the method are expected to approach this attracting set; however, the modified problem may become very stiff depending on the choice of $a(y)$.

In view of the above limitations several techniques have been proposed to preserve invariants in practical numerical integrators. Thus, del Buono and Mastroserio in [1], starting with an explicit s-stage RK method ($A = (a_{ij}), b = (b_i)$) given by the equations

$$\varphi_h(y_n) = y_n + h \sum_{j=1}^{s} b_j f(Y_{n,j}), \tag{2}$$

with

$$Y_{n,1} = y_n, \quad Y_{n,i} = y_n + h\sum_{j=1}^{i-1} a_{ij} f(Y_{n,j}) \quad (2 \le i \le s), \tag{3}$$

modify the advancing solution (2) by introducing at each step a scalar $\gamma_n = \gamma_n(y_n, h)$, so that the new method $\widehat{\varphi}_h$ defined by (3) and

$$\widehat{\varphi}_h(y_n) = y_n + h\gamma_n \sum_{j=1}^{s} b_j f(Y_{n,j}) \tag{4}$$

preserves a given invariant of (1). Clearly, this requirement implies that γ_n will be a problem-dependent scalar such that $\gamma_n \to 1$ when $h \to 0$ and may vary from step to step. These authors [1] have proved that for all differential systems (1) possessing a quadratic invariant $G(y) = y^T S y$ with a symmetric S, there exist fourth-order explicit RK methods φ_h with four stages such that the modified methods $\widehat{\varphi}_h$ given by (3) and (4) preserve the quadratic invariant and have order three for a suitable value γ_n. Further, $\widehat{\varphi}_h$ attains order four at $t_n + h\gamma_n$, and therefore $\widehat{\varphi}_h$ would get (global) order four at the nonuniform grid $\{\widehat{t}_n\}_{n=0}^N$ with $\widehat{t}_0 = t_0$ and $\widehat{t}_{n+1} = \widehat{t}_n + h\gamma_n$. To compute γ_n at each step, the authors [1] use the formula $\gamma_n = 1 - (\delta_n/\eta_n)$ with

$$\eta_n = \sum_{i,j=1}^{4} b_i b_j \langle f(Y_{n,i}), f(Y_{n,j})\rangle, \quad \delta_n = \eta_n - 2\sum_{i=2}^{4}\sum_{j=1}^{i-1} b_i a_{ij} \langle f(Y_{n,i}), f(Y_{n,j})\rangle,$$

where $\langle u, v\rangle \equiv u^T S v$ for $u, v \in \mathbb{R}^m$ is the bilinear form associated to the invariant $G(y) = y^T S y$. This means that the computational cost of $\widehat{\varphi}_h$ at each step includes $s(s-1)/2 = 6$ bilinear forms $\langle f(Y_{n,i}), f(Y_{n,j})\rangle$, apart from the function evaluations of (3). Note that $\widehat{\varphi}_h(y_n) = \varphi_h(y_n) + (\gamma_n - 1)(\varphi_h(y_n) - y_n)$ can be considered as the projection of $\varphi_h(y_n)$ onto the invariant manifold $\mathcal{M}_{y_n} = \{y \ ; \ G(y) = G(y_n)\}$ along the direction $\varphi_h(y_n) - y_n$. In the remainder of this paper this technique will be called the incremental direction technique.

An alternative technique (see [5, p. 106]), known as the standard projection method, consists of combining a standard RK method φ_h (or any one-step method) together with an orthogonal projection P onto the invariant manifold $\mathcal{M}_{y_n}$ at each step $(t_n, y_n) \to (t_{n+1} = t_n + h, y_{n+1})$. For the Euclidean norm, it would be given by $y_{n+1} = (P \cdot \varphi_h)(y_n)$, with y_{n+1} defined by $y_{n+1} = \varphi_h(y_n) + \nabla G(y_{n+1})\ \lambda_n$, with $G(y_{n+1}) = G(y_n)$. Due to the implicitness, this system is replaced in [5] by

$$y_{n+1} = \varphi_h(y_n) + \nabla G(\varphi_h(y_n))\ \lambda_n \quad \text{with} \quad G(y_{n+1}) = G(y_n), \tag{5}$$

and the calculation of λ_n at each step is carried out by applying a simplified Newton iteration to the corresponding nonlinear equation. As remarked in [5], since λ_n is small, taking the starting value $\lambda_{n,0} = 0$, one simplified Newton iteration is usually enough for practical purposes. It can be proved that (5) preserves the order of the original method $\varphi_h(y_n)$ but requires us to provide an analytical expression of the gradient $\nabla G(y)$ that will be evaluated at each step and also will be used to obtain λ_n by a simplified Newton iterative scheme in the solution of the implicit equations $G(y_{n+1}) = G(y_n)$. On the other hand, although RK methods preserve linear invariants and are affine invariant, the standard projection method (5) in general does

not preserve these properties, and this fact may be inconvenient for the qualitative behavior of the integrator.

As follows from (4) and (5), the above projection techniques can be included in the general form

$$\widehat{\varphi}_h(y_n) = \varphi_h(y_n) - \lambda_n \Phi_n, \tag{6}$$

where the vector $\Phi_n \in \mathbb{R}^m$ defines the direction of projection and λ_n is a scalar chosen so that $\widehat{\varphi}_h(y_n)$ belongs to the invariant manifold. In the case (4) the direction $\Phi_n = \varphi_h(y_n) - y_n$ is defined by the original method and a zero-order method, whereas in the case (5) of the simplified projection, Φ_n is orthogonal to the invariant at $\varphi_h(y_n)$. Clearly, other possible directions Φ_n may be considered. In particular, taking into account that for many explicit RK methods several low order embedded approximations of type

$$\widetilde{\varphi}_h(y_n) = y_n + h \sum_{j=1}^{s} \widetilde{b}_j f(Y_{n,j}) \tag{7}$$

are usually available without additional cost, one is tempted to take $\Phi_n = \varphi_h(y_n) - \widetilde{\varphi}_h(y_n)$, and then the resulting projected method (6)

$$\widehat{\varphi}_h(y_n) = y_n + h \sum_{j=1}^{s} ((1-\lambda_n) b_j + \lambda_n \widetilde{b}_j) f(Y_{n,j}) \tag{8}$$

has the form of an RK method with (variable) weights $\widehat{b}_j = (1-\lambda_n) b_j + \lambda_n \widetilde{b}_j$ and retains important properties of these methods, such as the preservation of linear invariants, the affine invariance, and a low computational cost.

The aim of this paper is to study projected explicit RK methods of type (8) that preserve general invariants with the purpose of being used in numerical integrators based on adaptive RK methods. In section 2, a revision of the case $\widetilde{\varphi}_h(y_n) = y_n$, in which the companion method has order zero, is presented with the purpose of generalizing the results given in [1] and also of showing a more efficient way of computing λ_n. In section 3, a general study for quadratic invariants and embedded methods $\widetilde{\varphi}_h$ with order q $(1 \le q < p)$ is carried out. In section 4 this study is generalized to nonquadratic scalar invariants and to the case of several scalar invariants. Further, some remarks on how to include the proposed schemes in adaptive RK codes are also given. Finally, in section 5, some numerical experiments with the classical embedded pair of explicit RK methods of orders five and four, DoPri5(4) [4], are shown in order to make clear how our technique performs on several test problems.

2. The incremental direction technique revisited. In [1] del Buono and Mastroserio propose to advance the integration $t_n \to t_{n+1} = t_n + h$ of (1) with the RK method $\widehat{\varphi}_h$ given by (4), where γ_n is properly chosen so that the method preserves a fixed quadratic invariant $G(y) = y^T S\, y = \langle y, y \rangle$. Here we will write $\widehat{\varphi}_h$ in the equivalent form

$$\widehat{\varphi}_h(y_n) = \varphi_h(y_n) - \lambda_n(\varphi_h(y_n) - y_n) = (1-\lambda_n)\varphi_h(y_n) + \lambda_n y_n, \tag{9}$$

so that the scalar $\lambda_n = 1 - \gamma_n$ will be a small quantity at each step.

Next it will be seen that for all RK methods φ_h with order $p \ge 2$ the projected method $\widehat{\varphi}_h$ has order $p - 1$. This result generalizes Proposition 4 of [1] in two

directions: First, it extends the statement on fourth-order RK methods to any order $p \geq 2$; second, there are no restrictions on the coefficients of the method φ_h in contrast with the situation in [1]. Finally, we will show that for any order p, λ_n (or γ_n) can be computed with two bilinear form evaluations $\langle \cdot, \cdot \rangle$ per step.

With the notation (9) and putting $\varphi_h(y_n) - y_n = h\Delta_n$, the preservation of the quadratic invariant $G(y) = y^T S y = \langle y, y \rangle$ holds if and only if λ_n satisfies

$$2\langle y_n, h\Delta_n \rangle + (1 - \lambda_n)\langle h\Delta_n, h\Delta_n \rangle = 0. \tag{10}$$

If $\langle y_n, h\Delta_n \rangle = \langle h\Delta_n, h\Delta_n \rangle = 0$, then $\langle \widehat{\varphi}_h(y_n), \widehat{\varphi}_h(y_n) \rangle = \langle y_n, y_n \rangle$ for all λ_n, and therefore it would be enough to take $\lambda_n = 0$ (i.e., the original method φ_h) to preserve the invariant. If either $\langle y_n, h\Delta_n \rangle = 0$ or else $\langle h\Delta_n, h\Delta_n \rangle = 0$, equation (10) cannot be satisfied by $\lambda_n \neq 1$ (note that $\lambda_n = 1$ leads to the method $\widehat{\varphi}_h(y_n) = y_n$ producing a constant output) and therefore it is not possible to preserve the invariant. Hence the condition

$$\langle y_n, \Delta_n \rangle \, \langle \Delta_n, \Delta_n \rangle \neq 0 \tag{11}$$

must be satisfied in order to make (10) solvable. Observe that if φ_h is Euler's method, since $\Delta_n = f(y_n)$, condition (11) does not hold. Thus, only methods with order $p \geq 2$ will be considered.

THEOREM 2.1. [illegible] φ_h [illegible] $p \geq 2$ [illegible] (11) [illegible]

(i) [illegible] $\widehat{\varphi}_h$ [illegible] (9) [illegible] $G(y) = \langle y, y \rangle$ [illegible] λ_n [illegible]

$$\lambda_n = \frac{\langle 2y_n + h\Delta_n, h\Delta_n \rangle}{\langle h\Delta_n, h\Delta_n \rangle} = \frac{\langle \varphi_h(y_n), \varphi_h(y_n) \rangle - \langle y_n, y_n \rangle}{\langle \varphi_h(y_n) - y_n, \varphi_h(y_n) - y_n \rangle}. \tag{12}$$

(ii) [illegible] $\langle f(y_n), f(y_n) \rangle \neq 0$, [illegible] $\widehat{\varphi}_h$ [illegible] $p - 1$ [illegible]. Under the assumption (11), equation (10) has the solution λ_n given by (12), and therefore the method $\widehat{\varphi}_h$ preserves the quadratic invariant. On the other hand, since φ_h has order p, we have

$$\varphi_h(y_n) = y_n + h\Delta_n = y(t_n + h) + \epsilon_{n+1},$$

where $\epsilon_{n+1} = C_{p+1}h^{p+1} + \mathcal{O}(h^{p+2})$ is the local error of φ_h at y_n with step size h and $y(t_n + t)$ is the local solution of (1) at (t_n, y_n); i.e., it satisfies $y(t_n) = y_n$ and

$$\langle y(t_n + h), y(t_n + h) \rangle = \langle y(t_n), y(t_n) \rangle = \langle y_n, y_n \rangle$$

for all h. Then, for the numerator and denominator of λ_n in (12) we have

$$\begin{aligned} \langle \varphi_h(y_n), \varphi_h(y_n) \rangle - \langle y_n, y_n \rangle &= 2\langle y_n, C_{p+1} \rangle h^{p+1} + \mathcal{O}(h^{p+2}), \\ \langle h\Delta_n, h\Delta_n \rangle &= h^2 \langle y'(t_n), y'(t_n) \rangle + \mathcal{O}(h^3), \end{aligned}$$

and λ_n behaves like $O(h^{p-1})$ providing that $\langle y'(t_n), y'(t_n) \rangle = \langle f(y_n), f(y_n) \rangle \neq 0$, and, according to (9), this implies that $\widehat{\varphi}_h$ is a method of order $p - 1$. □

To end this section, observe that it follows from (12) that the computation of λ_n requires only two bilinear forms per step in contrast with [1] which needs six bilinear forms for $s = 4$. Moreover, since the two required bilinear forms are $\langle \varphi_h(y_n), \varphi_h(y_n) \rangle, \langle \varphi_h(y_n), y_n \rangle$, only one matrix-vector product is necessary.

3. A general projection technique for quadratic invariants. Let φ_h be an explicit s-stage RK method (A, b) with order p defined by equations (2)–(3), and let $\widetilde{\varphi}_h$ be an embedded method (7) with order q $(1 \le q < p)$ defined by the weights $\widetilde{b}_i,\ i = 1, \ldots, s$. As remarked above, practical RK methods possess embedded methods with several orders $(1 \le q \le p-1)$, so that we have wide freedom in the choice of $\widetilde{\varphi}_h$ and the computation of the embedded method does not require additional function evaluations. Suppose that (1) possesses a quadratic invariant $G(y) = y^T S y = \langle y, y\rangle$. We propose to complete the step $t_n \to t_{n+1}$, projecting $\varphi_h(y_n)$ onto the manifold defined by the invariant $\mathcal{M}_{y_n} = \{y \in \mathbb{R}^m; \langle y, y\rangle = \langle y_n, y_n\rangle\}$ along the direction given by the unit vector

$$w_n = w(y_n, h) = \frac{\varphi_h(y_n) - \widetilde{\varphi}_h(y_n)}{\|\varphi_h(y_n) - \widetilde{\varphi}_h(y_n)\|_2}, \tag{13}$$

where $\|\cdot\|_2$ is the Euclidean norm.

Then the projected method $\widehat{\varphi}_h$ is given by

$$\widehat{\varphi}_h(y_n) = \varphi_h(y_n) - \lambda_n w_n, \tag{14}$$

with the scalar $\lambda_n = \lambda(y_n, h)$ such that

$$G(\widehat{\varphi}_h(y_n)) - G(y_n) \equiv \langle \widehat{\varphi}_h(y_n), \widehat{\varphi}_h(y_n)\rangle - \langle y_n, y_n\rangle = 0. \tag{15}$$

Substituting (14) into (15), λ_n must satisfy

$$\alpha_n \lambda_n^2 - 2\beta_n \lambda_n + \delta_n = 0, \tag{16}$$

with

$$\begin{aligned} \alpha_n &= \alpha(y_n, h) = \langle w_n, w_n\rangle, \\ \beta_n &= \beta(y_n, h) = \langle \varphi_h(y_n), w_n\rangle, \\ \delta_n &= \delta(y_n, h) = \langle \varphi_h(y_n), \varphi_h(y_n)\rangle - \langle y_n, y_n\rangle. \end{aligned} \tag{17}$$

Given $h > 0$ and y_n, if $\delta(y_n, h) = 0$, the original method $\varphi_h(y_n)$ preserves the quadratic invariant in this step and in such a case we would take $\lambda_n = 0$.

If $\delta_n \neq 0$, let us suppose $\beta_n^2 - \alpha_n \delta_n > 0$; then we take for λ_n the real root of (16) closest to $\lambda = 0$ given by

$$\lambda_n = \begin{cases} \dfrac{\delta_n}{\beta_n + \operatorname{sign}\beta_n \sqrt{\beta_n^2 - \alpha_n \delta_n}} & \text{if} \quad \beta_n \neq 0, \\ (-\delta_n/\alpha_n)^{1/2} & \text{if} \quad \beta_n = 0. \end{cases} \tag{18}$$

To study the order of the projected method (14), (15), observe that denoting by $C_{q+1}(y_n)h^{q+1}$ the leading term of the local error of the embedded method $\widetilde{\varphi}_h(y_n)$, $w_n = C_{q+1}/\|C_{q+1}\|_2 + \mathcal{O}(h)$ and then

$$\beta(y_n, 0) = \langle \varphi_h(y_n), w(y_n, h)\rangle|_{h=0} = \left\langle y_n, \frac{C_{q+1}}{\|C_{q+1}\|_2} \right\rangle. \tag{19}$$

Now we have the following theorem.

THEOREM 3.1. [illegible] φ_h [illegible] $\widetilde{\varphi}_h$ [illegible] RK [illegible] p [illegible] q $(1 \le q \le p-1)$.

(i) [illegible] $\langle y_n, C_{q+1}(y_n)\rangle \neq 0$ [illegible] $h^* > 0$ [illegible] $\widehat{\varphi}_h$ [illegible] (14) [illegible] $\lambda_n = \lambda(y_n, h)$ [illegible] (18) [illegible] $\widehat{p} \ge p$ [illegible] $h \in (0, h^*]$

(ii) [illegible] β_n [illegible] (17) [illegible] $\beta_n = B(y_n)h^r + \mathcal{O}(h^{r+1})$ [illegible] $B(y_n) \neq 0$ $r \ge 1$ [illegible] $p - 2r + 1 > 0$, [illegible] $\widehat{\varphi}_h$ [illegible] $\widehat{p} \ge p - r$ [illegible] $h \in (0, h^*]$

[illegible]. (i) Since $\beta(y_n, 0) \neq 0$ and by (17) we have $\alpha_n = \mathcal{O}(1)$, $\delta_n = \mathcal{O}(h^{p+1})$, there exist some $h^* > 0$ such that $\beta_n^2 - \alpha_n\delta_n > 0$ for $h \in (0, h^*]$. Then λ_n is well defined by (18) and $\lambda_n = \mathcal{O}(h^{p+1})$, which implies that the projected method (14) has order $\widehat{p} \ge p$.

(ii) Now, since $\beta_n = \mathcal{O}(h^r)$, $\alpha_n = \mathcal{O}(1)$, $\delta_n = \mathcal{O}(h^{p+1})$, and $2r < p+1$, $\beta_n^2 - \alpha_n\delta_n > 0$ also holds for $h \in (0, h^*]$. Further it follows from (18) that $\lambda_n = \mathcal{O}(h^{p+1-r})$ and now the order $\widehat{p}$ of the projected method is $\ge p - r$. ∎

In practical application of (14), given an explicit RK method φ_h of order p, we usually have the freedom to choose several embedded methods $\widetilde{\varphi}_h$ with different orders $(1 \le q \le p-1)$. Then the question is how to choose $\widetilde{\varphi}_h$. It will be seen that the simplest choice, namely Euler's method, in general provides a projected method with order $\ge p$. In fact, for $\widetilde{\varphi}_h(y_n) = y_n + hf(y_n)$, we have $q = 1$ and $C_2 = y_n''(t_n)/2 = f'(f(y_n))/2$, and the condition

$$\beta(y_n, 0) = \left\langle y_n, \frac{f'(f(y_n))}{\|f'(f(y_n))\|_2} \right\rangle \neq 0$$

implies that the projected method has order $\ge p$.

Finally, observe that $\langle y_n(t+h), y_n(t+h)\rangle - \langle y_n, y_n\rangle = 0$ along the local solution $y_n(t)$ at (t_n, y_n) for all h sufficiently small implies $\langle y_n, f'(f(y_n))\rangle = -\langle f(y_n), f(y_n)\rangle$ and the above condition can be easily checked.

[illegible]. The above theory extends easily to more general invariants which include both quadratic and linear terms with the form $G(y) = y^T S y + d^T y$, where S is a symmetric constant matrix and d a constant vector. In fact, for $\widehat{\varphi}_h$ given by (14), (15) holds if and only if λ_n satisfies

$$\alpha_n\lambda_n^2 - 2\beta_n'\lambda_n + \delta_n' = 0, \tag{20}$$

with $\beta_n' = \beta_n + (1/2)d^T w_n$ and $\delta_n' = \delta_n + d^T\varphi_h(y_n) - d^T y_n$. If $\delta_n' = 0$, for a given h, the original method φ_h preserves the invariant and we take $\lambda_n = 0$. Otherwise, the existence of a real solution of the quadratic equation (20) depends on the sign of $(\beta_n')^2 - \alpha_n\delta_n'$, and sufficient conditions similar to those in Theorem 3.1 can be given in order to guarantee the solvability.

4. General invariants. In this section we extend our technique to nonquadratic scalar invariants and to the case of several scalar invariants.

For a general scalar invariant given by a smooth function $G = G(y) : \widehat{\mathcal{D}} \subset \mathbb{R}^m \to \mathbb{R}$ the proposed projection technique reduces to the solution of the nonlinear scalar equation

$$G\Big(\varphi_h(y_n) - \lambda_n w_n\Big) - G(y_n) = 0, \tag{21}$$

where $w_n = w_n(y_n, h)$ is given by (13). In this case, the existence of solution λ_n is guaranteed under the conditions of the following theorem.

THEOREM 4.1. φ_h $\widetilde{\varphi}_h$ p q $(1 \le q \le p-1)$, $C_{q+1}(y_n)h^{q+1}$ $\widetilde{\varphi}_h(y_n)$

(i) $\nabla G(y_n)^T C_{q+1}(y_n) \neq 0$ $h^* > 0$ (21) $\lambda_n = \lambda(y_n, h)$ $h \in [0, h^*]$ $\widehat{\varphi}(y_n)$ $\hat{p} \ge p$

(ii) $\nabla G(\varphi_h(y_n))^T w_n = B(y_n)h^r + \mathcal{O}(h^{r+1})$ $r \ge 1$ $B(y_n) \neq 0$ $2r \le p$ $h^* > 0$ (21) $\lambda_n = \lambda(y_n, h)$ $h \in (0, h^*]$ $\widehat{\varphi}(y_n)$ $\hat{p} \ge p - r$

Proof. (i) For λ, h in a neighborhood of the origin let us define the real function $g(\lambda, h) \equiv G(\varphi_h(y_n) - \lambda w_n) - G(y_n)$. Taking into account the smoothness of G and φ_h and that

$$g(0,0) = G(y_n) - G(y_n) = 0, \quad \frac{\partial g}{\partial \lambda}(0,0) = -\nabla G(y_n)^T \frac{C_{q+1}(y_n)}{\|C_{q+1}(y_n)\|_2} \neq 0,$$

the implicit function theorem ensures the existence of a neighborhood $[0, h^*]$ and a unique smooth function $\lambda_n = \lambda_n(h)$ satisfying $\lambda_n(0) = 0$ and $g(\lambda_n(h), h) = 0$ for all $h \in [0, h^*]$.

Moreover, we can expand

$$g(\lambda_n, h) = g(0, h) + \frac{\partial g}{\partial \lambda_n}(0, h)\lambda_n + \mathcal{O}(\lambda_n^2),$$

with

$$g(0,h) = G(\varphi_h(y_n)) - G(y_n) = \mathcal{O}(h^{p+1}),$$
$$\frac{\partial g}{\partial \lambda_n}(0,h) = \frac{\partial g}{\partial \lambda_n}(0,0) + \mathcal{O}(h).$$

Then $\lambda_n = \lambda_n(h) = \mathcal{O}(h^{p+1})$ and the projected method has order $\hat{p} \ge p$.

(ii) Now we consider the real function $z(\mu, h)$ defined in the neighborhood of the origin by

$$z(\mu, h) = h^{-2r}\left[G(\varphi_h(y_n) - \mu h^r w_n) - G(y_n)\right] \quad \text{for } h \neq 0, \tag{22}$$

where $z(\mu, 0)$ is given by the corresponding limit

$$\lim_{h \to 0} z(\mu, h) = -B(y_n)\mu + \frac{1}{2\|C_{q+1}(y_n)\|_2^2} C_{q+1}(y_n)^T \frac{\partial^2 G}{\partial y^2}(y_n) C_{q+1}(y_n)\mu^2. \tag{23}$$

Clearly z is continuous for all $h > 0$. To check the continuity of z at $h = 0$ note that by the mean value theorem,

$$G(\varphi_h(y_n) - \mu h^r w_n) = G(\varphi_h(y_n)) - \mu h^r \int_0^1 \nabla G(\varphi_h(y_n) - \theta \mu h^r w_n)^T w_n \mathrm{d}\theta;$$

then for $h \neq 0$ we have

$$z(\mu, h) = -\mu h^{-r} \int_0^1 \nabla G(\varphi_h(y_n) - \theta \mu h^r w_n)^T w_n \mathrm{d}\theta + h^{-2r}\big(G(\varphi_h(y_n)) - G(y_n)\big).$$

Let us consider the smooth function defined as $\varsigma(y) := \nabla G(y)^T w_n$. By considering again the mean value theorem applied to ς, we can write

$$\varsigma(\varphi_h(y_n) - \theta\mu h^r w_n) = \varsigma(\varphi_h(y_n)) - \theta\mu h^r \int_0^1 \nabla\varsigma(\varphi_h(y_n) - \rho\theta\mu h^r w_n)^T w_n \mathrm{d}\rho.$$

Thus, we have that

$$\begin{aligned} z(\mu,h) = {} & h^{-2r}(G(\varphi_h(y_n)) - G(y_n)) - \mu h^{-r}\varsigma(\varphi_h(y_n)) \\ & + \mu^2 \int_0^1 \int_0^1 \theta\nabla\varsigma(\varphi_h(y_n) - \rho\theta\mu h^r w_n)^T w_n \mathrm{d}\rho\mathrm{d}\theta. \end{aligned} \tag{24}$$

Since $p+1 > 2r$, we deduce (23) from (24), and then $z(\mu,h)$ is a continuous function at $h = 0$.

Moreover, $\partial z/\partial\mu$ is also a continuous function in a neighborhood of the origin, and $\partial z/\partial\mu(0,0) = -B(y_n) \neq 0$; therefore by the implicit function theorem there exist $h^* > 0$ and a continuous $\mu = \mu(h)$ such that $z(\mu(h),h) = 0$ for all $h \in [0,h^*]$.

Taking into account (22), for all $h \in (0,h^*]$, $\lambda_n(h)$ given by $\lambda_n(h) = h^r\mu(h)$ is the unique solution of $G(\varphi_h(y_n) - \lambda_n w_n) - G(y_n) = 0$.

Finally, by considering the expansion

$$z(\mu,h) = z(0,h) + \frac{\partial z}{\partial\mu}(0,h)\mu + \mathcal{O}(\mu^2),$$

with

$$\begin{aligned} z(0,h) &= h^{-2r}\big(G(\varphi_h(y_n)) - G(y_n)\big) = \mathcal{O}(h^{p+1-2r}), \\ \frac{\partial z}{\partial\mu}(0,h) &= \frac{\partial z}{\partial\mu}(0,0) + \mathcal{O}(h) = -B(y_n) + \mathcal{O}(h), \end{aligned}$$

it follows that $\mu = \mathcal{O}(h^{p+1-2r})$ and therefore $\lambda_n = \mathcal{O}(h^{p+1-r})$, which implies that the projected method has order $\hat{p} \geq p - r$. □

1. Since w_n is a unit vector in the direction of the local error of the embedded method $\widetilde{\varphi}_h(y_n)$ and $w_n = C_{q+1}(y_n)/\|C_{q+1}(y_n)\|_2 + \mathcal{O}(h)$, $C_{q+1}(y_n) \neq 0$, condition $\nabla G(y_n)^T C_{q+1}(y_n) \neq 0$ of Theorem 4.1(i) is equivalent to the nonvanishing of the derivative of $G(y)$ in the direction of the leading error term $C_{q+1}(y_n)$ of w_n at y_n. Geometrically it means that $C_{q+1}(y_n)$ is not tangent to the level surface $G(y) - G(y_n) = 0$ at y_n. Further, if the leading term of $\nabla G(\varphi_h(y_n))^T w_n$ given by $\nabla G(y_n)^T C_{q+1}(y_n)$ vanishes for some y_n, but $\nabla G(\varphi_h(y_n))^T w_n = \mathcal{O}(h^r)$ with $1 \leq r \leq p/2$, it is possible to ensure the preservation of the invariant at the cost of an order reduction ($\hat{p} \geq p - r$) of the projected numerical solution.
2. Theorem 4.1 can also be applied to prove that the standard orthogonal projection of an RK method of order p has order $\geq p$. In fact (5) has the form (14) with $w_n = -\nabla G(\varphi_h(y_n))$, and in this case since $\nabla G(y_n)^T w_n = \|\nabla G(y_n)\|_2^2 + \mathcal{O}(h)$, the method (5) has order $\geq p$. If $\nabla G(y_n) = 0$, the standard projection cannot be applied, but in such a case if $\nabla G(\varphi_h(y_n))^T w_n = \mathcal{O}(h^r)$, our projection technique could be applied. Note also that the standard orthogonal projection requires having an analytical expression of $\nabla G(y)$, whereas in our approach the gradient is not necessary.

3. Recall that for all affine transformations $z \to y = Pz + q$ with constant $P \in \mathbb{R}^{m\times m}$, $q \in \mathbb{R}^m$, if we denote by $\psi_{f,t}$ the flow map of $y' = f(y)$ and by $\psi_{\tilde{f},t}$ the corresponding flow map to the transformed system $z' = \tilde{f}(z) = P^{-1}f(Pz+q)$, they are affine invariants in the sense that $\psi_{f,t}(y_0) = P\psi_{\tilde{f},t}(z_0) + q$ whenever $y_0 = Pz_0+q$. It can be seen that this property holds for RK methods and also for the proposed methods (14). However, the standard projection preserves the affine invariance only for orthogonal P.

4.1. Implementation in adaptive RK codes. To use our technique in adaptive codes based on an embedded pair of RK formulas with orders p and $p-1$, recall that the step size $t_n \to t_{n+1} = t_n + h_n$ is chosen so that a norm of the local error estimator $\mathrm{EST} = y_{n+1} - y_{n+1}^{(p-1)}$ given by the difference between the solutions y_{n+1} and $y_{n+1}^{(p-1)}$ of orders p and $p-1$, respectively, is lower than a given error tolerance TOL. For the new projected solution $\widehat{y}_{n+1} = y_{n+1} - \lambda_n w_n$ one is tempted to use the new estimator $\widehat{\mathrm{EST}} = \widehat{y}_{n+1} - y_{n+1}^{(p-1)}$, but $\widehat{y}_{n+1}$ could lose (at some isolated times) one or more orders of accuracy and then this is not a reliable estimate. Thus we propose a more conservative policy in which we check both EST and the correction of projection $\|\lambda_n w_n\| = |\lambda_n|$ and accept a step whenever $\max\{\|\mathrm{EST}\|, |\lambda_n|\} \le \mathrm{TOL}/2$.

Observe that $\|\mathrm{EST}\| = \mathcal{O}(h^p)$ and that if the projected solution has no order reduction, then $|\lambda_n| = \mathcal{O}(h^{p+1})$. Hence the above step-size control will be usually governed by EST.

Then for a rejected step h_n' or the new step h_{n+1} we propose the new step size given by the modified standard formula

$$\mathrm{fac} * \left(\frac{\mathrm{TOL}}{2\max\{\|\mathrm{EST}\|, \lambda_n\}}\right)^{1/p} * h_n,$$

with the security factor fac of the original code.

On the other hand, note that in the projected method $\widehat{y}_{n+1} = \widehat{\varphi}_h(y_n)$ can be rewritten as

$$\widehat{y}_{n+1} = y_{n+1} - \lambda_n w_n = y_{n+1} - \frac{\lambda_n}{\|y_{n+1} - \widetilde{y}_{n+1}\|}\,(y_{n+1} - \widetilde{y}_{n+1})$$

and is an RK type method whenever $\lambda_n \|y_{n+1} - \widetilde{y}_{n+1}\|^{-1}$ is bounded. Since $\lambda_n = \mathcal{O}(h^{p+1-r})$ and $\|y_{n+1} - \widetilde{y}_{n+1}\| = \mathcal{O}(h^{q+1})$, the above quotient is $\mathcal{O}(h^{p-q-r})$; then for a strong order reduction ($r > p-q$) the projected method would lose its reliability. To protect the code from such eventual strong order reductions we propose to include the additional control $|\lambda_n| < \|y_{n+1} - \widetilde{y}_{n+1}\|$. If this condition does not hold at some step, we would take another $\widetilde{\varphi}_h$ method that would change the direction of the projected method. In our numerical experiments we have not detected such a strong order reduction that this control has been necessary.

Finally, in practical application of our technique with one scalar nonquadratic invariant we must solve at each step the (nonquadratic) equation $g(\lambda, h) = 0$. Many iterative methods have been proposed to solve nonlinear equations: In Newton-type methods we need to approximate the gradient of G (either analytically or numerically) one or more times per step depending on the convergence of the iteration. In our numerical experiments with the modified Newton, convergence was attained with 1–2 iterations. However, other iterative schemes that require only evaluations of G may be used. We have found that the secant method, with starting values $\lambda = 0$ and the

value λ obtained in the previous step, requires 1–2 iterations to attain convergence to the round-off level in the tested problems. Thus, we have used this iterative scheme because its convergence is similar to that of the modified Newton and it has a lower computational cost.

4.2. Several invariants. Suppose now that the differential system (1) possesses $l \geq 1$ smooth invariants $G_1(y), \ldots, G_l(y)$ defined in $\mathcal{D} \subset \mathbb{R}^m$, and let $G = (G_1, \ldots, G_l)^T : \mathcal{D} \subset \mathbb{R}^m \to \mathbb{R}^l$. Together with the starting method φ_h of order p we consider l linearly independent embedded methods $\widetilde{\varphi}_h^{(1)}, \ldots, \widetilde{\varphi}_h^{(l)}$ with orders $q_1 \leq q_2 \leq \cdots \leq q_l < p$. Then, generalizing (14), the projected method $\widehat{\varphi}_h$ will be defined by

$$\widehat{\varphi}_h(y_n) = \varphi_h(y_n) - \sum_{i=1}^{l} \lambda_n^{(i)} w_n^{(i)}, \tag{25}$$

where

$$w_n^{(i)} = \frac{\varphi_h(y_n) - \widetilde{\varphi}_h^{(i)}(y_n)}{\|\varphi_h(y_n) - \widetilde{\varphi}_h^{(i)}(y_n)\|}$$

and $\lambda_n^{(i)} = \lambda^{(i)}(y_n, h)$, $i = 1, \ldots, l$, are problem-dependent scalars that will be chosen so that

$$G\left(\varphi_h(y_n) - \sum_{i=1}^{l} \lambda_n^{(i)} w_n^{(i)}\right) = G(y_n). \tag{26}$$

We also introduce the $m \times l$ matrix T_n defined by its column vectors $T_n = [w_n^{(1)} | \ldots | w_n^{(l)}]$, and then

$$(\nabla G(\varphi_h(y_n)))^T \; T_n = M(y_n) + \mathcal{O}(h), \tag{27}$$

with $M(y_n) \in \mathbb{R}^{l \times l}$. With this notation we have the following theorem.

THEOREM 4.2. [illegible] (27) [illegible] $M(y_n)$ [illegible] $h^* > 0$ [illegible] $(\lambda^{(1)}(y_n, h), \ldots, \lambda^{(l)}(y_n, h))$ [illegible] $\widehat{\varphi}_h$ [illegible] (25) [illegible] l [illegible] $G_i, i = 1, \ldots, l$ [illegible] $h \in [0, h^*]$ [illegible] $\geq p$ [illegible]. Consider $\lambda = (\lambda^{(1)}, \ldots, \lambda^{(l)})^T$ and $g(\lambda, h) : \mathbb{R}^l \times \mathbb{R} \to \mathbb{R}$ defined by

$$g(\lambda, h) = G\left(\varphi_h(y_n) - \sum_{i=1}^{l} \lambda^{(i)} w_n^{(i)}\right) - G(y_n). \tag{28}$$

The proof follows in a way similar to that of statement (i) of Theorem 4.1. ∎

To end this section, the following result is given in order to ensure that Newton-type iterations applied to (26) provide numerical sequences which converge in small neighborhoods of the origin. In fact, we have the following corollary.

COROLLARY 4.1. [illegible] 4.2 [illegible] $h^* > 0$ [illegible] $0 < h < h^*$ [illegible] (26), [illegible] $\lambda := 0$ [illegible]

Considering, as in Theorem 4.2 above, the function $g(\lambda, h)$ (28), it is not difficult to see that

$$\gamma := \sup_{\substack{\eta,\mu\in\Lambda_0 \\ \eta\neq\mu}} \frac{\| \nabla_\lambda g(\eta, h) - \nabla_\lambda g(\mu, h) \|}{\| \eta - \mu \|} = \mathcal{O}(1),$$

$$\alpha := \| \nabla_\lambda g(0, h)^{-1} g(0, h) \| = \mathcal{O}(h^{p+1}),$$

$$\beta := \| \nabla_\lambda g(0, h)^{-1} \| = \mathcal{O}(1),$$

where $\Lambda_0 \subset \mathbb{R}^l$ stands for a sufficiently small neighborhood of $\lambda = 0$. Thus, there exists $h^* > 0$ such that for all $0 < h < h^*$, $\tau := \gamma\alpha\beta = \mathcal{O}(h^{p+1}) \leq 1/2$ is fulfilled. Then, the statement follows from the Newton–Kantorovich theorem (see, e.g., Theorem 5.3.6 in [14]). □

Concerning the implementation of the new projection technique with several invariants, it is straightforward to extend the step-size control of subsection 4.1 to several invariants.

5. Numerical experiments. Here we present some numerical experiments with the purpose of showing some features of the new projection technique. First of all we want to make clear that it is possible to include in any adaptive RK code the new projection technique in a simple way. Further, the resulting code retains its reliability and accuracy and has a low additional computational cost. For comparison we have also included the standard orthogonal projection technique which also preserves the order of the original method but requires explicit expressions of the gradients ∇G and in general has a higher computational cost. In our experiments we have employed as adaptive RK code the pair of Dormand and Prince, DoPri5(4) (see, e.g., [4] for details on the coefficients of this method). Some practical examples have been considered in order to illustrate how the projection technique performs on this kind of problem with invariants. For the sake of brevity, we will show here some results related to the following three problems. Further numerical examples can be found in [2].

1. A problem of micromagnetism [12].

$$f(y) := H_{eff} \times y + \lambda y \times (H_{eff} \times y),$$

with $\lambda = 1/20.1$ and $H_{eff}, y \in \mathbb{R}^3$. We have chosen the initial value $y(0) = (y_0[1], y_0[2], y_0[3]) = (\sin\theta_0 \cos\varphi_0, -\sin\theta_0 \sin\varphi_0, \cos\theta_0)^T$, where $\varphi_0 = \pi/4$ and $\theta_0 = \pi/3$.

This kind of equation appears in micromagnetism problems when solving the Landau–Lifshitz–Gilbert equation (see, e.g., [8, 12, 13]). Since $y'(t)$ is orthogonal to the exact solution $y(t)$, it is clear that $G(y) = y^T y$ is a quadratic invariant. In case of $H_{eff} := (1, 0, 0)^T$, the exact solution is given by

$$y(t) = \left(\frac{a(t)}{b(t)}, \frac{2}{b(t)}(y_0[2]\cos t - y_0[3]\sin t), \frac{2}{b(t)}(y_0[2]\sin t + y_0[3]\cos t) \right)^T,$$

with

$$a(t) = e^{\lambda t}(1 + y_0[1]) - e^{-\lambda t}(1 - y_0[1]),$$
$$b(t) = e^{\lambda t}(1 + y_0[1]) + e^{-\lambda t}(1 - y_0[1]).$$

2. The restricted three body problem (see, e.g., [6, pp. 129–130]). Here $F : \mathbb{R}^4 \to \mathbb{R}^4$ is given by

$$f(y) := \left(y_3, y_4, y_1 + 2y_4 - \bar{\mu}\frac{y_1+\mu}{D_1^{3/2}} - \mu\frac{y_1-\bar{\mu}}{D_2^{3/2}}, y_2 - 2y_3 - \bar{\mu}\frac{y_2}{D_1^{3/2}} - \mu\frac{y_2}{D_2^{3/2}} \right)^T,$$

with $D_1 := (y_1+\mu)^2 + y_2^2$, $D_2 := (y_1-\bar{\mu})^2 + y_2^2$, $y = (y_1, y_2, y_3, y_4)^T$, and $y(0) = (0.994, 0, 0, -2.00158510637908252240537862224)^T$, where $\bar{\mu} = 1-\mu$ and $\mu = 0.012277471$. For this initial condition we get a periodic solution with a minimal period $T_2 := 17.06521656015796255889172062 49$. Now the total energy

$$G(y) = \frac{1}{2}(y_3^2 + y_4^2 - y_1^2 - y_2^2) - \bar{\mu}D_1^{-1/2} - \mu D_2^{-1/2}$$

is a nonquadratic first integral.

3. Euler equations (see, e.g., [1] and [5, pp. 95–96]).

$$f(y) := \big((\alpha-\beta)y_2y_3,\ (1-\alpha)y_3y_1,\ (\beta-1)y_1y_2\big)^T,$$

with $y = (y_1, y_2, y_3)^T$ and $y(0) = (0, 1, 1)^T$, where $\alpha = 1 + \frac{1}{\sqrt{1.51}}$ and $\beta = 1 - \frac{0.51}{\sqrt{1.51}}$. In these equations the exact solution can be written as

$$y(t) = \left(\sqrt{1.51}\ \mathrm{sn}(t, 0.51), \mathrm{cn}(t, 0.51), \mathrm{dn}(t, 0.51) \right)^T,$$

where sn, cn, dn stand for the elliptic Jacobi functions. This problem has two quadratic integrals:

$$\text{(i)}\ \ G_1(y) = y_1^2 + y_2^2 + y_3^2, \qquad \text{(ii)}\ \ G_2(y) = y_1^2 + \beta y_2^2 + \alpha y_3^2.$$

These problems have been integrated with the following three methods: DoPri5(4) (the classical pair of Dormand and Prince [4]), DoPri5(4)+SP (the pair of Dormand and Prince with the standard projection technique), and DoPri5(4)+DP (the same pair with the proposed directional projection technique). To define the directional projection we have used the Euler method as embedded method $\widetilde{y}_{n+1}$ (of order $q = 1$) when only one scalar invariant must be preserved (Problems 1 and 2). For Problem 3, which posses two invariants, we need two "independent" embedded methods of orders q_1 and q_2 to define the projected method. We have used the Euler method ($q_1 = 1$) and a second-order embedded method ($q_2 = 2$).

For a quadratic invariant $G(y) = y^T Sy$, since the gradient $\nabla G = 2Sy$ is immediately available, we compute λ_n from the quadratic equation (12). For nonquadratic invariants λ_n is computed by a modified Newton iteration in the case of orthogonal projection and by a secant iterative scheme for our projected solution. For several invariants we have used modified Newton for both projection techniques (in the case of our projected solution Broyden's iteration is under consideration).

The first problem has been integrated with several fixed step sizes in the interval $[0, T_{end} = 16\pi]$. The global and invariant errors (i.e., *ge* and *ie*, resp.) for DoPri5(4) and DoPri5(4)+DP and the different step sizes have been plotted in Figure 1. It can be observed that the projected method preserves the invariant and also provides a numerical solution with the same order (the slopes of the global error curves are the same) but with a smaller global error than classical Dopri5(4).

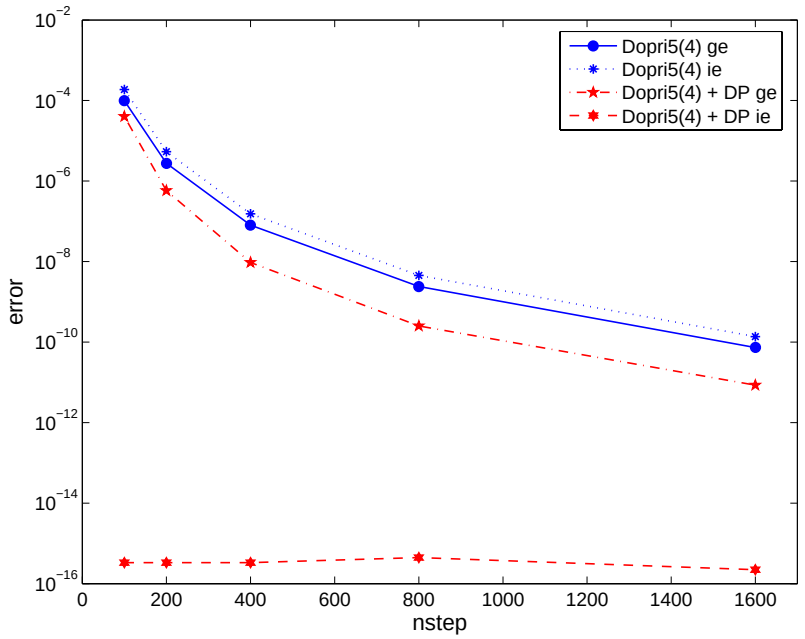


FIG. 1. *Problem* 1 *integrated with DoPri*5(4) *and DoPri*5(4)+*DP.*

On the other hand, Problems 2 and 3 have been integrated with variable step size using the technique proposed in subsection 4.1, with values $ATOL = 10^{-6}$ and $RTOL = ATOL/10$ representing the absolute and relative tolerance errors, respectively. The fourth-order embedded method was considered in order to estimate the local error at each point in the nonuniform grid. The end point was taken in each problem as $T_{end} = 3T_2$ and $T_{end} = 100$, respectively.

Figures 2 and 3 show the numerical orbits in the (y_1, y_2)-plane when Problem 2 is integrated with DoPri5(4) and DoPri5(4)+DP, respectively (the figure for DoPri5(4)+SP is very similar to the one corresponding to DoPri5(4)+DP). It is seen that both projected methods give a better qualitative behavior than DoPri5(4) because they preserve the corresponding first integral, as shown in Figure 4. It must be remarked that some round-off errors came up when solving the nonlinear equations defining the projected methods in neighborhoods of the singularities of the underlying problem, and this is the reason the projected methods do not preserve properly the invariant in neighborhoods of the singularities. Efficiency curves have not been presented for this problem because they show a similar behavior to those in Problems 1 and 3.

Figures 5 and 6 correspond to Problem 3, integrated with variable step size. In this case, we have displayed in Figure 5 the global error produced by the three methods in relation to the number of function evaluations, whereas Figure 6 plots the error in both invariants. Again, the projected methods provide numerical solutions with a smaller global error than the classical pair 5(4) for a similar number of derivative function evaluations.

6. Conclusions. A projection technique to preserve invariants by means of explicit RK methods has been proposed. The projected methods preserve both the order

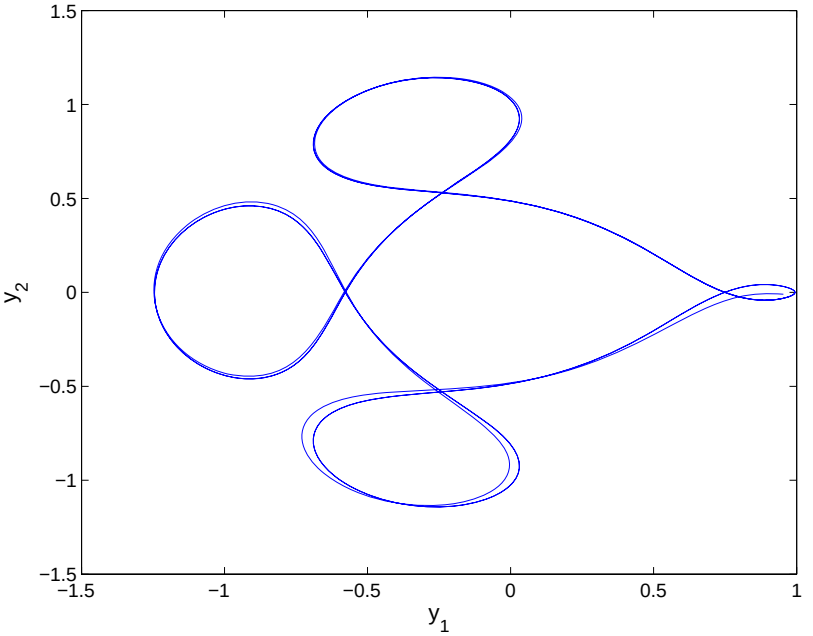


FIG. 2. *Problem* 2 *integrated with DoPri*5(4).

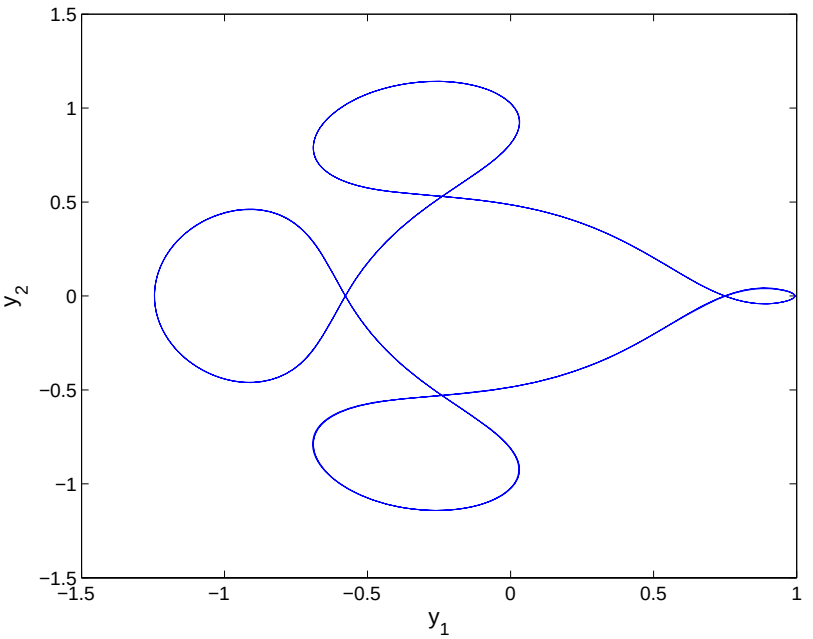


FIG. 3. *Problem* 2 *integrated with DoPri*5(4)+*SP or DoPri*5(4)+*DP*.

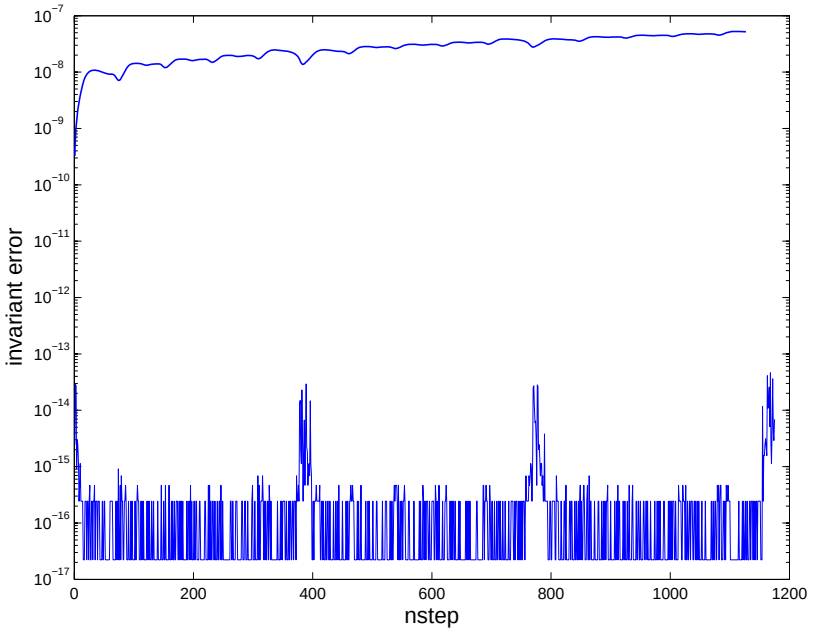


FIG. 4. *Problem* 2. *Invariant errors with DoPri*5(4) *and DoPri*5(4)+*DP.*

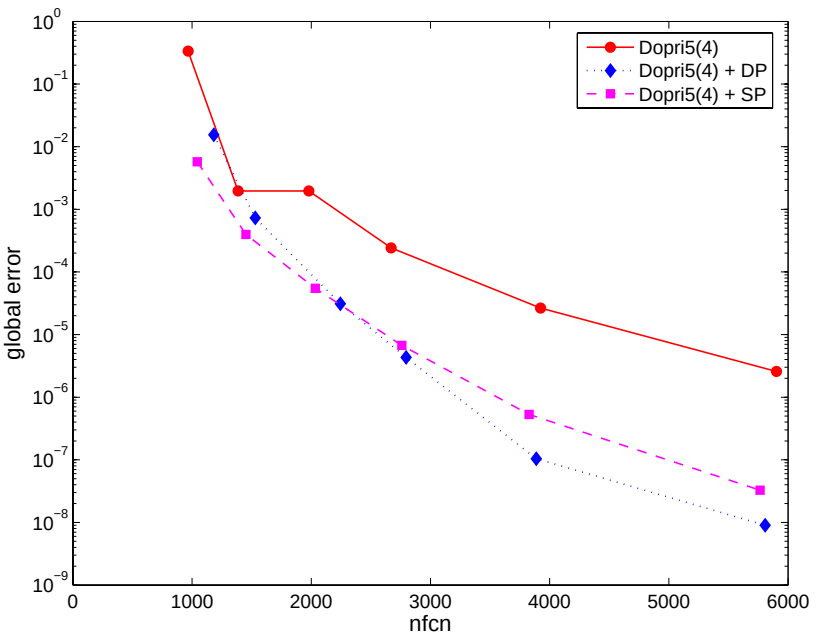


FIG. 5. *Problem* 3. *Global errors with DoPri*5(4), *DoPri*5(4)+*SP, and DoPri*5(4)+*DP.*

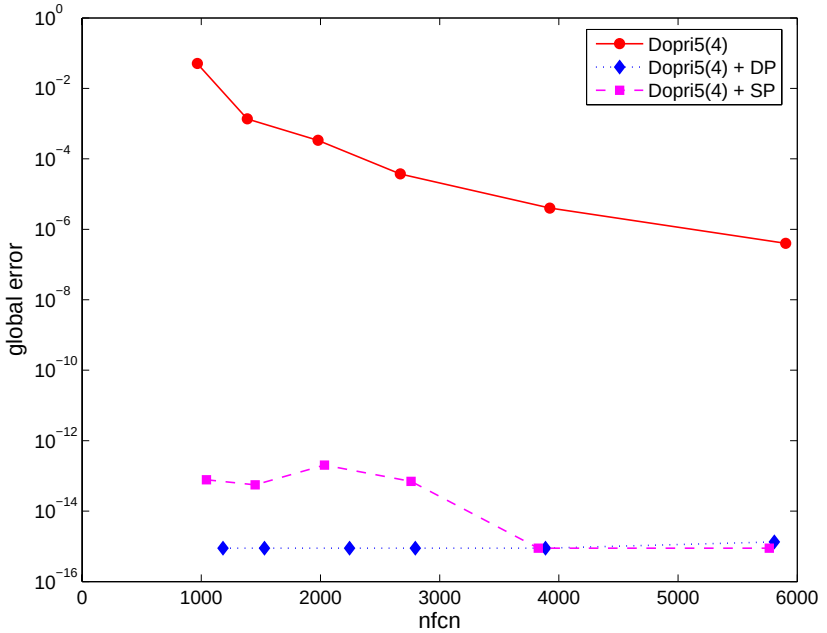


FIG. 6. *Problem* 4. *Invariant errors with DoPri*5(4), *DoPri*5(4)+*SP, and DoPri*5(4)+*DP.*

of consistency of the original methods and the selected invariants of the problem under consideration. A study of the order of consistency has been carried out.

The main advantages of the new approach over the standard projection technique are as follows: The gradient of the invariants is not required. The approach preserves linear invariants and affine invariance. Further, the new projection methods are well suited for use in adaptive RK methods with a low extra cost.

Finally the new technique and the standard projection have been implemented in the code DoPri5(4), and the results of some numerical experiments are presented.

REFERENCES

[1] N. DEL BUONO AND C. MASTROSERIO, *Explicit methods based on a class of four stage fourth order Runge–Kutta methods for preserving quadratic laws*, J. Comput. Appl. Math., 140 (2002), pp. 231–243.

[2] M. CALVO, D. HERNÁNDEZ-ABREU, J. I. MONTIJANO, AND L. RÁNDEZ, *Explicit Runge-Kutta methods for the preservation of invariants*, Technical report, Departamento Matemática Aplicada, Universidad Zaragoza, Zaragoza, Spain, 2004.

[3] G. J. COOPER, *Stability of Runge–Kutta methods for trajectory problems*, IMA J. Numer. Anal., 7 (1987), pp. 1–13.

[4] J. R. DORMAND AND P. J. PRINCE, *A family of embedded Runge-Kutta formulae*, J. Comp. Appl. Math., 6 (1980), pp. 19-26.

[5] E. HAIRER, C. LUBICH, AND G. WANNER, *Geometric Numerical Integration: Structure Preserving Algorithms for Ordinary Differential Equations*, Springer-Verlag, Berlin, 2002.

[6] E. HAIRER, S. P. NØRSETT, AND G. WANNER, *Solving Ordinary Differential Equations*. I. *Non-stiff Problems*, Springer Ser. Comput. Math. 8, Springer-Verlag, Berlin, 1993.

[7] A. ISERLES AND A. ZANNA, *Preserving algebraic invariants with Runge-Kutta methods*, J. Comput. Appl. Math., 125 (2000), pp. 69–81.

[8] D. LEWIS AND N. NIGAM, *Geometric integration on spheres and some interesting applications*, J. Comput. Appl. Math., 151 (2003), pp. 141–170.

[9] G. R. W. Quispel and D. I. McLaren, *Integral-preserving integrators*, J. Phys. A, 37 (2004), pp. L489–L495.
[10] J. Schropp, *Conserving first integrals under discretization with variable step size integration procedures*, J. Comput. Appl. Math., 115 (2000), pp. 503–517.
[11] L. F. Shampine, *Conservation laws and the numerical solution of ODEs*, Comput. Math. Appl. Part B, 12 (1986), pp. 1287–1296.
[12] M. Slodička and I. Cimrák, *An iterative approximation scheme for the Landau-Lifshitz-Gilbert equation*, J. Comput. Appl. Math., 169 (2004), pp. 17–32.
[13] M. Slodička and I. Cimrák, *Numerical study of nonlinear ferromagnetic materials*, Appl. Numer. Math., 46 (2003), pp. 95–111.
[14] J. Stoer and R. Bulirsch, *Introduction to Numerical Analysis*, Springer-Verlag, New York, 1983.
[15] A. Wintner, *The Analytical Foundations of Celestial Mechanics*, Princeton University Press, Princeton, NJ, 1941.

SIAM J. SCI. COMPUT.
Vol. 28, No. 3, pp. 886–900
© 2006 Society for Industrial and Applied Mathematics

SPECTRAL METHODS FOR PARTIAL DIFFERENTIAL EQUATIONS IN IRREGULAR DOMAINS: THE SPECTRAL SMOOTHED BOUNDARY METHOD*

ALFONSO BUENO-OROVIO†, VÍCTOR M. PÉREZ-GARCÍA†, AND FLAVIO H. FENTON‡

Abstract. In this paper, we propose a numerical method to approximate the solution of partial differential equations in irregular domains with no-flux boundary conditions. The idea is to embed the domain into a box and use a smoothing term to encode the boundary conditions into a modified equation that can be approached by standard spectral methods. The main features of this method are its capability to deal with domains of arbitrary shape and its easy implementation via fast Fourier transform routines. We discuss several examples of practical interest and test the results against standard numerical methods.

Key words. spectral methods, irregular domains, phase field methods, reaction-diffusion equations

AMS subject classifications. 65M70, 65T50

DOI. 10.1137/040607575

1. Introduction.

Spectral methods [1, 2, 3] are among the most extensively used methods for the discretization of spatial variables in partial differential equations and have been shown to provide very accurate approximations of sufficiently smooth solutions. Because of their high-order accuracy, spectral methods have become widespread over the years in various fields, including fluid dynamics, quantum mechanics, heat conduction, and weather prediction [4, 5, 6, 7, 8]. However, these methods have some limitations which have prevented them from being extended to many problems, where finite-difference and finite-element methods continue to be used predominantly. One limitation is that the discretization of partial differential equations by spectral methods leads to the solution of large systems of linear or nonlinear equations involving full matrices. Finite-difference and finite-element methods, on the other hand, lead to systems involving sparse matrices that can be handled by appropriate methods to reduce the complexity of the calculations substantially. Another drawback of spectral methods is their inability to handle irregularly shaped domains, which is why these methods have had limited use in many engineering problems, where finite-element methods are preferred because of their flexibility in describing complex geometries despite the computational costs associated with constructing an appropriate solution grid. Although there have been attempts to use spectral methods in irregular domains [9, 10], these approaches usually involve either incorporating finite-element preconditioning or using the so-called spectral elements. We are not aware of any

*Received by the editors April 29, 2004; accepted for publication (in revised form) January 5, 2006; published electronically June 9, 2006. This work was supported by grants BFM2003-02832 (Ministerio de Ciencia y Tecnología, Spain) and PAC-02-002 (Consejería de Ciencia y Tecnología de la Junta de Comunidades de Castilla–La Mancha (CCyT–JCCM), Spain). We also acknowledge the National Biomedical Computation Resource (NIH grant P41RR08605).

http://www.siam.org/journals/sisc/28-3/60757.html

†Departamento de Matemáticas, Universidad de Castilla–La Mancha, E.T.S.I. Industriales, Av. Camilo José Cela, 3, Ciudad Real, E-13071, Spain (alfonso.bueno@uclm.es, victor.perezgarcia@uclm.es). The work of the first author was supported by CCyT–JCCM under Ph.D. grant 03-056. The work of the first author was also partially supported by NSF grant 0320865.

‡Department of Biomedical Sciences, Cornell University, Ithaca, NY 14853, and Beth Israel Medical Center, New York, NY (fhf3@cornell.edu).

previous study in which purely spectral methods, particularly those involving fast Fourier transforms (FFTs), have been used to obtain solutions in complex irregular geometries.

In this paper we present an easy-to-use method for approximating the solution of partial differential equations in irregular domains with no-flux boundary conditions using spectral methods. The idea is based on what, in dendritic solidification, is known as the phase-field method [11]. This method is used to locate and track the interface between the solid and liquid states and has been applied to a wide variety of problems including viscous fingering [12, 13], crack propagation [14], and the tumbling of vesicles [15]. For a comprehensive review see [16].

In what follows we use the idea behind phase-field methods to illustrate how the solution of several partial differential equations can be obtained in irregular domains using spectral methods, understanding for irregular any arbitrarily shaped domain beyond the standard rectangular or circular geometries. Throughout, for simplicity, we will refer to the combination of the phase-field and spectral methods as the spectral smoothed boundary (SSB) method. Our approach consists of two steps. First, the idea of the phase-field method is formalized and its convergence analyzed for the case of homogeneous Neumann boundary conditions. Then we discuss how the new formulation is useful for the direct use of spectral methods, specifically those based on trigonometric polynomials. This formulation makes the problem suitable for efficient solution using FFTs [17]. Since it is our intention that the resulting methodology be used in a variety of problems in engineering and applied science, we have concentrated on the important underlying concepts, reserving some of the more formal questions related to these methods for a subsequent analysis.

2. The phase-field (smoothed boundary) method. In this work we focus on applying the phase-field method to partial differential equations of the form

$$\nabla(\boldsymbol{D}^{(j)}\nabla u_j) + f(u_1, \ldots, u_N, t) = \partial_t u_j \tag{2.1a}$$

for N unknown real functions u_j defined on an irregular domain $\Omega \subset \mathbb{R}^n$, where $n = 1, 2, 3$ is the spatial dimensionality of the problem, together with appropriate initial conditions $u_j(x, 0) = u_{j0}(x)$ and subject to Neumann boundary conditions

$$\vec{\boldsymbol{n}} \cdot \boldsymbol{D}^{(j)}\nabla u_j = 0 \tag{2.1b}$$

on $\partial\Omega$, where $\boldsymbol{D}^{(j)}(x)$ is a family of $n \times n$ matrices that may depend on the spatial variables. Equations (2.1a) and (2.1b) include many reaction-diffusion models, such as those describing population dynamics or cardiac electrical activity. Here we will restrict the analysis to equations of the form (2.1a), although the idea behind the method can be extended to many other problems involving complex boundaries and different types of partial differential equations.

Instead of discretizing (2.1a), the smoothed boundary method (SBM) relies on considering the auxiliary problem

$$\nabla(\phi^{(\xi)}\boldsymbol{D}^{(j)}\nabla u_j^{(\xi)}) + \phi^{(\xi)} f(u_1^{(\xi)}, \ldots, u_N^{(\xi)}, t) = \partial_t(\phi^{(\xi)} u_j^{(\xi)}) \tag{2.2}$$

for the unknown functions $u_j^{(\xi)}$ on an enlarged domain Ω' satisfying the following conditions: (i) $\Omega \subset \Omega'$ and (ii) $\partial\Omega \cap \partial\Omega' = \emptyset$. The function $\phi^{(\xi)}$ is continuous in Ω' and has the value one inside Ω and smoothly decays to zero outside Ω, with ξ

identifying the width of the decay. That is, if χ_Ω is the characteristic function of the set Ω defined as

$$\chi_\Omega = \begin{cases} 1, & x \in \Omega, \\ 0, & x \in \Omega' \backslash \Omega, \end{cases} \tag{2.3}$$

then $\phi^{(\xi)} : \Omega' \to \mathbb{R}$ is a regularized approximation to χ_Ω such that $\lim_{\xi \to 0} \phi_\xi = \chi_\Omega$.

The key idea of the SBM is that when $\xi \to 0$, the solutions $u_j^{(\xi)}$ of (2.2) on any domain Ω' with arbitrary boundary conditions on $\partial\Omega'$ satisfy $u_j^{(\xi)} \to u_j$; in other words, they tend pointwise to the solution of (2.1a), automatically incorporating the boundary conditions (2.1b). To see why, let us first realize that inside Ω the statement is inmediate since $\phi^{(\xi)} \to 1$ in Ω as $\xi \to 0$ and (2.2) becomes (2.1a). At the boundary, we consider for simplicity the situation with $n = 2$ (the extension to $n = 3$ is immediate). Assuming smoothness of $\partial\Omega$ (which in this case will be a curve) and choosing any connected curve $\Gamma \subset \partial\Omega$, we define two families of differentiable curves $\Gamma_{\delta(+)} \subset \Omega' \backslash \Omega$ and $\Gamma_{\delta(-)} \subset \Omega$ whose ends coincide with those of Γ and which tend uniformly to Γ following the parameter δ. The curves $\Gamma_{\delta(+)}$ and $\Gamma_{\delta(-)}$ are then the boundaries of a region A_δ whose boundary $\partial A_\delta = \Gamma_{\delta(+)} \cup \Gamma_{\delta(-)}$ (see Figure 2.1).

We now integrate (2.2) over A_δ:

$$\int\int_{A_\delta} \left[\nabla(\phi^{(\xi)} \boldsymbol{D}^{(j)} \nabla u_j^{(\xi)}) + \phi^{(\xi)} f(u,t) \right] dx = \int\int_{A_\delta} \partial_t(\phi^{(\xi)} u_j^{(\xi)}) dx. \tag{2.4}$$

Using the Gauss (or Green) theorem for the first term of (2.4), we obtain

$$\oint_{\partial A_\delta} \vec{\boldsymbol{n}} \cdot \phi^{(\xi)} \boldsymbol{D}^{(j)} \nabla u_j^{(\xi)} dx + \int\int_{A_\delta} \phi^{(\xi)} f(u,t) dx = \int\int_{A_\delta} \partial_t(\phi^{(\xi)} u_j^{(\xi)}) dx, \tag{2.5}$$

where $\oint$ denotes a line integral over ∂A_δ.

Now we take the limit $\xi \to 0$ in (2.5) to obtain

$$\begin{aligned} \lim_{\xi \to 0} \oint_{\partial A_\delta} \vec{\boldsymbol{n}} \cdot \phi^{(\xi)} \boldsymbol{D}^{(j)} \nabla u_j^{(\xi)} dx &= -\lim_{\xi \to 0} \int\int_{A_\delta} \phi^{(\xi)} f(u,t) dx \\ &\quad + \lim_{\xi \to 0} \int\int_{A_\delta} \partial_t(\phi^{(\xi)} u_j^{(\xi)}) dx \\ &= m(A_\delta) \left[-\phi^{(\xi)} f(u,t) + \partial_t(\phi^{(\xi)} u_j^{(\xi)}) \right]_{x=\zeta}, \end{aligned} \tag{2.6}$$

where the last equality comes from the mean value theorem for integrals and $m(A_\delta)$ is the measure of the set A_δ. Here we assume that the solutions to (2.2) and its time derivatives are bounded so that the right-hand side of (2.6) is finite. On the left-hand side we decompose $\oint_{\partial A_\delta}$ as $\int_{\Gamma_{\delta(+)}} + \int_{\Gamma_{\delta(-)}}$. It is straightforward that $\lim_{\xi \to 0} \int_{\Gamma_{\delta(+)}} \vec{\boldsymbol{n}} \cdot \phi^{(\xi)} \boldsymbol{D}^{(j)} \nabla u_j^{(\xi)} dx = 0$, since in this limit $\phi^{(\xi)} = 0$ over all $\Gamma_{\delta(+)}$ and $\phi^{(\xi)} = 1$ on $\Gamma_{\delta(-)}$. As we were interested in proving that the boundary conditions are satisfied, we now make the width of the integration region A_δ tend to zero. Since $\Gamma_{\delta(-)} \to \Gamma$ as $\delta \to 0$, and $\lim_{\delta \to 0} m(A_\delta) = 0$, we obtain

$$\begin{aligned} \lim_{\delta \to 0} \lim_{\xi \to 0} \oint_{\partial A_\delta} \vec{\boldsymbol{n}} \cdot \phi^{(\xi)} \boldsymbol{D}^{(j)} \nabla u_j^{(\xi)} dx &= \lim_{\delta \to 0} \int_{\Gamma_{\delta(-)}} \vec{\boldsymbol{n}} \cdot \boldsymbol{D}^{(j)} \nabla u_j^{(\xi)} dx \\ &= \int_\Gamma \vec{\boldsymbol{n}} \cdot \boldsymbol{D}^{(j)} \nabla u_j^{(\xi)} dx = 0. \end{aligned} \tag{2.7}$$

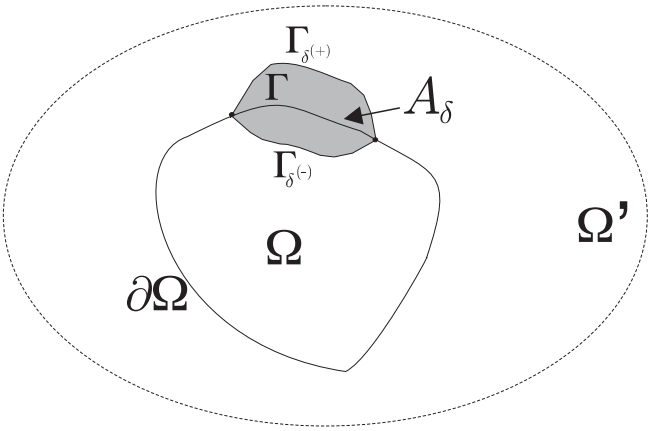


FIG. 2.1. *Illustration of an irregular domain* Ω *with an example of the connected curve* Γ *and the domain* A_δ *used in the proof of convergence of the method.*

Since (2.7) is true for any boundary segment Γ, we obtain the final result that, in the limit $\xi \to 0$, (2.2) satisfies $\vec{\boldsymbol{n}} \cdot \boldsymbol{D}^{(j)} \nabla u_j^{(\xi)} = 0$ for $j = 1, \ldots, N$, i.e., the boundary conditions.

The idea of the SBM is then to consider (2.2) for a small but finite ξ and to discretize this problem instead of (2.1a) and (2.1b). The main advantage is that one can search for the approximation $u^{(\xi)}$ on any enlarged domain Ω' such that $\Omega \subset \Omega'$. The enlarged discrete problem can then be solved with any proper boundary conditions on $\partial\Omega'$, since the fulfillment of the boundary conditions for u on $\partial\Omega$ is guaranteed in the limit $\xi \to 0$.

3. The SSB method. We want to discretize (2.2) on an enlarged domain Ω', chosen to be a rectangular region containing Ω, and we will expand $u^{(\xi)}$ in the basis of Cartesian products of trigonometric polynomials $e^{ik_x x} e^{ik_y y}$. Thus, we will seek an extension of the solution u of (2.1a) and (2.1b) that is periodic on the enlarged region Ω'.

Note that since $\phi^{(\xi)}$ is located inside the time derivative of the right term of (2.2), it is possible for the integration domain itself to evolve in time, and thus this method could be used to solve moving boundary problems once a coupling equation is added for the movement of $\phi^{(\xi)}$. However, here we will deal only with stationary integration domains; thus, $\partial_t \phi^{(\xi)} = 0$ and the right side of (2.2) can be simplified as $\phi^{(\xi)} \partial_t u_j^{(\xi)}$. Dividing (2.2) by $\phi^{(\xi)}$, we get

$$\nabla \log \phi^{(\xi)} \cdot \boldsymbol{D^{(j)}} \nabla u_j^{(\xi)} + \nabla(\boldsymbol{D^{(j)}} \nabla u_j^{(\xi)}) + f(u^{(\xi)}, t) = \partial_t u_j^{(\xi)}, \tag{3.1}$$

which is an easier version of the equation of the SBM that we will use here to perform the numerical simulations.

To implement numerically any solution method for (3.1), we need to make a specific choice for $\phi^{(\xi)}$. In practice, any method that produces a smooth characteristic function can be used. In the context of phase-field methods, the standard procedure for obtaining the values of $\phi^{(\xi)}$ (which is called the "phase-field") is to integrate an auxiliary diffusion equation of the form $\partial_t \phi = \xi^2 \Delta\phi + (2\phi - 1)/2 - (2\phi - 1)^3/2$, with initial conditions $\phi^{(\xi)}(t = 0) = \chi_\Omega$, until a steady state is reached [16, 18]. Alternatively, since we seek only a smoothed boundary, we choose to obtain $\phi^{(\xi)}$ from

χ_Ω using a convolution of the form

$$\phi^{(\xi)} = \chi_\Omega * G^{(\xi)}, \tag{3.2}$$

where $G^{(\xi)}$ is any family of functions such that $\lim_{\xi\to 0} G^{(\xi)}(x) = \delta(x)$, with $\delta(x)$ being the Dirac delta function. In particular, Gaussian functions of the form

$$G^{(\xi)}(x) = \prod_{k=1}^{n} \exp(-x_k^2/\xi^2) \tag{3.3}$$

can be chosen. In this paper, all the functions $\phi^{(\xi)}$ have been obtained using this n-dimensional discrete convolution given by (3.3), which also have the advantage of being efficiently computed using FFTs. An example of the creation of $\phi^{(\xi)}$ is shown in Figure 3.1, where it can be seen that the width of the interface in which $\phi^{(\xi)}$ changes from zero to one depends on the value used for ξ (in fact, it is of order ξ).

To avoid computational difficulties for very small values of $\phi^{(\xi)}$, we approximate $\log\phi^{(\xi)} \approx \log(\phi^{(\xi)} + \epsilon)$, where ϵ is the machine precision. Numerically, $\phi^{(\xi)}$ and $(\phi^{(\xi)} + \epsilon)$ are equal up to roundoff errors, but this correction bounds the value of $\log\phi^{(\xi)}$ as $\phi^{(\xi)} \to 0$. Due to the effect of the boundaries in the numerical stability of the scheme, we also need a reasonable margin between the boundaries of the physical and the enlarged domains. We found that a margin of value $M = 10\xi$ is sufficient for all the simulations to be stable [19]. Finally, all the spatial derivatives of $u^{(\xi)}$ in (3.1) are computed in Cartesian coordinates using differentiation in Fourier space, which as mentioned previously implicitly assumes periodic boundary conditions on $\partial\Omega'$. It is significant that only Fourier transforms are used for these calculations instead of differentiation matrices, thereby avoiding the generation and storage of these matrices and yielding more efficient codes and shorter execution times, especially when FFT routines are used.

In this paper, we are not concerned with designing the most efficient implementation of the SSB method, but only with proving that such a method can be used to integrate partial differential equations in irregular domains. Thus, for time integration we use a simple second-order explicit method. In the particular case when all the coefficients of the diffusion tensor $\boldsymbol{D}^{(j)}$ are constants, we can write (3.1) in the form

$$\mathcal{L}u^{(\xi)} + \mathcal{N}(u^{(\xi)}, t) = \partial_t u^{(\xi)}, \tag{3.4}$$

where $\mathcal{L}u^{(\xi)} = \nabla(\boldsymbol{D}^{(j)}\nabla u_j^{(\xi)})$ is the linear part of the equation, while the nonlinear term is represented by $\mathcal{N}(u^{(\xi)}, t) = \nabla\log\phi^{(\xi)} \cdot \boldsymbol{D}^{(j)}\nabla u_j^{(\xi)} + f(u^{(\xi)}, t)$. Then a second-order in time operator splitting scheme of the form

$$U^{(\xi)}(t + \Delta t) = e^{\mathcal{L}\Delta t/2} e^{\mathcal{N}\Delta t} e^{\mathcal{L}\Delta t/2} U^{(\xi)}(t) \tag{3.5}$$

can be used to solve the equation in time [20]. For the examples to be presented later, we solve the nonlinear term by a second-order (half-step) explicit method and integrate the linear part exactly in Fourier space by exponential integration, which reduces the stiffness of the problem considerably and allows for the use of larger time steps.

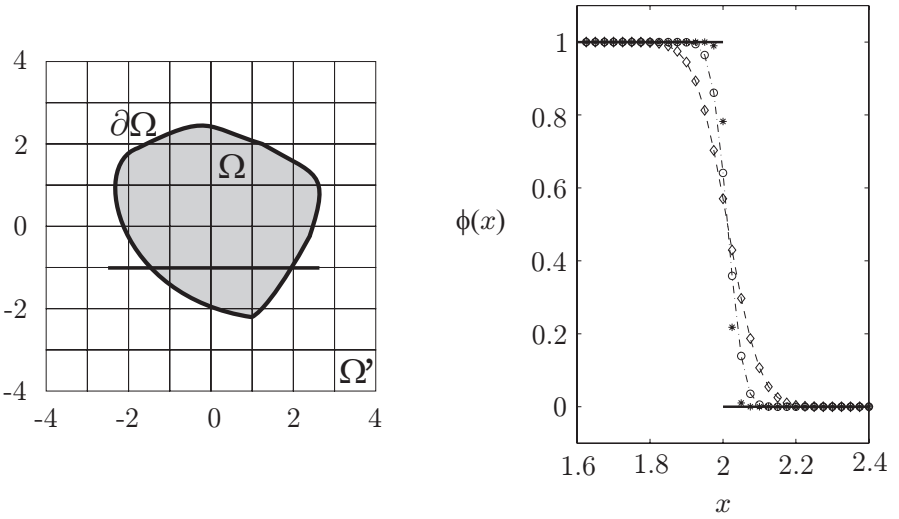


FIG. 3.1. *Left: Example of an irregular domain* Ω *defined on a Cartesian grid and an enlarged domain* Ω'*. Right: Smoothing of the irregular boundary in a one-dimensional section of the domain. The solid line on the right shows a small section of the characteristic function* χ_Ω *(with value* 0 *or* 1*) corresponding to part of the thicker line shown in the left part of the figure. Smoothed boundary functions* $\phi^{(\xi)}$ *obtained from* χ_Ω *for* $\xi = 0.10$*,* 0.05*, and* 0.025 *are labeled by diamonds, circles, and stars, respectively.*

4. Examples of the methodology.

4.1. The heat equation. As a first example, we will consider a simple linear heat equation. This first case will allow us to make a quantitative study of the errors of the SSB method. Specifically, we are interested in solving the following heat equation with sources:

$$\partial_t u = D\Delta u - r\cos(2\theta) \tag{4.1}$$

in the annulus Ω defined by $1 \le r \le 2$ with homogeneous Neumann boundary conditions on $\partial\Omega$, $\partial_r u|_{r=1} = \partial_r u|_{r=2} = 0$. The diffusion coefficient is taken to be constant with value $D = 1$. Figure 4.1 shows one example of the generation of the smoothed boundary for this domain.

Equation (4.1) in this geometry has an explicit steady state solution of the form

$$u_{st}(r,\theta) = \left(\frac{1}{5}r^3 - \frac{31}{50}r^2 - \frac{8}{25}\frac{1}{r^2}\right)\cos(2\theta), \tag{4.2}$$

which can be used to determinate the contribution to the error introduced by the smoothing term in (2.2). Thus, (4.1) is evolved in time using the SSB method and this explicit steady state as initial data until the new steady state is reached and deviations from the initial condition are measured. Figure 4.2 (left) shows the L^2-norm of the error of several simulations for different values of ξ and grid resolutions. Note that in general these maximum errors decrease as the thickness of the interface is reduced. The left and right interior plots show the log-log plots of the L^2-norm of the error versus ξ (values at $\eta = 5$, see below), indicating an $O(\xi^2)$ accuracy of the method as $\xi \to 0$ when the integration domain Ω does not have sharp corners.

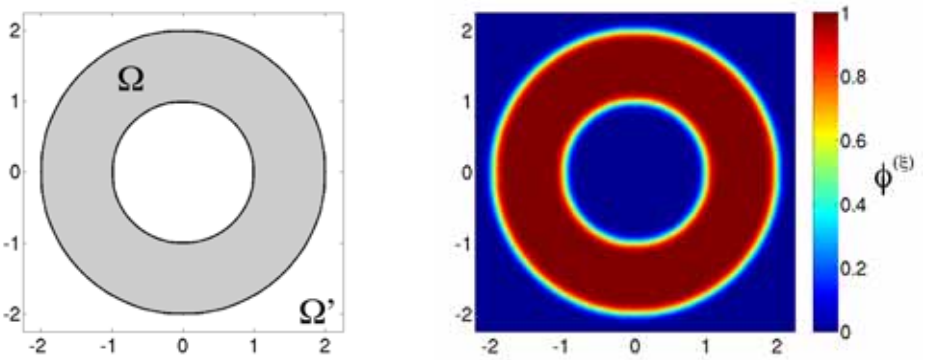


FIG. 4.1. *Left: The rectangular domain Ω' in which* (4.1) *is solved using the SSB method, with the irregular domain Ω (an annulus) shown in gray. Right: Smoothed boundary function $\phi^{(\xi)}$ given by* (3.2) *for $\xi = 0.10$.*

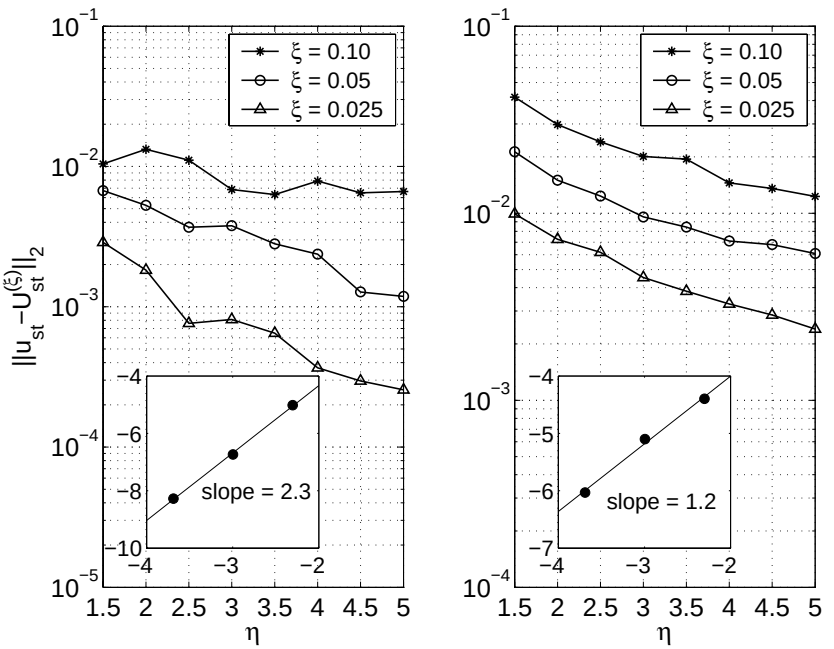


FIG. 4.2. *Left: L^2-error of the numerical steady state solution of* (4.1) *in the annulus compared to the exact steady solution as a function of the parameter $\eta = \xi/\Delta x$. Right: Same as left but for the quarter-annulus geometry. See text for details.*

Although $\phi^{(\xi)}$ is a continuous function, it is necessary to have a grid fine enough to properly resolve the boundary layers in which it quickly changes from zero to one. For this reason, errors are represented in Figure 4.2 not as a function of the number of grid points but as a function of the parameter $\eta = \xi/\Delta x$, which gives an idea of the number of points that lie in the interface (with number of points varying in this case from 90 to 300 points for $\xi = 0.10$, from 150 to 500 points for $\xi = 0.05$, and from 270 to 900 points for $\xi = 0.025$). To ensure that this error is produced only by the spatial

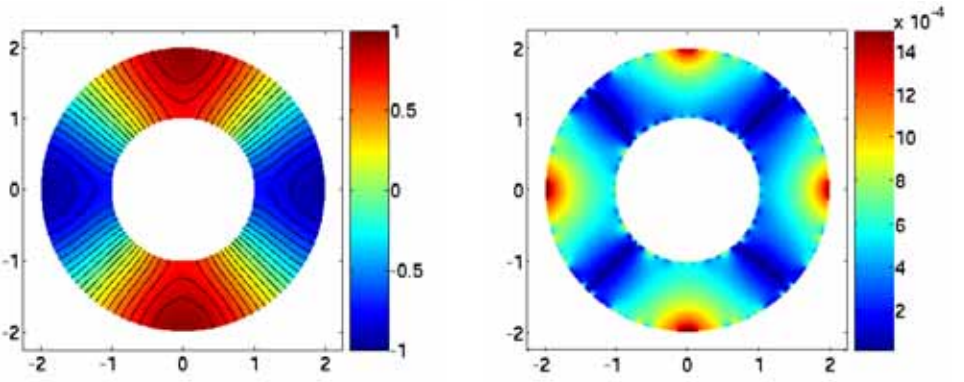


FIG. 4.3. *Left: Solution of* (4.1) *obtained with the SSB method at steady state in the annulus* $1 \leq r \leq 2$. *Grid size is* 540×540 *with* $\xi = 0.025$. *Right: Spatial distribution of the absolute error* $|u_{st} - U_{st}^{(\xi)}|$ *over the annulus for the simulation solution compared to the analytical steady solution.*

discretization and is not due to the order of the method chosen to perform the time integration, we have also run the simulations with a first-order explicit (Euler) time-integration method and obtained errors of the same order of magnitude. Figure 4.3 shows the solution to (4.1) obtained at stationary state with the SSB method. Contour lines are also included to illustrate that the no-flux boundary conditions at $r = 1$ and $r = 2$ are satisfied.

The analytical steady state solution of (4.1) also satisfies homogeneous Neumann boundary conditions on the quarter-annulus delimited by $1 \leq r \leq 2$, $0 \leq \theta \leq \pi/2$ (see Figure 4.4), which also allows us to use this related geometry to show how the SSB method performs when sharp corners are present in a given geometry. As before, the L^2-norms of the deviations from the analytical steady state are shown in Figure 4.2 (right) for the simulations on the quarter-annulus. Both geometries show similar convergence properties. However, errors are slightly larger than for the full annulus due to the presence of the sharp corners, which become slightly blunted when the smoothed boundary is generated. This is translated into a loss of accuracy of the method from $O(\xi^2)$ to $O(\xi)$ for regions with sharp corners (see interior plot in Figure 4.2, right). This can also be seen in Figure 4.4, which shows the error distribution $|u_{st} - U_{st}^{(\xi)}|$ at steady state over the domain Ω, along with the corresponding solution. See how errors are larger when close to the corners of the domain.

In Figures 4.3 and 4.4, we have shown the solutions of (4.1) within the physical domains of interest, Ω. However, solutions $U^{(\xi)}$ are calculated over the entire extended domain Ω'. Figure 4.5 shows the solutions over Ω' in the full and quarter-annulus examples. While no-flux boundary conditions are implemented along $\partial\Omega$, the overall solution has periodic boundary conditions. Note that, as the solution $U^{(\xi)}$ is not discontinuous on $\partial\Omega$, solutions do not present Gibbs phenomena due to the irregular boundaries.

An important advantage of the SSB method is that when the new formulation given by (3.1) is used, separate equations are not written for the boundaries, as the solution automatically adapts to satisfy the boundary conditions on $\partial\Omega$, which results in a very simple computational implementation. Alterations to the domain geometry therefore can be handled straightforwardly, without generating and implementing additional boundary condition equations. As an example, we present in Figure 4.6

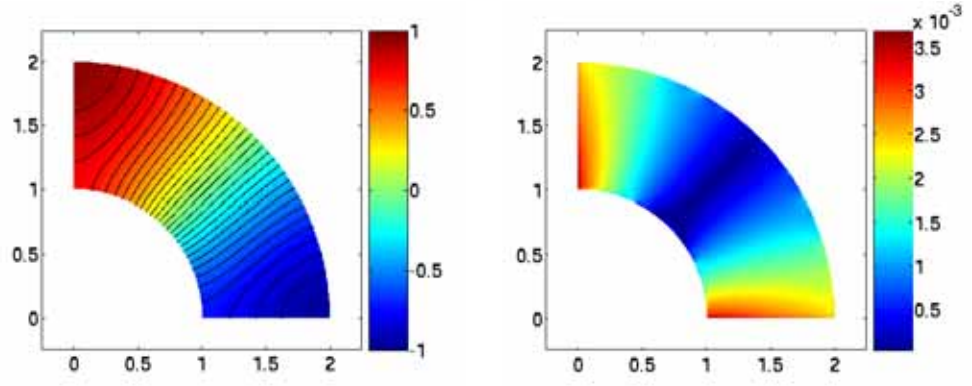


FIG. 4.4. *Left: Solution to* (4.1) *in the quarter-annulus* $1 \leq r \leq 2$, $0 \leq \theta \leq \pi/2$ *at steady state using the SSB method. Grid resolution is* 500×500 *and* $\xi = 0.025$. *Right: Spatial distribution of absolute error* $|u_{st} - U_{st}^{(\xi)}|$ *over the quarter-annulus for the simulation solution compared to the analytical steady solution.*

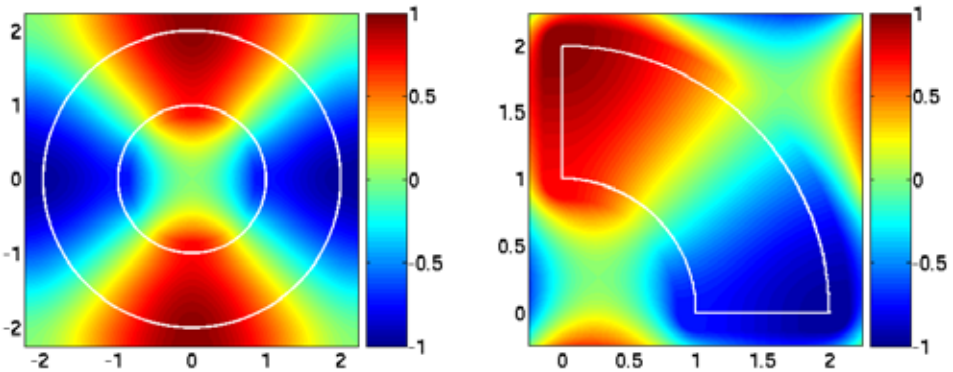


FIG. 4.5. *Solution to* (4.1) *using the SSB method over the entire domain of integration* Ω' *for the annulus shown in Figure* 4.3 *(left) and the quarter-annulus shown in Figure* 4.4 *(right). Note that the periodicity along* $\partial\Omega'$ *imposed by the FFT does not alter the solution in* Ω *or at the boundary* $\partial\Omega$ *(shown in white), where the zero-flux boundary conditions are satisfied.*

the solution to (4.1) in a more complicated geometry that combines both polar and Cartesian coordinates. As an analytic steady solution to the equation cannot be easily obtained in this domain, we evolve it until time $t = 5$ and compare our solution to the one obtained using finite elements in the finest mesh that our machine could handle (P1-Lagrangian elements, mesh made of 31152 nodes and 61440 triangles). At this resolution, and as shown in Figure 4.6, maximum differences between the solutions obtained by both methods are small.

4.2. The Allen–Cahn equation. An important partial differential equation which arises in the modeling of the formation and motion of phase boundaries is the Allen–Cahn [21] equation

$$\begin{aligned} \partial_t u &= \epsilon^2 \Delta u - f(u), && x \text{ in } \Omega, \\ \vec{n} \cdot \nabla u &= 0, && x \text{ on } \partial\Omega, \end{aligned} \tag{4.3}$$

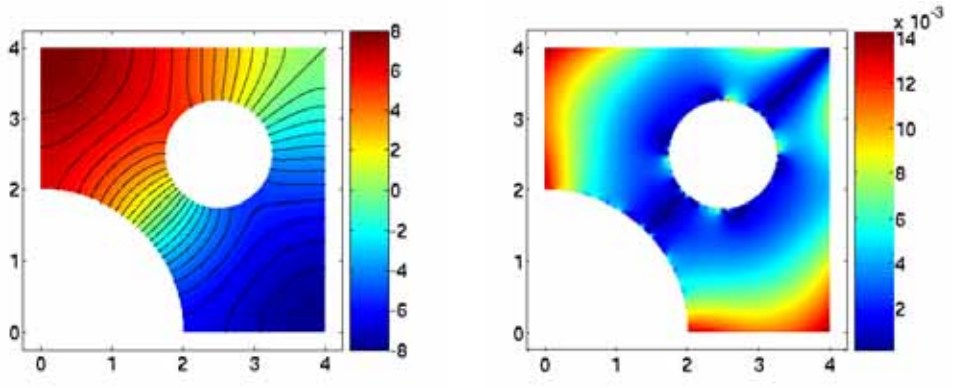


FIG. 4.6. *Solution to* (4.1) *at time* $t = 5$ *in a more complicated domain (left) and spatial distribution of the difference between the solutions obtained by the SSB method and using finite elements (right). Grid resolution is* 400×400 *and* $\xi = 0.05$.

where ϵ is a small positive constant and $f(u)$ is the derivative of a potential function $W(u)$ that has two wells of equal depth. For simplicity, we will assume that $W(u) = (u^2-1)^2/4$, which makes $f(u) = u^3-u$. In this manner, the Allen–Cahn equation may be seen as a simple example of a nonlinear reaction-diffusion equation. As explained in [2], this equation has three fixed-point solutions, $u = -1$, $u = 0$, and $u = 1$. The middle state is unstable, but the states $u = \pm 1$ are attracting, and solutions tend to exhibit flat areas close to these values separated by interfaces of increasing sharpness as the control parameter ϵ is reduced to zero. Figure 4.7 shows the solution of the Allen–Cahn equation at $t = 25$ solved with Neumann boundary conditions on an annulus with a z-shaped hole using the SSB method, with $\epsilon = 0.1$. The annulus structure is given by $1 \le r \le 5$ and the z-hole is formed using radii at 2, 3, and 4 and angles in steps of 15 degrees (15, 30, 60, and 75 degrees). As an initial condition, two positive and two negative Gaussian functions located in different parts of the domain were chosen:

$$u(r,\theta,0) = \sum_{i=1}^{n=4} (-1)^{n+1} \exp(-20((x - r_i\cos\theta_i)^2 + (y - r_i\sin\theta_i)^2)), \tag{4.4}$$

where pairs $(r_1,\theta_1) = (1.5,\pi/6)$, $(r_2,\theta_2) = (4,\pi/12)$, $(r_3,\theta_3) = (4.5,\pi/3)$, and $(r_4,\theta_4) = (4,11\pi/24)$ represent the positions of the Gaussians in polar coordinates.

For comparison we also solved the Allen–Cahn equation using finite elements in space as in the previous example (P2-Lagrangian elements, mesh made of 8304 elements and 348 boundary elements, 16956 degrees of freedom) and implicit backward differentiation formula for time integration. Due to the sharp transition of the solution between -1 and 1, a slight difference in the position of the interfaces can be seen as a significant error when comparing the solutions obtained by the two methods, but as shown in Figure 4.7 (right) both the SSB and finite elements solutions are in good agreement.

4.3. Reaction-diffusion equations and excitable media. Models of excitable media form another significant class of nonlinear parabolic partial differential equations and describe systems as diverse as chemical reactions [22, 23], aggregation of amoebae in the cellular slime mold [illegible] [24], calcium waves

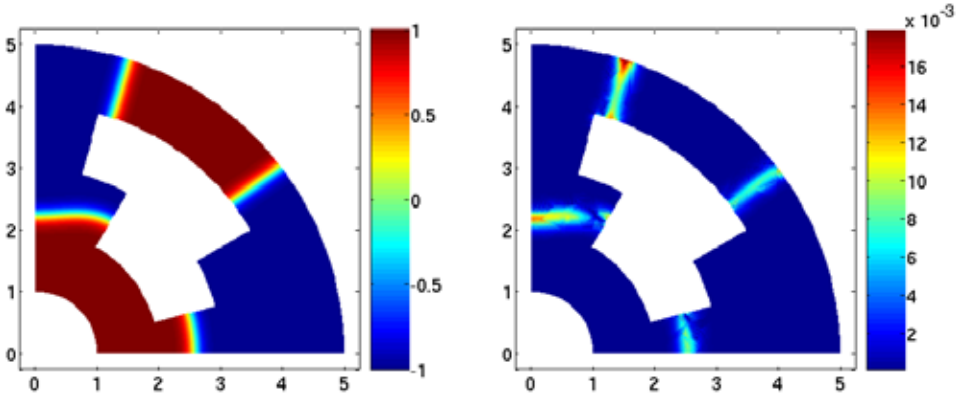


FIG. 4.7. *Solution of the Allen–Cahn equation* (4.3) *at time* $t = 25$ *using the SSB method (left) and errors when compared to the solution obtained using* P2*-Lagrangian finite elements (right). Grid size is* 360×360 *with* $\xi = 0.025$ *for the SSB method.*

[25], and the electrical properties of neural [26] and cardiac cells [27, 28], among others. The equations of excitable systems extend the Allen–Cahn equation by including one or more additional variables that govern growth and decay of the waves. Solutions of excitable media consist of excursions in state space from a stable rest state and a return to rest, with the equations describing the additional variables determining the time courses of excitation and recovery. In spatially extended systems, diffusive coupling allows excitation to propagate as nonlinear waves, and in multiple dimensions complex patterns can be formed, including two-dimensional spiral waves [22, 23, 24, 25, 29] and their three-dimensional analogues, scroll waves [30, 31]. Well-known examples of excitable media equations include the Hodgkin–Huxley [26] model of cells and its generalized simplification, the FitzHugh–Nagumo [32] model.

The dynamics of wave propagation in excitable media has been studied extensively in regular domains. However, the complex geometry inherent to some systems, such as the heart, often can have a significant influence on wave stability and dynamics [33]. This fact, combined with the need for high-order accuracy to resolve the sharp wave fronts characteristic of cardiac models, should make the SSB method a useful tool for studying electrical waves in realistic heart geometries. Figure 4.8 shows an example of a propagating wave of action potential in both an idealized (left) and a realistic (right) slice [34] of ventricular tissue using the SSB method and a phenomenological ionic cell model [31, 33] with equations of the form

$$\partial_t u(\mathbf{x},t) = \nabla\cdot(\boldsymbol{D}\nabla u) - J_{fi}(u,v) - J_{so}(u) - J_{si}(u,w), \tag{4.5a}$$

$$\partial_t v(\mathbf{x},t) = \Theta(u_c-u)(1-v)/\tau_v^-(u) - \Theta(u-u_c)v/\tau_v^+, \tag{4.5b}$$

$$\partial_t w(\mathbf{x},t) = \Theta(u_c-u)(1-w)/\tau_w^- - \Theta(u-u_c)w/\tau_w^+, \tag{4.5c}$$

$$J_{fi}(u,v) = -\frac{v}{\tau_d}\Theta(u-u_c)(1-u)(u-u_c), \tag{4.5d}$$

$$J_{so}(u) = \frac{u}{\tau_0}\Theta(u_c-u) + \frac{1}{\tau_r}\Theta(u-u_c), \tag{4.5e}$$

$$J_{si}(u,w) = -\frac{w}{2\tau_{\mathrm{si}}}(1+\tanh[k(u-u_c^{\mathrm{si}})]), \tag{4.5f}$$

$$\tau_v^-(u) = \Theta(u-u_v)\tau_{v1}^- + \Theta(u_v-u)\tau_{v2}^-, \tag{4.5g}$$

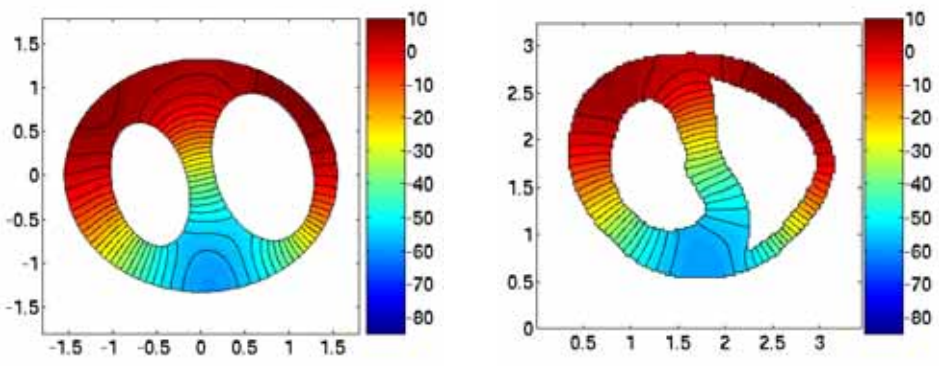


FIG. 4.8. *Propagating wave of electrical potential in an idealized (left) and a realistic (right) slice of ventricular tissue using the SSB method. Grid size is* 400×400 *and* $\xi = 0.025$ *in both cases. The color code denotes tissue voltage in mV, where red corresponds to cells with higher potential (depolarization) and blue to cells returning to resting state (repolarization).*

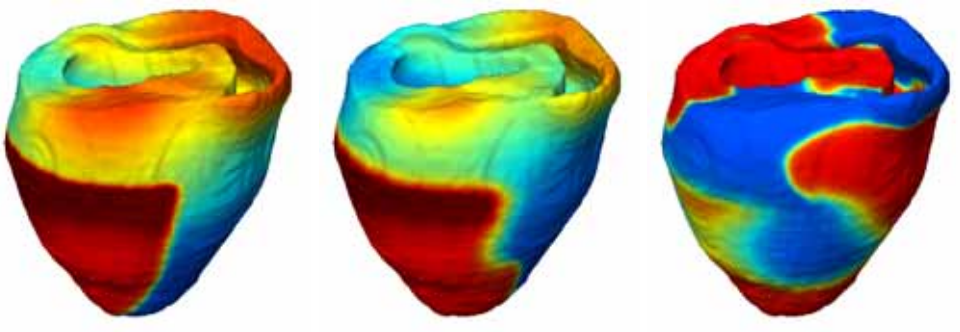


FIG. 4.9. *Propagating wave of electrical potential and generation of a spiral wave in a realistic three-dimensional model of rabbit ventricles using the SSB method. See text for details.*

where u is the membrane potential; J_{fi}, J_{so}, and J_{si} are phenomenological currents; v and w are ionic gate variables; and $\boldsymbol{D}$ is the diffusion tensor (isotropic for these simulations, with value $D = 1$ cm^2/s). In all these formulas, $\Theta(x)$ is the standard Heaviside step function defined by $\Theta(x) = 1$ for $x \geq 0$ and $\Theta(x) = 0$ for $x < 0$, and the set of parameters of the model is chosen to reproduce different cellular dynamics measured experimentally (for simulations presented here, $\tau_d = 0.25$, $\tau_r = 50$, $\tau_{\mathrm{si}} = 45$, $\tau_0 = 8.3$, $\tau_v^+ = 3.33$, $\tau_{v1}^- = 1000$, $\tau_{v2}^- = 19.2$, $\tau_w^+ = 667$, $\tau_w^- = 11$, $u_c = 0.13$, $u_v = 0.055$, and $u_c^{\mathrm{si}} = 0.85$). Contour lines demonstrate that boundary conditions are correctly approximated by the method in both geometries.

Finally, Figure 4.9 illustrates the application of the proposed method to realistic three-dimensional geometries, using for that purpose the full anatomical model of rabbit heart ventricles. All the information in the ventricular model originally digitalized in a finite elements mesh, including fiber orientation, was interpolated and mapped onto a regular Cartesian grid slightly larger than the structure, of sizes $L_x = 3.4$, $L_y = 3.2$, and $L_z = 3.7$ cm and $150 \times 150 \times 160$ grid points, and smoothed with a value of $\xi = 0.05$. The simulation shows the creation of a spiral wave of electrical activity in the rabbit ventricles, solutions to the reaction-diffusion equations that are

closely related to the development of arrhythmias [29, 31, 33]. In the left image, a second excitation shock is applied in the wake of the preceding excitation wave to initiate the spiral wave. In the first moments the second shock cannot move upwards (central panel) since the tissue has not yet recovered excitability, but starts curving at the bottom of the heart. The right image of Figure 4.9 shows the evolution of the self-maintained spiral wave in the tissue. The simulation is perfectly handled by the SSB method, and the extension of the algorithm is absolutely straightforward from two to three dimensions.

5. Discussion and future work. In this paper, we have presented a new method for implementing homogeneous Neumann boundary conditions using spectral methods for several problems of general interest. The SSB method offers several advantages over finite-difference and finite-element alternatives. Because ghost cells are not needed, the implementation of boundary conditions requires less coding than finite-difference stencils. The use of simple Cartesian grids also makes the SSB method easier to use than finite elements with multiple domain shapes, since [illegible]. Furthermore, the use of FFT routines in the SSB method ensures efficiency and also makes extension of the method to three dimensions straightforward using well-established routines. Since the method is directly based on the FFT, it is very simple to implement on high performance computers by using native parallel or vector FFT libraries.

The most significant limitation of the SSB method is that its accuracy is directly controlled by the value of parameter ξ, and therefore it will be more moderate than the accuracy achieved by any spectral method in a typical rectangular domain. The error of the method also depends directly on the ratio of the width of the smoothed boundary ξ to the spatial step Δx; this implies that when using uniform grids the number of points in the discretization needs to be large. This limitation is perhaps not important for certain classes of problems in which the solution contains steep wavefronts or other sharp features that require a fine spatial resolution to correctly reproduce the dynamics of the system, such as electrical waves in cardiac tissue or shock waves in fluid mechanics. However, for problems with smooth behavior, such as the heat equation with slowly varying sources, the adequate reduction of error in domains with irregular boundaries using this method may require an increase in spatial resolution of a factor of 10 or 20 in each direction of the mesh when compared to what is typically needed to obtain the same accuracy using a spectral method in a rectangular domain. This need for a large number of points is also a clear disadvantage in terms of computational efficiency, and since the accuracy of the method is controlled more by the width of the smoothed boundary rather than by the order of the method used to compute the spatial derivatives, an implementation of the method using a lower order scheme in space, such as the one presented in [18] using finite differences, can be significantly more efficient than the implementation presented here using spectral methods.

Thus, an important future extension of this work is to try to improve the performance of the SSB method for problems that do not track features with sharp spatial gradients. For instance, if the boundary is stationary, it could be useful to compute the spatial derivatives using any numerical scheme featuring adaptability, such as a combination of a nonuniform grid with extra resolution along the boundaries and a nonuniform fast Fourier transform (NFFT), or the use of wavelets [35]. However, if the boundary moves over time, it might be more efficient to use a fine spatial discretization than to keep track of the boundary for such problems. Other planned

future work includes properly handling complex anisotropies in the diffusion matrices, such as those found in cardiac muscle, examining whether the method can be used to satisfy other types of boundary conditions including Dirichlet and Robin, and implementing nonstationary boundaries.

In conclusion, we have presented a new numerical method which imposes homogeneous Neumann boundary conditions in complex geometries using spectral methods. We have used this method to solve various partial differential equations in domains with irregular boundaries and have found good agreement between our solutions and the ones obtained by standard numerical methods. Along with the overall advantage of allowing domains of different shapes to be considered with spectral methods in a very simple way, this method also offers highly accurate discretizations of spatial derivatives, ease of implementation, straightforward extension to three dimensions, and applicability to a wide variety of equations. Moreover, SSB codes need not change to implement different geometries since all the information on the geometry is contained in the function $\phi^{(\xi)}$, with the additional advantage that this function is easy to generate and, unlike finite-element methods, does not require the use of special software for grid generation.

Acknowledgments. We would like to thank Elizabeth M. Cherry for useful discussions and valuable comments on the manuscript. A. Bueno-Orovio also would like to thank the Physics Department at Hofstra University for its hospitality during his visit there (July–September, 2003).

REFERENCES

[1] B. Fornberg, *A Practical Guide to Pseudospectral Methods*, Cambridge University Press, Cambridge, UK, 1996.

[2] Ll. N. Trefethen, *Spectral Methods in MATLAB*, SIAM, Philadelphia, 2000.

[3] J. M. Sanz-Serna, *Fourier techniques in numerical methods for evolutionary problems*, in Proceedings of the 3rd Annual Granada Seminar on Computational Physics, P. L. Garrido and J. Marro, eds., Lecture Notes in Phys. 448, Springer-Verlag, Berlin, 1995, pp. 145–200.

[4] C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang, *Spectral Methods in Fluid Dynamics*, Springer-Verlag, Berlin, 1988.

[5] D. Gottlieb and S. A. Orszag, *Numerical Analysis of Spectral Methods: Theory and Applications*, SIAM, Philadelphia, 1977.

[6] R. Peyret, *Spectral Methods for Incompressible Viscous Flow*, Springer-Verlag, New York, 2002.

[7] I. D. Iliev, E. Kh Khristov, and K. P. Kirchev, *Spectral Methods in Soliton Equations*, CRC Press, Boca Raton, FL, 1994.

[8] G. Ben-Yu, *Spectral Methods and Their Applications*, World Scientific, River Edge, NJ, 1998.

[9] S. A. Orszag, *Spectral methods for problems in complex geometries*, J. Comput. Phys., 37 (1980), pp. 70–92.

[10] K. Z. Korczak and A. T. Patera, *An isoparametric spectral element method for solution of the Navier-Stokes equations in complex geometry*, J. Comput. Phys., 62 (1986), pp. 361–382.

[11] A. Karma and W.-J. Rappel, *Quantitative phase-field modeling of dendritic growth in two and three dimensions*, Phys. Rev. E, 57 (1998), pp. 4323–4349.

[12] R. Folch, J. Casademunt, A. Hernández-Machado, and L. Ramírez-Piscina, *Phase-field model for Hele-Shaw flows with arbitrary viscosity contrast.* I. *Theoretical approach*, Phys. Rev. E, 60 (1999), pp. 1724–1733.

[13] R. Folch, J. Casademunt, and A. Hernández-Machado, *Phase-field model for Hele-Shaw flows with arbitrary viscosity contrast.* II. *Numerical study*, Phys. Rev. E, 60 (1999), pp. 1734–1740.

[14] A. Karma, D. Kessler, and H. Levine, *Phase-field model of mode* III *dynamic fracture*, Phys. Rev. Lett., 87 (2001), 045501.

[15] T. Biben and C. Misbah, *Tumbling of vesicles under shear flow within an advected-field approach*, Phys. Rev. E, 67 (2003), 031908.

[16] R. Gonzalez-Cinca, R. Folch, R. Benitez, L. Ramirez-Piscina, J. Casademunt, and A. Hernandez-Machado, *Phase-field models in interfacial pattern formation out of equilibrium*, in Advances in Condensed Matter and Statistical Mechanics, E. Korucheva and R. Cuerno, eds., Nova Science Publishers, Hauppauge, NY, 2004, pp. 203–236.

[17] M. Frigo and S. G. Johnson, *FFTW: An adaptive software architecture for the FFT*, in Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, IEEE, Seattle, WA, 1998, pp. 1381–1384.

[18] F. H. Fenton, E. M. Cherry, A. Karma, and W. J. Rappel, *Modeling wave propagation in realistic heart geometries using the phase-field method*, Chaos, 15 (2005), 013502.

[19] A. Bueno-Orovio and V. M. Pérez-García, *Spectral smoothed boundary methods: The role of external boundary conditions*, Numer. Methods Partial Differential Equations, 22 (2006), pp. 435–448.

[20] G. Strang, *On the construction and comparison of difference schemes*, SIAM J. Numer. Anal., 5 (1968), pp. 506–517.

[21] S. M. Allen and J. W. Cahn, *A microscopic theory for antiphase boundary motion and its application to antiphase domain coarsening*, Acta Metall. Mater., 27 (1979), pp. 1085–1095.

[22] A. N. Zaikin and A. M. Zhabotinsky, *Concentration wave propagation in two-dimensional liquid-phase self-oscillating system*, Nature, 225 (1970), pp. 535–537.

[23] M. Bar, Ch. Zlicke, M. Eiswirth and G. Ertl, *Theoretical modeling of spatiotemporal self-organization in a surface catalyzed reaction exhibiting bistable kinetics*, J. Chem. Phys., 96 (1992), pp. 8595–8604.

[24] P. Foerster, S. C. Muller, and B. Hess, *Curvature and spiral geometry in aggregation patterns of Dictyostelium discoideum*, Development, 109 (1990), pp. 11–16.

[25] M. E. Harris-White, S. A. Zanotti, S. A. Frautschy, and A. C. Charles, *Spiral intracellular calcium waves in hippocampal slice cultures*, J. Neurobiol., 79 (1998), pp. 1045–1052.

[26] A. L. Hodgkin and A. F. Huxley, *A quantitative description of membrane current and its application to conduction and excitation in nerve*, J. Physiol., 117 (1952), pp. 500–544.

[27] G. W. Beeler and H. Reuter, *Reconstruction of the action potential of ventricular myocardial fibres*, J. Physiol., 268 (1977), pp. 177–210.

[28] C. Luo and Y. Rudy, *A model of the ventricular cardiac action potential. Depolarization, repolarization, and their interaction*, Circ. Res., 68 (1991), pp. 1501–1526.

[29] J. M. Davidenko, A. M. Pertsov, R. Salomonsz, W. Baxter, and J. Jalife, *Stationary and drifting spiral waves of excitation in isolated cardiac muscle*, Nature, 355 (1992), pp. 349–351.

[30] R. A. Gray, A. M. Pertsov, and J. Jalife, *Spatial and temporal organization during cardiac fibrillation*, Nature, 392 (1998), pp. 75–77.

[31] F. Fenton and A. Karma, *Fiber-rotation-induced vortex turbulence in thick myocardium*, Phys. Rev. Lett., 81 (1998), pp. 481–484.

[32] R. FitzHugh, *Impulses and physiological states in theoretical models of nerve membranes*, Biophys. J., 1 (1961), pp. 445–466.

[33] F. H. Fenton, E. M. Cherry, H. M. Hastings, and S. J. Evans, *Multiple mechanisms of spiral wave breakup in a model of cardiac electrical activity*, Chaos, 12 (2002), pp. 852–892.

[34] F. J. Vetter and A. D. McCulloch, *Three-dimensional analysis of regional cardiac function: A model of rabbit ventricular anatomy*, Prog. Biophys. Mol. Biol., 69 (1998), pp. 157–183.

[35] S. Bertoluzza, *An adaptive collocation method based on interpolating wavelets*, in Multiscale Wavelet Methods for Partial Differential Equations, W. Dahmen, A. J. Kurdila, and P. Oswald, eds., Academic Press, New York, 1997, pp. 109–135.

SIAM J. SCI. COMPUT.
Vol. 28, No. 3, pp. 901–928
© 2006 Society for Industrial and Applied Mathematics

MULTI-ELEMENT GENERALIZED POLYNOMIAL CHAOS FOR ARBITRARY PROBABILITY MEASURES*

XIAOLIANG WAN† AND GEORGE EM KARNIADAKIS†

Abstract. We develop a multi-element generalized polynomial chaos (ME-gPC) method for arbitrary probability measures and apply it to solve ordinary and partial differential equations with stochastic inputs. Given a stochastic input with an arbitrary probability measure, its random space is decomposed into smaller elements. Subsequently, in each element a new random variable with respect to a conditional probability density function (PDF) is defined, and a set of orthogonal polynomials in terms of this random variable is constructed numerically. Then, the generalized polynomial chaos (gPC) method is implemented element-by-element. Numerical experiments show that the cost for the construction of orthogonal polynomials is negligible compared to the total time cost. Efficiency and convergence of ME-gPC are studied numerically by considering some commonly used random variables. ME-gPC provides an efficient and flexible approach to solving differential equations with random inputs, especially for problems related to long-term integration, large perturbation, and stochastic discontinuities.

Key words. uncertainty, polynomial chaos, stochastic differential equations, random inputs

AMS subject classifications. 65C20, 65C30

DOI. 10.1137/050627630

1. Introduction.

(PC) has been used extensively recently to model uncertainty in physical applications. It originated from first defined by Wiener as the span of Hermite polynomial functionals of a Gaussian process [31]. Other names, such as Wiener-chaos and Hermite-chaos, are also used in literature. Ghanem and Spanos first combined Hermite-chaos with a finite element method to model uncertainty in solid mechanics [14, 12, 13]. A more general extension, termed (gPC), was proposed in [32] by Xiu and Karniadakis based on the correspondence between the probability density functions (PDFs) of certain random variables and the weight functions of orthogonal polynomials of the Askey scheme. The family of gPC includes Hermite-chaos as a subset and supplies optimal bases for stochastic processes represented by random variables of commonly used distributions, such as uniform distribution and Beta distribution. In [18, 19], PC was combined with wavelets to deal with discontinuities for uniform random inputs for which standard PC or gPC fails to converge.

To solve differential equations with stochastic inputs following the procedure established by Ghanem and Spanos, the random solution is expanded spectrally by PC, and a Galerkin projection scheme is subsequently used to transform the original stochastic problem into a deterministic one with a larger dimensional parameter [14, 32, 7, 21, 20]. On the other hand, Deb, Babuška, and Oden have proposed employing finite elements in the random space to approximate the stochastic dependence of the solution [4]; this approach also reduces a stochastic differential equation to a high-dimensional deterministic one. This method was later studied theoretically within the framework of a deterministic finite element method in [1]. Since a finite

*Received by the editors March 25, 2005; accepted for publication (in revised form) December 13, 2005; published electronically June 19, 2006. This work was supported by NSF, AFOSR, and ONR.
http://www.siam.org/journals/sisc/28-3/62763.html

†Division of Applied Mathematics, Brown University, Providence, RI 02912 (xlwan@dam.brown.edu, gk@dam.brown.edu).

element method is generally used to solve the obtained deterministic PDE system, the above methods are called [illegible] in [1], while the scheme in [14, 32] is classified as the $p \times h$ version and the scheme in [4] as the $k \times h$ version, where p denotes the polynomial order of PC, k the element size in the random space, and h the element size in the physical space. Both schemes use finite elements in the physical space. The $p \times h$ version relies on the spectral representation in the entire random space by PC, while the $k \times h$ version is based on the discretization of the random space using the same basis as the deterministic finite element method to approximate the random field locally. For simplicity, here we refer to the scheme in [14, 32] as a PC method of p-version and the scheme in [4] as a k-version. Both the concepts and the terminology introduced here share similarities with those of the spectral/[illegible] element method for deterministic problems [17, 25].

In this work, we present a multi-element generalized polynomial chaos (ME-gPC) method, which can deal with stochastic processes represented by random variables of arbitrary distributions and achieve kp-convergence in the random space. In the ME-gPC method, we discretize the random space and use a spectral expansion locally to represent the desired random field. However, the PC basis is, in general, not orthogonal in a random element since the PDF is also discretized simultaneously. The only exception is the uniform distribution which has been considered so far in previous works [14, 4, 32, 28]. To overcome this difficulty, we reconstruct the orthogonal basis numerically in each random element. From the practical standpoint, it is more effective to employ a relatively low polynomial order ($p = 5$ to $p = 8$) for the ME-gPC method. Thus, we can do this reconstruction [illegible] with high accuracy for most continuous or discrete arbitrary probability measures.

In certain nonlinear stochastic differential equations, random inputs can give rise to singularities as discussed in [18, 19] or the Kraichnan–Orszag problem [24], where the gPC method may converge very slowly or even fail to converge. By noting that singularities usually occur at a lower-dimensional manifold in the random space for such problems, we aim to "capture" the singular region by random elements related to a very small probability measure, and implement the gPC method in the regular region. Hence, the error in the singular region will be dominated by the size (k-convergence) of the random elements, while the error in the regular region decreases fast (p-convergence)—possibly exponentially fast. To reduce the total computational cost, we also present a criterion to implement the decomposition of random space adaptively.

2. Multi-element generalized polynomial chaos (ME-gPC) method.

2.1. Overview. The original PC was first proposed by Wiener [31]. It employs the Hermite polynomials in terms of Gaussian random variables as the trial basis to expand stochastic processes in the random space. According to the theorem by Cameron and Martin [3] such expansion converges for any second-order processes in the L_2 sense. gPC was proposed in [33] and employs more types of orthogonal polynomials from the Askey family. It is a generalization of the original Wiener's Hermite-chaos and can deal with non-Gaussian random inputs more efficiently.

Let $(\Omega, \mathcal{F}, P)$ be a probability space, where Ω is the sample space, $\mathcal{F}$ is the σ-algebra of subsets of Ω, and P is a probability measure. Let

$$\boldsymbol{\xi} = (\xi_1, \ldots, \xi_d) : (\Omega, \mathcal{F}) \to (\mathbb{R}^d, \mathcal{B}^d) \tag{2.1}$$

be an $\mathbb{R}^d$-valued continuous random variable, where $d \in \mathbb{N}$ and $\mathcal{B}^d$ is the σ-algebra of Borel subsets of $\mathbb{R}^d$. A general second-order random process $R(\omega) \in {}_2(\Omega, \mathcal{F}, P)$ can

be expressed by gPC as

$$R(\omega) = \sum_{i=0}^{\infty} \hat{a}_i \Phi_i(\boldsymbol{\xi}(\omega)), \tag{2.2}$$

where ω is the random event and $\Phi_i(\boldsymbol{\xi}(\omega))$ denotes the gPC basis of degree p in terms of the random variable $\boldsymbol{\xi}$. The family $\{\Phi_i\}$ is an orthogonal basis in $L_2(\Omega, \mathcal{F}, P)$ with orthogonality relation

$$\mathbb{E}[\Phi_i \Phi_j] = \mathbb{E}[\Phi_i^2]\delta_{ij}, \tag{2.3}$$

where δ_{ij} is the Kronecker delta and $\mathbb{E}$ denotes the expectation with respect to the probability measure $\mathrm{d}P(\omega) = f(\boldsymbol{\xi}(\omega))\mathrm{d}\omega$. The index in (2.2) and $d \in \mathbb{N}$ are, in general, infinite. In practice, both limits will be truncated at a certain level. For example, the dimension d of the stochastic input is usually determined by the Karhunen–Loeve (K-L) expansion based on the decay rate of eigenvalues [14].

For a certain random variable $\boldsymbol{\xi}$, the orthogonal gPC basis $\{\Phi_i\}$ can be chosen in such a way that its weight function has the same form as the PDF $f(\boldsymbol{\xi})$ of $\boldsymbol{\xi}$. The corresponding types of classical orthogonal polynomials $\{\Phi_i\}$ and their associated random variable $\boldsymbol{\xi}$ can be found in [32]. In this paper we generalize the idea of gPC and construct numerically the orthogonal polynomials with respect to an arbitrary probability measure.

2.2. ME-gPC formulation. We next present the scheme of an adaptive ME-gPC method, which consists of the decomposition of random space, the construction of orthogonal polynomials, and an adaptive procedure.

2.2.1. Decomposition of random space. We assume that $\boldsymbol{\xi}$ is a random variable defined on $B = \times_{i=1}^{d}[a_i, b_i]$, where a_i and b_i are finite or infinite in $\mathbb{R}$ and the components of $\boldsymbol{\xi}$ are independent identically-distributed (i.i.d.) random variables. We define a decomposition $\mathbf{D}$ of B as

$$\mathbf{D} = \begin{cases} B_k = [a_{k,1}, b_{k,1}) \times [a_{k,2}, b_{k,2}) \times \cdots \times [a_{k,d}, b_{k,d}], \\ B = \bigcup_{k=1}^{N} B_k, \\ B_{k_1} \bigcap B_{k_2} = \emptyset \text{ if } k_1 \neq k_2, \end{cases} \tag{2.4}$$

where $k, k_1, k_2 = 1, 2, \ldots, N$. Based on the decomposition $\mathbf{D}$, we define the following indicator random variables:

$$I_{B_k} = \begin{cases} 1 & \text{if } \boldsymbol{\xi} \in B_k, \\ 0 & \text{otherwise.} \end{cases} \qquad k = 1, 2, \ldots, N. \tag{2.5}$$

Thus, $\Omega = \cup_{k=1}^{N} I_{B_k}^{-1}(1)$ is a decomposition of the sample space Ω, where

$$I_{B_i}^{-1}(1) \cap I_{B_j}^{-1}(1) = \emptyset \text{ for } i \neq j. \tag{2.6}$$

Given any point $\boldsymbol{q} = (q_1, q_2, \ldots, q_d)$, we use $\boldsymbol{\xi} \leq \boldsymbol{q}$ to denote $\xi_i \leq q_i$ for $i = 1, 2, \ldots, d$. According to the law of total probability, we can obtain

$$\Pr(\boldsymbol{\xi} \leq \boldsymbol{q}) = \sum_{k=1}^{N} \Pr(\boldsymbol{\xi} \leq \boldsymbol{q} | I_{B_k} = 1) \Pr(I_{B_k} = 1). \tag{2.7}$$

Using Bayes's rule, (2.7) implies that we can define a new random variable $\boldsymbol{\xi}_k : I_{B_k}^{-1}(1) \mapsto B_k$ on the probability space $(I_{B_k}^{-1}(1), \mathcal{F} \cap I_{B_k}^{-1}(1), P(\cdot|I_{B_k}=1))$ subject to a conditional PDF

$$\hat{f}_k(\boldsymbol{\xi}_k|I_{B_k}=1) = \frac{f(\boldsymbol{\xi}_k)}{\Pr(I_{B_k}=1)} \tag{2.8}$$

in each random element B_k, where $\Pr(I_{B_k}=1) > 0$.

Let $u(\boldsymbol{x},t;\boldsymbol{\xi}) \in L_2(\Omega,\mathcal{F},P)$ denote a space-time related random field. For simplicity, we may drop $\boldsymbol{x}$ and t.

PROPOSITION 2.1. $\mathcal{P}_M u(\boldsymbol{\xi})$ $u(\boldsymbol{\xi})$ $\{\Phi_i(\boldsymbol{\xi})\}$ M $i=1,2,\ldots,M$ $\mathcal{P}_M u(\boldsymbol{\xi})$ $u(\boldsymbol{\xi})$ L_2 $f(\boldsymbol{\xi})$ $\mathcal{P}_M u(\boldsymbol{\xi}_k)$ $u(\boldsymbol{\xi}_k)$ L_2 $\hat{f}_k(\boldsymbol{\xi}_k|I_{B_k}=1)$ $k=1,2,\ldots,N$

. According to the assumption, we know that

$$\mathbb{E}[(u(\boldsymbol{\xi})-\mathcal{P}_M u(\boldsymbol{\xi}))^2] = \int_B (u(\boldsymbol{\xi})-\mathcal{P}_M u(\boldsymbol{\xi}))^2 f(\boldsymbol{\xi})\mathrm{d}\boldsymbol{\xi} \to 0 \quad \text{as } M \to \infty.$$

By noting

$$\mathbb{E}[(u(\boldsymbol{\xi})-\mathcal{P}_M u(\boldsymbol{\xi}))^2] = \sum_{k=1}^{N} \Pr(I_{B_k}=1) \int_{B_k} (u(\boldsymbol{\xi}_k)-\mathcal{P}_M u(\boldsymbol{\xi}_k))^2 \hat{f}_k(\boldsymbol{\xi}_k|I_{B_k}=1)\mathrm{d}\boldsymbol{\xi}_k, \tag{2.9}$$

we obtain

$$\int_{B_k} (u(\boldsymbol{\xi}_k)-\mathcal{P}_M u(\boldsymbol{\xi}_k))^2 \hat{f}_k(\boldsymbol{\xi}_k|I_{B_k}=1)\mathrm{d}\boldsymbol{\xi}_k \to 0 \quad \text{as } M \to \infty,$$

since both $\Pr(I_{B_k}=1)$ and the integrand on the right-hand side of (2.9) are positive for $k=1,2,\ldots,N$. □

To this end, we know that there exists a local polynomial approximation for each random element B_k, which converges in the L_2 sense with respect to the local conditional PDF. For the orthogonal basis $\{\Phi_i(\boldsymbol{\xi})\}$ on the entire random space, $u(\boldsymbol{\xi}) - \mathcal{P}_M u(\boldsymbol{\xi})$ is orthogonal to the space $V(M,\boldsymbol{\xi}) := \mathrm{span}\{\Phi_i(\boldsymbol{\xi}) : i \le M\}$. We note here that although $\mathcal{P}_M u(\boldsymbol{\xi}_k)$ converges to $u(\boldsymbol{\xi}_k)$ in the L_2 sense within random element B_k, $u(\boldsymbol{\xi}_k) - \mathcal{P}_M u(\boldsymbol{\xi}_k)$ is not orthogonal to the space $V(M,\boldsymbol{\xi}_k)$ with respect to the conditional PDF $\hat{f}_k(\cdot|I_{B_k}=1)$, since the orthogonality $\mathbb{E}[\Phi_i\Phi_j] = \delta_{ij}$ is, in general, valid only on the entire random space of $\boldsymbol{\xi}$ with respect to the PDF $f(\boldsymbol{\xi})$. Due to the efficiency of orthogonal polynomials in representing stochastic processes, we will reconstruct the local polynomial chaos modes numerically to make them mutually orthogonal with respect to the local conditional PDF $\hat{f}(\boldsymbol{\xi}_k|I_{B_k}=1)$. According to [26, theorem 2.1.1], such an orthogonal polynomial system $\{\Phi_{k,i}(\boldsymbol{\xi}_k)\}$ always exists. Since in each random element we perform a spectral expansion as in gPC, we call this method " " (ME-gPC).

To approximate a random field $u(\boldsymbol{\xi})$ using ME-gPC, we expand the random field spectrally in each element B_k, then reconstruct the entire random field by the following proposition.

PROPOSITION 2.2. $\hat{u}_k(\boldsymbol{\xi}_k)$ [illegible] B_k [illegible]

$$u^r(\boldsymbol{\xi}) = \sum_{k=1}^{N} \hat{u}_k(\boldsymbol{\xi}) I_{B_k} = \sum_{k=1}^{N} \sum_{j=0}^{M} \hat{u}_{k,j} \Phi_{k,j}(\boldsymbol{\xi}) I_{B_k}, \tag{2.10}$$

[illegible] $u(\boldsymbol{\xi})$ [illegible] L_2 [illegible]

$$\int_B (u^r(\boldsymbol{\xi}) - u(\boldsymbol{\xi}))^2 f(\boldsymbol{\xi}) \mathrm{d}\boldsymbol{\xi} \to 0 \quad \text{[illegible]} \quad M \to \infty. \tag{2.11}$$

[illegible]. It is easy to see that

$$\begin{aligned} &\int_B (u^r(\boldsymbol{\xi}) - u(\boldsymbol{\xi}))^2 f(\boldsymbol{\xi}) \mathrm{d}\boldsymbol{\xi} \\ &= \sum_{k=1}^{N} \Pr(I_{B_k} = 1) \int_{B_k} (u^r(\boldsymbol{\xi}_k) - u(\boldsymbol{\xi}_k))^2 \hat{f}_k(\boldsymbol{\xi}_k | I_{B_k} = 1) \mathrm{d}\boldsymbol{\xi}_k. \end{aligned}$$

Since $u^r(\boldsymbol{\xi}_k) = \hat{u}_k(\boldsymbol{\xi}_k)$ and the spectral expansion $\hat{u}_k(\boldsymbol{\xi}_k)$ converges locally in the L_2 sense with respect to $\hat{f}(\boldsymbol{\xi}_k | I_{B_k} = 1)$, it is clear that the right-hand side of the above equation goes to zero as $M \to \infty$. The conclusion follows immediately. □

By Bayes's rule and the law of total probability, any statistics can be obtained as

$$\int_B g\left(u(\boldsymbol{\xi})\right) f(\boldsymbol{\xi}) \mathrm{d}\boldsymbol{\xi} \approx \sum_{k=1}^{N} \Pr(I_{B_k} = 1) \int_{B_k} g\left(\hat{u}_k(\boldsymbol{\xi}_k)\right) \hat{f}_k(\boldsymbol{\xi}_k | I_{B_k} = 1) \mathrm{d}\boldsymbol{\xi}_k, \tag{2.12}$$

where $g(\cdot) \in L_1(\Omega, \mathcal{F}, P)$ is a functional of random field $u(\boldsymbol{\xi})$.

In the appendix, we demonstrate for the interested reader some details on how to solve PDEs with stochastic coefficients using ME-gPC.

2.2.2. Construction of orthogonal polynomials. We next discuss the numerical construction of orthogonal polynomials with respect to a conditional PDF $\hat{f}_k(\cdot | I_{B_k} = 1)$. For simplicity, here we discuss only the construction of one-dimensional orthogonal polynomials, since the high-dimensional basis can be obtained using tensor products of one-dimensional basis.

It is a distinctive feature of orthogonal polynomials, compared to other orthogonal systems, that they satisfy a three-term recurrence relation,

$$\begin{aligned} &\pi_{i+1}(\tau) = (\tau - \alpha_i)\pi_i(\tau) - \beta_i \pi_{i-1}(\tau), \quad i = 0, 1, \ldots, \\ &\pi_0(\tau) = 1, \quad \pi_{-1}(\tau) = 0, \end{aligned} \tag{2.13}$$

where $\{\pi_i(\tau)\}$ is a set of (monic) orthogonal polynomials,

$$\pi_i(\tau) = \tau^i + \text{lower-degree terms}, \quad i = 0, 1, \ldots, \tag{2.14}$$

and the coefficients α_i and β_i are uniquely determined by a positive measure, which corresponds to a probability measure in the construction we propose herein.

For a continuous measure $m(\tau)$ there are two classical methods to compute the recurrence coefficients α_i and β_i: the Stieltjes procedure and the modified Chebyshev

algorithm [8]. The Stieltjes procedure uses the fact that the coefficients α_i and β_i can be expressed by the following simple formulas:

$$\alpha_i = \frac{(\tau\pi_i, \pi_i)}{(\pi_i, \pi_i)}, \quad i = 0, 1, 2, \dots \tag{2.15}$$

and

$$\beta_0 = (\pi_0, \pi_0), \quad \beta_i = \frac{(\pi_i, \pi_i)}{(\pi_{i-1}, \pi_{i-1})}, \quad i = 1, 2, \dots, \tag{2.16}$$

where $(\cdot, \cdot)$ denotes the inner product in terms of the measure $m(\tau)$. The above two formulas together with the recurrence relation (2.13) can be used to calculate recursively as many coefficients α_i and β_i as desired.

The modified Chebyshev algorithm is a generalization of the Chebyshev algorithm [8]. The Chebyshev algorithm relies on the fact that the first n pairs of recurrence coefficients α_i and β_i, $i = 0, 1, \dots, n-1$, can be uniquely determined by the first $2n$ moments μ_i:

$$\mu_i = \int_B \tau^i \mathrm{d}m(\tau), \quad i = 0, 1, \dots, 2n-1. \tag{2.17}$$

Analytical formulas are known which express α_i and β_i in terms of Hankel determinants in these moments. However, this algorithm is not reliable for a big n due to the increasing sensitivity of these formulas to small errors. The modified Chebyshev algorithm replaces the power τ^i with a properly chosen polynomial $h_i(\tau)$ of degree i:

$$\nu_i = \int_B h_i(\tau) \mathrm{d}m(\tau), \quad i = 0, 1, \dots, 2n-1. \tag{2.18}$$

Generally, we can assume that $h_i(\tau)$ are monic orthogonal polynomials satisfying a three-term relation

$$\begin{aligned} h_{i+1}(\tau) &= (\tau - \hat{\alpha}_i) h_i(\tau) - \hat{\beta}_i h_{i-1}(\tau), \quad i = 0, 1, \dots, \\ h_0(\tau) &= 1, \quad h_{-1}(\tau) = 0. \end{aligned} \tag{2.19}$$

Using the $2n$ modified moments in (2.18) and the $2n-1$ pairs of recurrence coefficients $\hat{\alpha}_i$ and $\hat{\beta}_i$, $i = 0, 1, \dots, 2n-2$, in (2.19), the first n desired pairs of recurrence coefficients α_i and β_i, $i = 0, 1, \dots, n-1$, can be generated [8].

For a discrete measure

$$\mathrm{d}m_M(\tau) = \sum_{i=1}^{M} w_i \delta(\tau - \tau_i) \mathrm{d}\tau, \quad i = 0, 1, \dots, M \tag{2.20}$$

with δ being the Dirac delta function, we have another choice: the Lanczos algorithm [2, 8]. Given (2.20), there exists an orthogonal matrix $Q^{(M+1)\times(M+1)}$ with the first column being $[1, 0, \dots, 0]^T \in \mathbb{R}^{(M+1)\times 1}$ such that

$$Q^T A_M Q = J_M, \tag{2.21}$$

where

$$A_M = \begin{bmatrix} 1 & \sqrt{w_1} & \sqrt{w_2} & \cdots & \sqrt{w_M} \\ \sqrt{w_1} & \tau_1 & 0 & \cdots & 0 \\ \sqrt{w_2} & 0 & \tau_2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \sqrt{w_M} & 0 & 0 & \cdots & \tau_M \end{bmatrix}, \tag{2.22}$$

and J_M is the Jacobian matrix

$$J_M = \begin{bmatrix} 1 & \sqrt{\beta_0} & 0 & \cdots & 0 \\ \sqrt{\beta_0} & \alpha_0 & \sqrt{\beta_1} & \cdots & 0 \\ 0 & \sqrt{\beta_1} & \alpha_1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \alpha_{M-1} \end{bmatrix}. \tag{2.23}$$

The above algorithms were presented in [8], and a Fortran package based on these algorithms was proposed in [9]. To implement the Stieltjes procedure and the modified Chebyshev algorithm we need to evaluate the inner product with high precision. In practice, we usually employ a Gauss-type quadrature rule. Such a quadrature rule can be regarded as a discrete measure that yields the corresponding discrete versions of these two methods [8]. The stability of the Stieltjes procedure is not completely clear. One interesting aspect about this procedure is that it is reliable in many cases where a discretization for the inner product of a continuous measure is employed, while it breaks down in some cases where the measure itself is discrete. In [6], the Stieltjes procedure for discrete measures is discussed and improved. The stability of the modified Chebyshev algorithm is determined by the condition of the map from the $2n$ modified moments to the n pairs of recurrence coefficients. Such a condition was studied in [8, 5]. The Lanczos algorithm has good stability properties but it may be considerably slower than the Stieltjes procedure. In this work, we use the Stieltjes procedure and the Lanczos algorithm.

It can be seen that the basic operation in the above procedures is the inner product with respect to a given measure $m(\tau)$. Since an explicit formula is generally unavailable, we discretize the inner product using an appropriate quadrature rule. We know that an n-point Gauss-type quadrature rule can reach the highest algebraic accuracy of order $2n-1$. However, such quadrature points and the corresponding weights are not in general explicitly known. In this work, we use the interpolatory quadrature rule [23] relative to the Legendre weight function on $[-1,1]$. An n-point interpolatory quadrature rule has an algebraic precision $n-1$. Compared to the Gauss–Legendre quadrature rule, the advantage of the interpolatory quadrature rule is that the nodes and weights are explicitly known; thus the expensive Newton–Raphson iterations are avoided.

Using the Stieltjes procedure or the Lanczos algorithm, the recurrence coefficients α_i and β_i are computed iteratively using the following stopping criterion [8]:

$$|\beta_i^s - \beta_i^{s-1}| \le \epsilon \beta_i^s, \quad i = 0, 1, \ldots, n-1, \tag{2.24}$$

where s is the iteration step and ϵ is the relative error.

If a measure $m(\tau)$ is given and orthogonal polynomials up to pth order are needed, we first compute the recurrence coefficients α_i and β_i, $i = 0, 1, \ldots, p-1$. Once the recurrence coefficients are obtained, the orthogonal polynomials are uniquely determined. From the recurrence coefficients, the Gauss-type quadrature points and the corresponding integration weights can be efficiently derived [9].

It is well known that the gPC method usually relies on a three-dimensional table $\mathbb{E}[\pi_l \pi_m \pi_n]$ for the Galerkin projection [14]. Here we use the notation $\mathbb{E}[\cdot]$ freely to denote the ensemble average with respect to a given measure $m(\tau)$ since $m(\tau)$ can be regarded as a probability measure in this work. If the recurrence coefficients are obtained—in other words, the orthogonal polynomials are uniquely determined—we

can use any numerical integration formula to calculate the table $\mathbb{E}[\pi_l \pi_m \pi_n]$. Here we consider the quadrature rule. We know that for the measure $m(\tau)$ there exists, for each $M \in \mathbb{N}$, a quadrature rule

$$\int_B f(\tau) \mathrm{d}\sigma(\tau) = \sum_{i=1}^{M} w_i f(\tau_i) + R_M(f), \tag{2.25}$$

where $R_M(f) = 0$ if f is a polynomial of degree $\leq 2M - c$. The value of c is determined by the type of quadrature used, which can be either classical Gauss ($c = 1$), Gauss–Radau ($c = 2$), or Gauss–Lobatto ($c = 3$). If we do not consider the numerical accuracy of the recurrence coefficients, we need

$$M = \left\lceil \frac{3p + c}{2} \right\rceil \tag{2.26}$$

quadrature points to get the three-term integration $\mathbb{E}[\pi_l \pi_m \pi_n]$ with zero error, where $\lceil * \rceil$ denotes the smallest integer no less than $*$. Thus, we need to compute all the recurrence coefficients α_i and β_i with $i \leq M - 1$ although we employ polynomials only up to order p.

2.2.3. An adaptive procedure. Adaptivity is necessary for many cases, such as problems related to long-term integration or discontinuity. In this work we present a heuristic adaptivity criterion based on the relative local errors similar to the adaptivity criteria employed in the spectral element method for deterministic problems [22, 15].

We assume that the gPC expansion of a random field in element k is

$$\hat{u}_k(\boldsymbol{\xi}_k) = \sum_{j=0}^{N_p} \hat{u}_{k,j} \Phi_{k,j}(\boldsymbol{\xi}_k), \tag{2.27}$$

where p is the highest order of PC and N_p denotes the total number of basis modes given by

$$N_p = \frac{(p + d)!}{p! d!} - 1, \tag{2.28}$$

where d is the dimension of $\boldsymbol{\xi}_k$. From the orthogonality of gPC we can easily obtain the local variance given by PC with order p:

$$\sigma_{k,p}^2 = \sum_{j=1}^{N_p} \hat{u}_{k,j}^2 \mathbb{E}[\Phi_{k,j}^2]. \tag{2.29}$$

The [illegible] global mean $\bar{u}$ and variance $\bar{\sigma}^2$ can be expressed as

$$\bar{u} = \sum_{k=1}^{N} \hat{u}_{k,0} \Pr(I_{B_k} = 1), \quad \bar{\sigma}^2 = \sum_{k=1}^{N} \left[\sigma_{k,p}^2 + (\hat{u}_{k,0} - \bar{u})^2 \right] \Pr(I_{B_k} = 1). \tag{2.30}$$

Compared to the error of variance, the error of mean is usually much smaller in the gPC method. Thus, we can express the [illegible] global variance as

$$\sigma^2 \approx \bar{\sigma}^2 + \sum_{k=1}^{N} \gamma_k \Pr(I_{B_k} = 1), \tag{2.31}$$

where γ_k denotes the error of local variance. We define the decay rate of relative error of PC in each element as follows:

$$\eta_k = \frac{\sum_{i=N_{p-1}+1}^{N_p} \hat{u}_{k,i}^2 \mathbb{E}[\Phi_{k,i}^2]}{\sigma_{k,p}^2}. \tag{2.32}$$

Based on η_k and the scaled parameter $\Pr(I_{B_k} = 1)$, we implement k-[illegible], in other words, decompose the current random element into smaller ones, if the criterion

$$\eta_k^\gamma \Pr(I_{B_k} = 1) \geq \theta_1, \quad 0 < \gamma < 1, \tag{2.33}$$

is satisfied, where γ and θ_1 are prescribed constants. A discussion about γ and θ_1 can be found in [28] for random inputs with [illegible] distributions.

The error of the variance should decrease exponentially (for continuous random inputs) given sufficient smoothness of the solutions; furthermore, the error of $\hat{u}_{k,0}$ is usually much smaller than the error of local variance. Thus, we can use η_k as an indicator of the true error for any orthogonal chaos basis, and the criterion (2.33) is a general one. For high-dimensional random inputs, we can choose the most sensitive random dimensions for refinement. We define the sensitivity for each random dimension as

$$r_i = \frac{(\hat{u}_{i,p})^2 \, \mathbb{E}[\Phi_{i,p}^2]}{\sum_{j=N_{p-1}+1}^{N_p} \hat{u}_j^2 \mathbb{E}[\Phi_j^2]}, \qquad i = 1, 2, \ldots, d, \tag{2.34}$$

where we drop the subscript k for clarity and the subscript $*_{i,p}$ denotes the one-dimensional mode of ξ_i with polynomial order p. Let us consider a d-dimensional mode $\phi_{(j_1,j_2,\ldots,j_d)}(\boldsymbol{\xi}) = \phi_{j_1}(\xi_1)\phi_{j_2}(\xi_2)\cdots\phi_{j_d}(\xi_d)$ with $\sum_{n=1}^d j_n \leq p$, which is a tensor product of one-dimensional modes $\phi_{j_n}(\xi_n)$, $n = 1, 2, \ldots, d$. The subscript $*_{i,p}$ means that $\phi_{i,p}(\boldsymbol{\xi}) = \phi_p(\xi_i)$, $i = 1, 2, \ldots, d$. All random dimensions which satisfy

$$r_i \geq \theta_2 \cdot \max_{j=1,\ldots,d} r_j, \quad 0 < \theta_2 < 1, \quad i = 1, 2, \ldots, d \tag{2.35}$$

will be split into two [illegible] random elements in the next time step while all other random dimensions will remain unchanged. Hence, we can reduce the total element number while gaining efficiency.

2.2.4. Implementation. Next we discuss issues related to the numerical implementation. Based on the decomposition of random space, it is straightforward to reconstruct orthogonal polynomials with respect to the conditional PDFs of the local random variables $\boldsymbol{\xi}_k$. However, such a procedure is not robust in practice because (π_i, π_i) may decay fast and cause a potential problem of overflow or underflow in the computer.

PROPOSITION 2.3. [illegible] $[a, b]$ [illegible] a [illegible] b [illegible]

$$(\pi_{i+1}, \pi_{i+1}) \leq (b - a)^2 (\pi_i, \pi_i). \tag{2.36}$$

[illegible]. From (2.15), we can obtain that $a \leq \alpha_i \leq b$. Using (2.13) and the Cauchy–Schwarz inequality, we know that

$$(\pi_{i+1}, \pi_{i+1}) = ((\tau - \alpha_i)\pi_i, \pi_{i+1}) \leq ((\tau - \alpha_i)\pi_i, (\tau - \alpha_i)\pi_i)^{1/2} (\pi_{i+1}, \pi_{i+1})^{1/2}, \tag{2.37}$$

which yields that

$$(\pi_{i+1}, \pi_{i+1}) \leq ((\tau - \alpha_i)\pi_i, (\tau - \alpha_i)\pi_i). \tag{2.38}$$

The proof concludes by noting that $a \leq \alpha_i \leq b$. □

Thus, if the element is small, the L_2 norm of π_i will lead to underflow quickly. In practice, we rescale the random elements with [illegible] boundaries by the following linear transformation:

$$\xi_{k,i} = \frac{b_{k,i} - a_{k,i}}{2} Y_{k,i} + \frac{b_{k,i} + a_{k,i}}{2}, \tag{2.39}$$

where we map the random variable $\boldsymbol{\xi}_k$ defined in element k to a random variable $\boldsymbol{Y}_k$ defined in $[-1, 1]^d$. The PDF of $\boldsymbol{Y}_k$ can be obtained as

$$\bar{f}(\boldsymbol{y}_k) = \det \left| \frac{\partial \boldsymbol{\xi}_k}{\partial \boldsymbol{y}_k} \right| \hat{f}(\boldsymbol{\xi}_k(\boldsymbol{y}_k)) = \frac{f(\boldsymbol{\xi}_k(\boldsymbol{y}_k))}{\Pr(I_{B_k} = 1)} \prod_{i=1}^{d} \frac{b_{k,i} - a_{k,i}}{2}. \tag{2.40}$$

Compared to $\boldsymbol{\xi}_k$, $\boldsymbol{Y}_k$ are much more tractable for the numerical construction of orthogonal polynomials as demonstrated in section 3.1.1. After such a transformation is employed, we can apply the ME-gPC scheme with respect to the new random variable $\boldsymbol{Y}_k$ instead of the random variable $\boldsymbol{\xi}_k$. Note that such a mapping is usually unnecessary for random elements with at least one [illegible] boundary because these elements can be related to a small probability $\Pr(I_{B_k} = 1)$.

We now summarize the overall ME-gPC algorithm.

ALGORITHM 2.1 (ME-gPC).

1. [illegible]
2. [illegible]
   i [illegible] 1 [illegible] M
   [illegible]
   [illegible] $\eta_k^{\gamma} \Pr(I_{B_k} = 1) \geq \theta_1$ [illegible] k [illegible]
   [illegible] $r_n \geq \theta_2 \cdot \max_{j=1,\ldots,d} r_j$ [illegible]
   [illegible] [illegible] ξ_n [illegible] [illegible] $\xi_{n,1}$ $\xi_{n,2}$
   [illegible] $\xi_{n,1}$ $\xi_{n,2}$ $Y_{n,1}$ $Y_{n,2}$ [illegible] $[-1, 1]$
   [illegible] [illegible] [illegible] [illegible] [illegible] $Y_{n,1}$ $Y_{n,2}$
   [illegible]
   [illegible] d [illegible]
   [illegible]
   [illegible]
   [illegible]
   [illegible]
3. [illegible]

2.1 (C^0 continuity in random space). In the ME-gPC scheme, we deal with the stochastic problem locally without considering the interface between two random elements; in other words, we split the original problem into N independent ones. The reason is that all the statistics are in L_q sense, which means that the statistics are determined by up to qth moments of the desired random field. Thus, the continuity

$$u_{B_1}(\boldsymbol{\xi}) = u_{B_2}(\boldsymbol{\xi}), \quad \boldsymbol{\xi} \in \bar{B}_1 \cap \bar{B}_2, \tag{2.41}$$

where $\bar{B}_1$ and $\bar{B}_2$ indicate the closure of two adjacent random elements, respectively, is not required since the measure of the interface is zero.

Remark 2.2 (construction cost of π_i). In practice, the K-L decomposition related to a random variable $\boldsymbol{\xi}$ is often used to reduce the dimensionality of colored noise. Since all the components of $\boldsymbol{\xi}$ are assumed to have identical distributions, we can maintain a table for one-dimensional orthogonal polynomials which have already been constructed. When the decomposition of random space is activated, we first look up the table to check if the edges of new random elements are in the table or not and construct orthogonal polynomials only for the missing edges. This way, the time cost for construction of orthogonal polynomials is almost independent of the random dimension.

Remark 2.3 (complexity of ME-gPC). In Algorithm 2.1, we decompose a stochastic problem into N independent ones, since C^0 continuity is unnecessary in random space. Thus, if polynomials of the same order are used in both gPC and ME-gPC, the cost of ME-gPC will increase linearly by a factor N compared to the cost of gPC, since the cost of numerical orthogonal polynomials is negligible as demonstrated in section 3.1.4.

3. Numerical results.

3.1. Construction of orthogonal polynomials. We first discuss the construction of orthogonal polynomials for random variables with some commonly used distributions. Due to the nice properties of uniform distribution, the orthogonality of Legendre-chaos can be naturally inherited in the decomposition of random space, which means that the polynomial construction is unnecessary for the uniform distribution. Here we focus on the Beta and Gaussian distributions, for which we need the on-the-fly polynomial construction.

3.1.1. Problem of overflow and underflow. We consider a random variable defined in $[-1, 1]$ with the following Beta distribution:

$$f(x) = \frac{(1-x)^\alpha (1+x)^\beta}{2^{\alpha+\beta+1} B(\alpha+1, \beta+1)}, \quad -1 \le x \le 1. \tag{3.1}$$

In a random element $[0, 0.001]$, we define a random variable X with a conditional PDF

$$\hat{f}(x) = \frac{f(x)}{\int_0^{0.001} f(x)\mathrm{d}x}, \tag{3.2}$$

and map X to another random variable Y defined in $[-1, 1]$ with a PDF

$$\bar{f}(y) = \frac{0.0005 f(x(y))}{\int_0^{0.001} f(x)\mathrm{d}x}, \tag{3.3}$$

where

$$x(y) = 0.0005y + 0.0005. \tag{3.4}$$

Here we let $\alpha = 1$ and $\beta = 4$. In Figure 3.1, we show the orthogonal polynomials for random variable X from first order up to fourth order. We can see that the maximum values of these polynomials decrease very quickly approaching the machine accuracy. In Figure 3.2, the corresponding orthogonal polynomials for random variable Y are shown. It can be seen that their shapes are the same as those of the orthogonal polynomials for X but are well scaled.

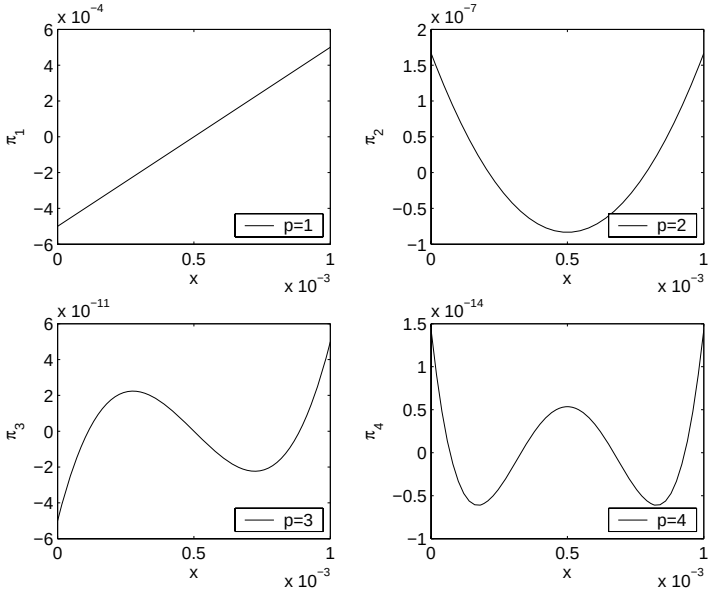


FIG. 3.1. *Orthogonal polynomials for random variable X, π_1 to π_4.*

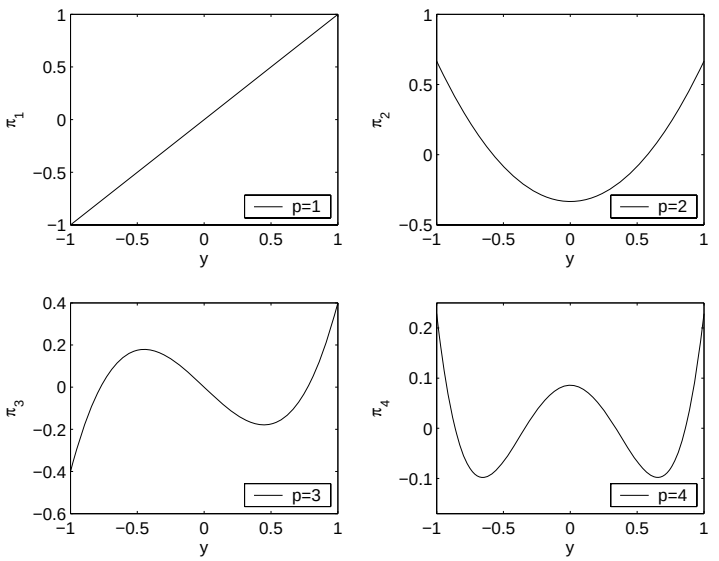


FIG. 3.2. *Orthogonal polynomials for random variable Y, π_1 to π_4.*

3.1.2. Orthogonal polynomials for Beta distribution. From the ME-gPC scheme, we know that the orthogonal basis in each random element depends on a particular part of the PDF of the random inputs. We now demonstrate such a dependence using a Beta-type random variable X with $\alpha = 1$ and $\beta = 4$. For simplicity, we consider only two random elements: $[-1, 0]$ and $[0, 1]$. We define a random variable

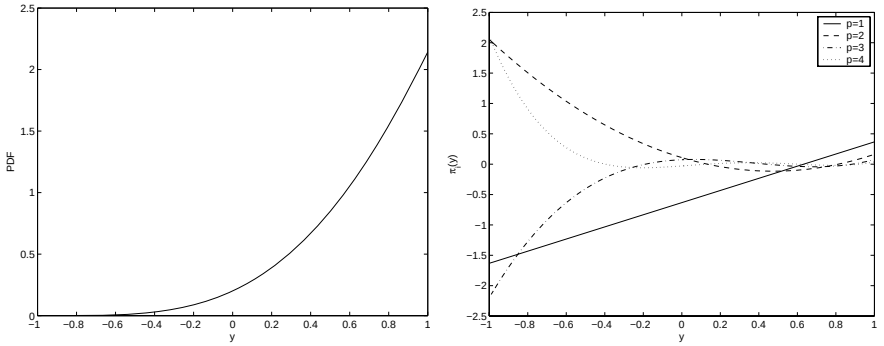


FIG. 3.3. *Left: PDF of random variable Y_l induced by Beta distribution with $\alpha = 1$ and $\beta = 4$. Right: orthogonal polynomials for Y_l.*

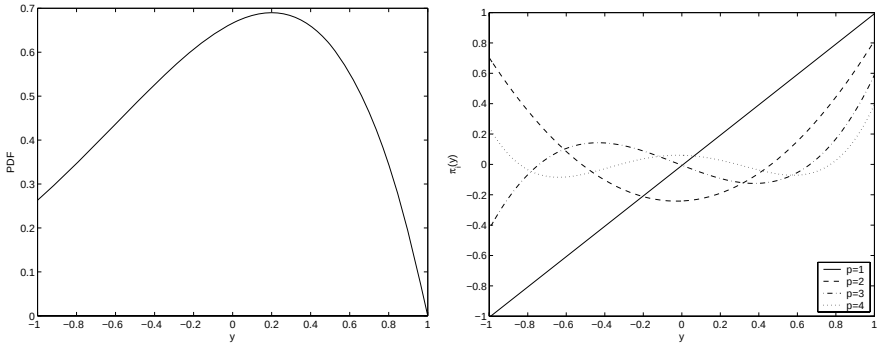


FIG. 3.4. *Left: PDF of random variable Y_r induced by Beta distribution with $\alpha = 1$ and $\beta = 4$. Right: orthogonal polynomials for Y_r.*

X_l in $[-1, 0]$ with a conditional PDF

$$f_l(x_l) = \frac{f(x_l)}{\int_{-1}^{0} f(x)\mathrm{d}x} \tag{3.5}$$

and another random variable X_r in $[0, 1]$ with a conditional PDF

$$f_r(x_r) = \frac{f(x_r)}{\int_{0}^{1} f(x)\mathrm{d}x}, \tag{3.6}$$

where $f(x)$ is the PDF of X. Due to the potential problem of overflow (underflow), we will not construct the orthogonal basis according to the above two conditional PDFs, but first map X_l and X_r to Y_l and Y_r, respectively, using the transformation

$$X_l = \frac{1}{2}Y_l - \frac{1}{2}, \quad Y_r = \frac{1}{2}Y_r + \frac{1}{2}, \tag{3.7}$$

and then construct orthogonal polynomials in terms of Y_l and Y_r. In Figures 3.3 and 3.4, we show the PDFs of Y_l and Y_r on the left and the corresponding orthogonal polynomials on the right. It is clear that these two sets of orthogonal polynomials are quite different due to their different weight functions (i.e., local PDFs).

3.1.3. Orthogonal polynomials for Gaussian distribution. Some random distributions, e.g., Gaussian distribution and Gamma distribution, have long tails. Next, we demonstrate the decomposition of random space and the corresponding orthogonal polynomials for this type of random variable. Given a Gaussian random variable $X \sim N(0,1)$, we decompose the random space into three random elements $(-\infty, -a]$, $[-a, a]$, and $[a, +\infty)$, where a is a positive constant. In the middle element $[-a, a]$, we define a random variable X_m with a conditional PDF

$$f_m(x_m) = \frac{e^{-\frac{1}{2}x_m^2}}{\int_{-a}^{a} e^{-\frac{1}{2}x^2}\mathrm{d}x}. \tag{3.8}$$

In one of the tail elements, say, $[a, +\infty)$, we define a random variable X_∞ with a conditional PDF

$$f_\infty(x_\infty) = \frac{e^{-\frac{1}{2}x_\infty^2}}{\int_{a}^{\infty} e^{-\frac{1}{2}x^2}\mathrm{d}x}. \tag{3.9}$$

We choose a in such a way that $\Pr(X \geq a) < \epsilon$, where ϵ is a small positive constant. From (2.31), we know that the error contribution of tail elements can be effectively weakened by the small value of $\Pr(X \geq a)$; thus we usually do not decompose the tail elements in applications, but rather the middle element $[-a, a]$. Furthermore, there is no overflow or underflow problem for the tail elements due to the infinite boundary. Here we take $a = 5$, which yields $\Pr(X \geq 5) = 2.87 \times 10^{-7}$. In Figure 3.5, we show the PDF of the rescaled random variable $Y_m = X_m/a$ on the left and the corresponding orthogonal polynomial basis on the right. The orthogonal polynomials have similar shapes with Hermite polynomials; however, they are bounded now. In Figure 3.6, the conditional PDF of X_∞ is shown on the left and the corresponding orthogonal basis on the right. Due to the infinite boundary the orthogonal polynomials look almost like Laguerre polynomials.

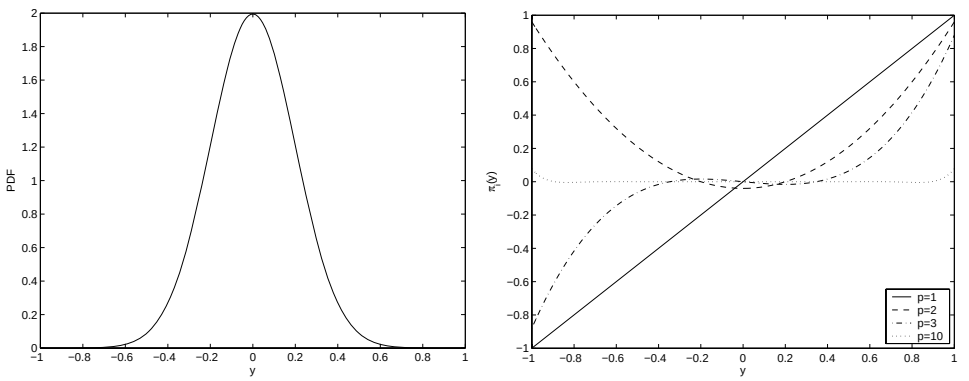


FIG. 3.5. *Left: PDF of random variable Y_m induced by Gaussian distribution. Right: orthogonal polynomials for Y_m.*

3.1.4. Efficiency of the solver for numerical orthogonal polynomials. It is clear that the ME-gPC method relies heavily on the numerical orthogonal polynomials. We now test the solver for the recurrence coefficients, quadrature points, and integration weights. Due to the absence of explicit formulas, we use the following

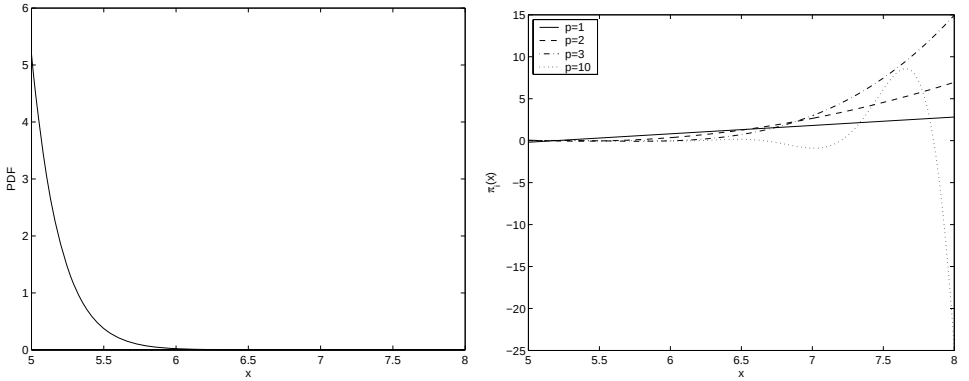


FIG. 3.6. *Left: PDF of random variable X_∞ induced by Gaussian distribution. Right: orthogonal polynomials for X_∞.*

example. Given a weight function $w(x)$ defined in $[a, b]$, we decompose $[a, b]$ into small elements and construct numerical orthogonal polynomials in these elements. Then we map a simple function $g(x) = c_1 x^p + c_2$ onto such elements numerically and compute the errors on the quadrature points in each element. In the kth element $[a_k, b_k]$, $g(x) = c_1 x^p + c_2$ should have the form

$$g_k(\tau) = c_1 \left(\frac{b_k - a_k}{2} \tau + \frac{b_k + a_k}{2} \right)^p + c_2, \tag{3.10}$$

since all the orthogonal polynomials in element k are defined in $[-1, 1]$. Expressed by the numerical orthogonal polynomials, $g_k(\tau)$ has another form:

$$\hat{g}_k(\tau) = \sum_{i=0}^{p} \nu_{k,i} \pi_{k,i}(\tau), \tag{3.11}$$

where the coefficients $\nu_{k,i}$ can be obtained by the inner product

$$\nu_{k,i} = (g_k(\tau), \pi_{k,i})/(\pi_{k,i}, \pi_{k,i}). \tag{3.12}$$

Next, we define the L_∞ error on the quadrature points over all the elements

$$\epsilon_{max} = \max \left| \frac{g_k(\tau_j) - \hat{g}_k(\tau_j)}{g_k(\tau_j)} \right|, \quad k = 1, 2, \ldots, N, \quad j = 1, 2, \ldots, n, \tag{3.13}$$

where N is the number of elements and n is the number of quadrature points in each element. In Table 3.1, we show ϵ_{max} and the time cost for different uniform meshes. We record the time used for the computation of recurrence coefficients, quadrature points, integration weights, and the table $\mathbb{E}[\pi_{k,i}\pi_{k,j}\pi_{k,m}]$, which represents the total set-up requirement for the standard Galerkin gPC method. We can see that the biggest error ϵ_{max} occurs in a one-element mesh. As the number of elements increases, ϵ_{max} is almost of the same order as the relative error of recurrence coefficients, which implies that the numerical polynomials can achieve good orthogonality and high accuracy. Furthermore, numerical experiments show that the time cost for the numerical polynomials is very small. For example, 10^5 sets of orthogonal polynomials up to 10th order can be computed within less than one-and-a-half minutes on a 1.5GHz

TABLE 3.1

Error ϵ_{max} and time cost for a uniform mesh with N elements. $p = 10$ and $g(x) = x^{10} + 1$. The relative error for the recurrence coefficients was set to be 10^{-13}. $\alpha = 1$ and $\beta = 4$ for Beta distribution. The middle element $[-6, 6]$ was used for Gaussian distribution. The computations were performed on a 1.5GHz AMD CPU.

	Beta distribution		Gaussian distribution	
N	Error	Time (sec)	Error	Time (sec)
1	7.98e-11	<0.01	1.50e-10	<0.01
2	2.58e-12	<0.01	4.94e-11	<0.01
4	2.38e-13	<0.01	1.53e-11	<0.01
6	4.58e-13	<0.01	4.66e-13	<0.01
8	3.53e-13	<0.01	3.60e-13	<0.01
10	7.45e-13	0.01	1.10e-12	0.01
10^2	9.45e-13	0.08	8.28e-13	0.08
10^3	1.00e-12	0.79	9.37e-13	0.84
10^4	3.69e-12	7.92	4.51e-13	7.87
10^5	9.28e-13	76.43	4.09e-13	78.74

AMD CPU. In practice, we do not need so many elements and the time cost for the construction of orthogonal polynomials actually can be ignored compared to the time used by the gPC solver.

3.2. Accuracy and convergence of ME-gPC. We next show the accuracy and convergence of ME-gPC by several numerical examples. We use a simple stochastic ODE model to demonstrate the *kp*-convergence of the ME-gPC method and the Kraichnan–Orszag problem to show how to deal with discontinuities introduced by random inputs using the ME-gPC method.

3.2.1. Basic stochastic ODE model. We consider the performance of the ME-gPC method for the following simple ODE equation studied with the original gPC in [32]:

$$\frac{\mathrm{d}u}{\mathrm{d}t} = -\lambda h(\zeta(\omega))u, \qquad u(0;\omega) = u_0, \tag{3.14}$$

where $\zeta(\omega)$ is a random variable, h is a function of $\zeta(\omega)$, and λ is constant. The exact solution can be easily found as

$$u(t;\omega) = u_0 e^{-\lambda h(\zeta(\omega))t}. \tag{3.15}$$

Using ME-gPC with uniform meshes, we study the following two cases:

(i) ζ is a Beta-type random variable. $\alpha = 1$, $\beta = 4$, and $h(\zeta(\omega)) = \zeta(\omega)$.

(ii) ζ is a Gaussian random variable. $h(\zeta(\omega)) = \zeta(\omega)$.

In Figure 3.7, we show the convergence of ME-gPC at $t = 5$ for case (i) with p-convergence on the left and k-convergence on the right. It can be seen that ME-gPC exhibits exponential convergence while the index of algebraic convergence goes asymptotically to a constant. The convergence of the ME-gPC method at $t = 5$ for case (ii) is shown in Figure 3.8. Note here for the Gaussian random variable we start with a three-element mesh: $(-\infty, -6]$, $[-6, 6]$, and $[6, +\infty)$. We refine only the middle element $[-6, 6]$ because the error contribution of the tail elements is very small compared with that of the middle element. We can see that the error curve of such a three-element mesh matches exactly with that given by the Hermite-chaos. The plot of p-convergence also shows that the four-element mesh works better than

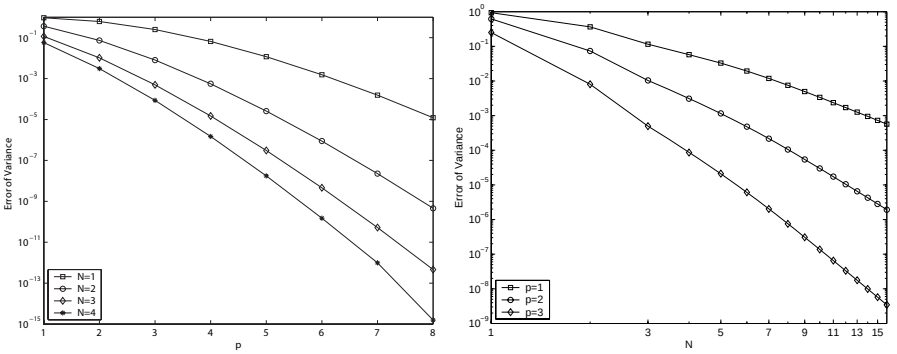


FIG. 3.7. *Convergence of ME-gPC for the simple ODE with ζ being a Beta-type random variable. $\lambda = 1$. Left: p-convergence. Right: k-convergence.*

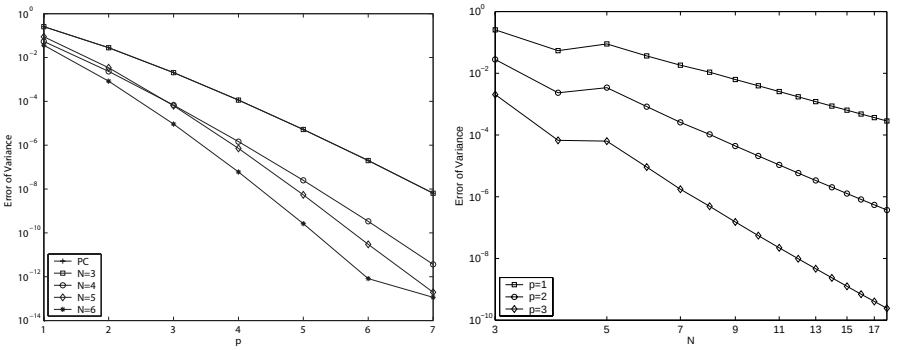


FIG. 3.8. *Convergence of ME-gPC for the simple ODE with ζ being a Gaussian random variable. $\lambda = 0.1$. Left: p-convergence. Right: k-convergence.*

the five-element mesh when $p < 3$. The behavior of (3.14) may be very different if $\zeta(\omega)$ is located in different random elements, which gives rise to a nonuniform error distribution in each element. For instance, when $p = 2$, $\zeta(\omega) < 0$ in element $[-6, 0]$ and the error of local variance is $\epsilon_{[-6,0]} = 5.06 \times 10^{-3}$; $\zeta(\omega) > 0$ in element $[0, 6]$ and $\epsilon_{[0,6]} = 2.48 \times 10^{-3}$.

From (2.31), we know that the error of global variance is of $O((\epsilon_{[-6,0]} + \epsilon_{[0,6]})/2)$, where we neglect the error contribution of the tail elements. For a five-element mesh the error contribution comes mainly from element $[-2, 2]$ in which $\Pr(I_{[-2,2]} = 1) = 0.95$. Thus, the error of global variance is of $O(\epsilon_{[-2,2]})$, where $\epsilon_{[-2,2]} = 5.39 \times 10^{-3}$. It can be seen that $\epsilon_{[-2,2]} > (\epsilon_{[-6,0]} + \epsilon_{[0,6]})/2$, which corresponds to the behavior of p-convergence and k-convergence in Figure 3.8. As the polynomial order p or the number N of the random elements increases, p-convergence or k-convergence will be dominant and such a phenomenon will disappear. In Table 3.2, we present the estimated indices of algebraic convergence for both case (i) and case (ii). It can be seen that for both mean and variance the index of algebraic convergence is close to $-2(p + 1)$, which is consistent with the error estimate given in [4]. From the theory of the deterministic finite element method [4], it can be shown that the index of algebraic convergence will go to $-2(p+1)$ for any PDF. When computing the recurrence coefficients α_i and β_i iteratively, a relative error 10^{-13} is imposed for both case (i) and case (ii). From the plots of p-convergence, it can be seen that the maximum accuracy ME-gPC can

TABLE 3.2
The indices of algebraic convergence of the basic stochastic ODE model. The algebraic indices are computed by the errors given by $N = 14$ and $N = 15$. $t = 5$.

	Beta distribution (case (i))		Gaussian distribution (case (ii))	
	Mean	Variance	Mean	Variance
$p = 1$	-3.95	-3.84	-3.90	-3.88
$p = 2$	-5.96	-5.87	-5.91	-5.90
$p = 3$	-7.96	-7.88	-7.59	-7.90

TABLE 3.3
A comparison between gPC, ME-gPC, and the standard Monte Carlo method for a fixed cost denoted by the number of floating point operations n. Beta distribution (case (i)*) is considered and normalized errors are used. The first row with $N = 1$ corresponds to the gPC results.*

	$\lambda = 1.0$		$\lambda = 1.5$	
$n = 200$	Error (mean)	Error (var)	Error (mean)	Error (var)
$N = 1$, $p = 9$	4.27e-12	7.84e-7	6.67e-9	1.95e-4
$N = 4$, $p = 4$	2.84e-9	1.49e-6	1.20e-7	3.98e-5
$N = 25$, $p = 1$	5.88e-6	9.91e-5	2.95e-5	4.59e-4
$N = 100$, $p = 0$	4.16e-4	1.73e-3	9.36e-4	3.76e-3
Monte Carlo	2.75e-1	5.13e-1	2.77e-1	8.31e-1

achieve is of $O(10^{-13})$, which implies that the constructed orthogonal polynomials are very accurate.

An obvious question that arises here is how well the *kp*-convergence of ME-gPC compares with the *k*-convergence or *p*-convergence. The answer is very dependent on the problem under consideration, since the properties of the solution, such as smoothness, may be significantly different. Given enough smoothness in the random space and a fixed time cost, *p*-convergence is usually a better choice than *k*-convergence or *kp*-convergence; however, *kp*-converge can be better than *p*-convergence for many cases. In the following we will measure computational complexity through n that indicates the number of operations. Using a standard gPC procedure with a Galerkin projection, we obtain the following deterministic equations for the coefficients of the polynomial chaos expansion of $u(\zeta)$:

$$\frac{\mathrm{d}u_i}{\mathrm{d}t} = \frac{1}{\mathbb{E}[\Phi_i^2(\zeta)]} \sum_{j=0}^{p_h} \sum_{k=0}^{p} h_j u_k \mathbb{E}[\Phi_i(\zeta)\Phi_j(\zeta)\Phi_k(\zeta)], \quad i = 0, 1, \ldots, p,$$

where p_h and p indicate the highest polynomial orders of polynomial chaos expansions for $h(\zeta)$ and $u(\zeta)$, respectively. Then we can use $n = N(p_h+1)(p+1)^2$ to measure the cost of ME-gPC, which is proportional to the number of operations of each time step if the Runge–Kutta method is employed for the integration in time. In Table 3.3 we present the accuracy of ME-gPC for different k and p while the cost is fixed ($n = 200$). We note that $p_h = 1$. It can be seen that *p*-convergence is superior for the case $\lambda = 1.0$. However, for a larger perturbation $\lambda = 1.5$, the ME-gPC with $N = p = 4$ achieves a better accuracy of the variance than gPC. This implies that for a large perturbation the *kp*-convergence of ME-gPC can enhance the accuracy and efficiency significantly. From (2.39), it can be seen that the degree of perturbation is reduced in ME-gPC from (λ) to $(\frac{b_i-a_i}{2}\lambda)$. Due to the fast (exponential) convergence of local gPC in each random element B_k, the overall performance of ME-gPC can be more efficient than that of gPC. We also expect that ME-gPC should be more robust than gPC for a large perturbation due to the decomposition of the random space. Furthermore, if

the solution is not smooth in the random space, the p-convergence may be destroyed and we need to resort to the (adaptive) kp-convergence, as demonstrated in the next section.

3.2.2. Kraichnan–Orszag (K-O) problem (stochastic discontinuity). It is well known that gPC fails for the K-O problem [24]. The K-O problem with uniform random inputs was studied in [28]. We now reconsider this problem with random inputs of Beta and Gaussian distributions. The transformed K-O problem can be expressed as [28]

$$\begin{aligned} \frac{\mathrm{d}y_1}{\mathrm{d}t} &= y_1 y_3, \\ \frac{\mathrm{d}y_2}{\mathrm{d}t} &= -y_2 y_3, \\ \frac{\mathrm{d}y_3}{\mathrm{d}t} &= -y_1^2 + y_2^2 \end{aligned} \tag{3.16}$$

subject to initial conditions

$$y_1(0) = y_1(0;\omega), \quad y_2(0) = y_2(0;\omega), \quad y_3(0) = y_3(0;\omega). \tag{3.17}$$

The deterministic solutions of this problem are periodic, and the period will go to infinity if the initial points are located at the planes $y_1 = 0$ and $y_2 = 0$. It was shown in [28] that gPC fails if the initial random inputs can pass through these two planes. We study the following three different kinds of initial conditions:

(i) $y_1(0) = 1$, $y_2(0) = 0.1\xi_1$, and $y_3(0) = 0$;
(ii) $y_1(0) = 1$, $y_2(0) = 0.1\xi_1$, and $y_3(0) = 0.1\xi_2$;
(iii) $y_1(0) = c\xi_1$, $y_2(0) = c\xi_2$, and $y_3(0) = c\xi_3$,

where c is constant and ξ_i are random variables of Beta or Gaussian distribution.

For all three cases, the relative error of recurrence coefficients α_i and β_i is set to be 10^{-12}. For case (i) we show the convergence of adaptive ME-gPC in Tables 3.4 and 3.5, where ξ_1 is of Beta distribution and Gaussian distribution, respectively. It can be seen that ME-gPC converges as θ_1 decreases. For all the cases in Tables 3.4 and 3.5, we have recorded the time used for the construction of orthogonal PC, which is less than 0.15% of the time used by the ME-gPC solver. Thus, the cost of the polynomial construction can be ignored compared with the total cost. In Figure 3.9, we show the adaptive meshes for Beta distribution and Gaussian distribution. Due to the symmetry of distribution, the mesh for Gaussian distribution is symmetric in contrast to the unsymmetric one for Beta distribution with $\alpha = 1$ and $\beta = 4$. From the adaptive criterion (2.33), we know that the element size is controlled by two factors: the relative error η_k and $\Pr(I_{B_k} = 1)$. We can see that around the discontinuity region ($\xi_1 = 0$) the mesh is refined because the error of gPC goes to $O(1)$, and the mesh is coarser where $\Pr(I_{B_k} = 1)$ is smaller. For the Gaussian distribution we refine only the middle element $[-6, 6]$ as before. In Figure 3.10 we show the speedup of ME-gPC for case (i) compared to the Monte Carlo (MC) method; n is defined as the number of operations. If the data in Figure 3.10 are approximated by a first-order polynomial in a least-squares sense, the accuracy of ME-gPC can be obtained as $(n^{-3.23})$, $(n^{-3.85})$, and $(n^{-5.55})$, corresponding to $p = 3$, $p = 4$, and $p = 5$, respectively, for Beta distribution and $(n^{-3.07})$, $(n^{-5.16})$, and $(n^{-6.33})$ for Gaussian distribution. It can be seen that the adaptive ME-gPC converges much faster than the Monte Carlo method and the speedup increases with the polynomial

TABLE 3.4

Maximum normalized errors of the variance of y_1, y_2, and y_3 at $t = 20$ for case (i) *of the K-O problem. $\alpha = 1/2$ and ξ_1 is of Beta distribution with $\alpha = 1$ and $\beta = 4$. (The results given by ME-gPC with $\theta_1 = 10^{-7}$ and polynomial order $p = 7$ are used as exact solutions.)*

	$\theta_1 = 10^{-2}$		$\theta_1 = 10^{-3}$		$\theta_1 = 10^{-4}$		$\theta_1 = 10^{-5}$	
	N	Error	N	Error	N	Error	N	Error
$p = 3$	24	2.98e-2	77	1.85e-3	236	3.61e-5	704	1.18e-6
$p = 4$	21	1.54e-2	45	1.05e-3	93	3.21e-4	209	4.62e-6
$p = 5$	16	6.25e-2	34	3.30e-3	57	1.50e-4	104	2.56e-6

TABLE 3.5

Maximum normalized errors of the variance of y_1, y_2, and y_3 at $t = 20$ for case (i) *of the K-O problem. $\alpha = 1/2$ and ξ_1 is of normal distribution. (The results given by ME-gPC with $\theta_1 = 10^{-7}$ and polynomial order $p = 7$ are used as exact solutions.)*

	$\theta_1 = 10^{-2}$		$\theta_1 = 10^{-3}$		$\theta_1 = 10^{-4}$		$\theta_1 = 10^{-5}$	
	N	Error	N	Error	N	Error	N	Error
$p = 3$	38	2.79e-2	98	2.21e-3	292	6.58e-5	878	1.40e-6
$p = 4$	32	4.02e-2	62	9.07e-3	128	4.34e-4	282	1.46e-6
$p = 5$	28	4.63e-2	42	3.28e-3	78	6.46e-5	138	2.61e-6

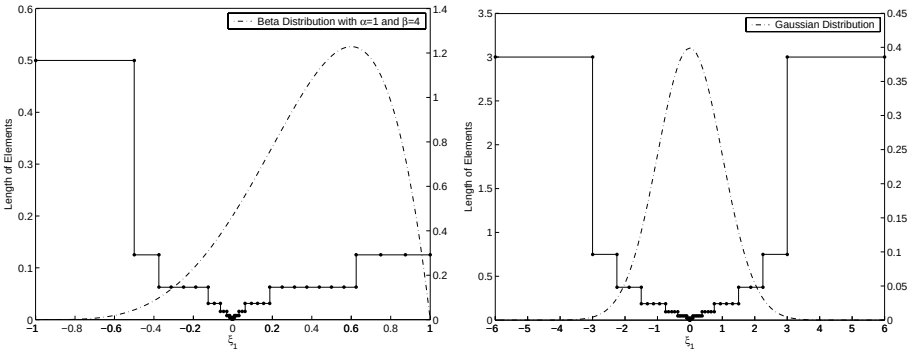


FIG. 3.9. *Adaptive mesh and corresponding random distribution for the K-O problem with one-dimensional random inputs. $p = 5$ and $\theta_1 = 10^{-3}$. Left: Beta distribution with $\alpha = 1$ and $\beta = 4$. Right: Gaussian distribution.*

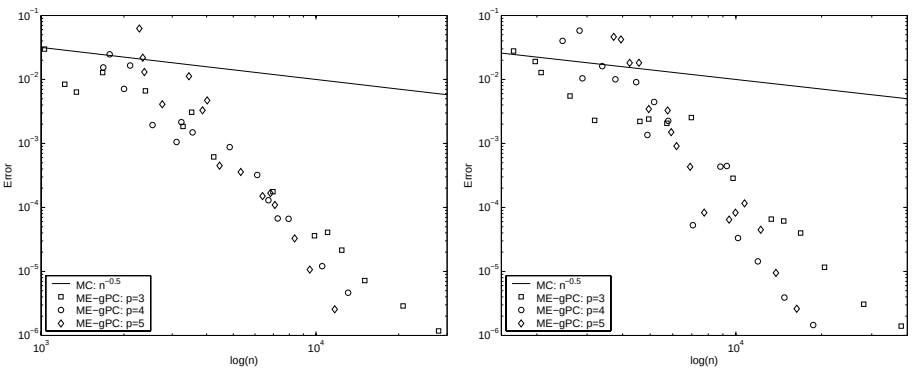


FIG. 3.10. *Speedup for the K-O problem with one-dimensional random inputs at $t = 20$. Left: Beta distribution with $\alpha = 1$ and $\beta = 4$. Right: Gaussian distribution.*

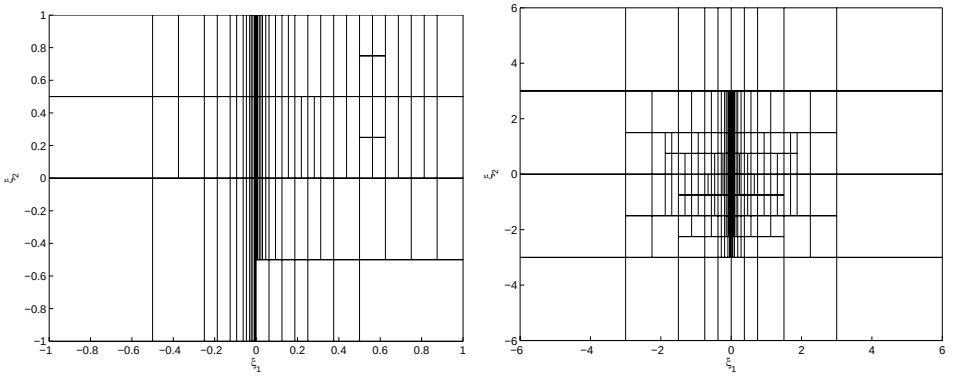


FIG. 3.11. *Adaptive meshes for case* (ii) *of the K-O problem.* $p = 4$, $\theta_1 = 10^{-4}$, *and* $\theta_2 = 10^{-2}$. *Left: Beta distribution with* $\alpha = 1$ *and* $\beta = 4$. *Right: Gaussian distribution.*

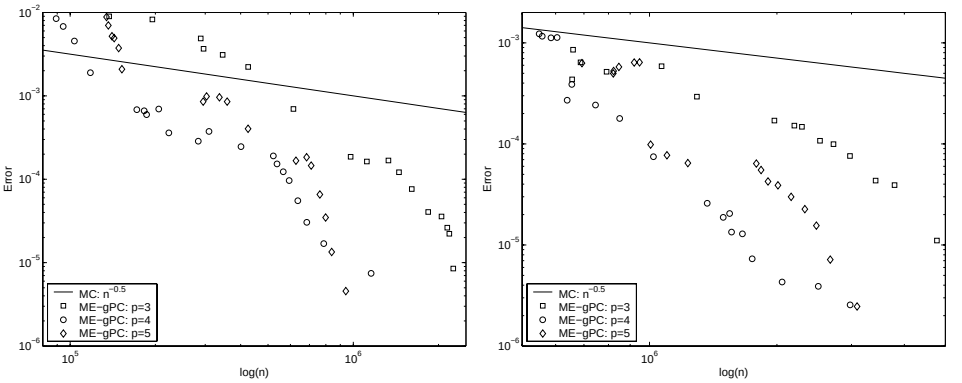


FIG. 3.12. *Speedup for case* (ii) *of the K-O problem at* $t = 10$. *Left: Beta distribution with* $\alpha = 1$ *and* $\beta = 4$. *Right: Gaussian distribution.*

order. However, since the error of gPC increases with time, the above speedups will decrease with time. The long-term behavior of gPC and ME-gPC was studied in [29].

In Figure 3.11, we show the adaptive meshes of case (ii) for Beta and Gaussian distributions. Since the discontinuity occurs at the line $\xi_1 = 0$ in the random space for this case, it can be seen that the meshes are well refined along the line $\xi_1 = 0$. It is not surprising that the random elements are bigger where the PDF is smaller because $\Pr(I_{B_k} = 1)$ is relatively smaller in these elements. In Figure 3.12, the speedup for case (ii) is shown. We can see that the Monte Carlo method is competitive for low accuracy and ME-gPC can achieve a good speedup for high accuracy.

In Figure 3.13, the evolution of y_1 in case (iii) is shown. We take $c = 1$ for the Beta distribution and 0.3 for the Gaussian distribution. For the purpose of comparison, we include the results of gPC. For both cases, gPC with $p = 3$ begins to fail at $t \approx 1$. Since increasing polynomial order does not improve gPC [28], the results of gPC with a higher order are not shown. ME-gPC can maintain convergence by increasing random elements adaptively. However, for a comparable accuracy of $O(10^{-3})$, the Monte Carlo method is about twice as fast as ME-gPC for the Beta distribution and about four times as fast as ME-gPC for the Gaussian distribution. This is mainly due to two factors. First, the discontinuity for case (iii) is strong because the discontinuity

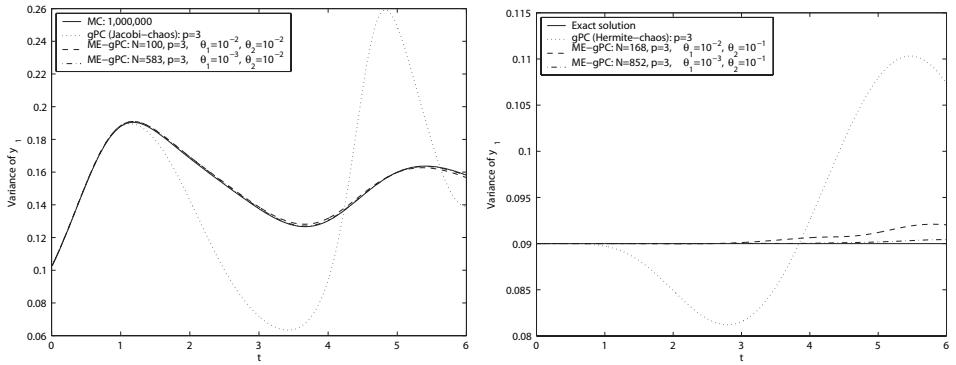


FIG. 3.13. *Evolution of the variance of* y_1 *for case* (iii) *of K-O problem. Left: Beta distribution with* $\alpha = 1$ *and* $\beta = 4$. $c = 1$. *Right: Gaussian distribution.* $c = 0.3$.

region consists of two planes: $\xi_1 = 0$ and $\xi_2 = 0$. Second, the cost of gPC increases very quickly with the random dimension. Since ME-gPC is a dimension dependent method, its efficiency decreases as the random dimension increases. The Monte Carlo method is still a better choice for obtaining a moderate accuracy for high-dimensional cases unless a high accuracy is really required.

3.2.3. ME-gPC for PDEs: Heat transfer in a grooved channel. Next, we demonstrate the use of ME-gPC to engineering applications, by simulating the heat transfer enhancement in a two-dimensional grooved channel subjected to stochastic excitation. The phenomenon of heat transfer enhancement in a grooved channel has been widely studied numerically and experimentally (see [11, 16] and references therein). The geometry to be considered is depicted in Figure 3.14. We use exactly the same geometry as in [11], which corresponds to $L = 6.6666$, $l = 2.2222$, and $a = 1.1111$.

The flow is assumed to be fully developed in the x (flow) direction. For the velocity field we have the incompressible Navier–Stokes equations

$$\frac{\partial \boldsymbol{v}}{\partial t} + (\boldsymbol{v} \cdot \nabla)\boldsymbol{v} = -\nabla \Pi + \nu \nabla^2 \boldsymbol{v}, \tag{3.18a}$$

$$\nabla \cdot \boldsymbol{v} = 0, \tag{3.18b}$$

where $\boldsymbol{v}(\boldsymbol{x}, t; \omega)$ is the stochastic velocity, Π is the pressure, and ν is the kinematic viscosity. The fully developed boundary conditions for the velocity are

$$\boldsymbol{v}(\boldsymbol{x}, t; \omega) = 0 \qquad \text{on } \partial D, \tag{3.19a}$$

$$\boldsymbol{v}(x + L, y, t; \omega) = \boldsymbol{v}(x, y, t; \omega), \tag{3.19b}$$

corresponding to no-slip and periodicity, respectively. The uncertainty is introduced by the following imposed flow-rate condition:

$$Q(t; \omega) = \int_{\partial D_B}^{\partial D_T} u(x, y, t; \omega) \mathrm{d}y = \frac{4}{3}(1 + 0.2\xi_H(\omega) \sin 2\pi \Omega_F t), \tag{3.20}$$

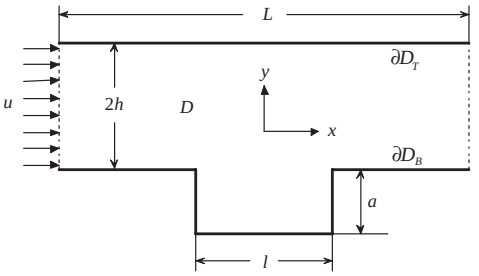


FIG. 3.14. *Schematic of the geometry of a grooved channel.*

where $\xi_H(\omega)$ is a random variable and Ω_F is a prescribed frequency. Due to the periodic boundary condition for the velocity, we require

$$\Pi(\boldsymbol{x}, t; \omega) = -h(t; \omega)x + \tilde{\Pi}(\boldsymbol{x}, t; \omega), \tag{3.21a}$$

$$\tilde{\Pi}(x, y, t; \omega) = \tilde{\Pi}(x + L, y, t; \omega) \tag{3.21b}$$

for the pressure; otherwise, a trivial solution for the velocity will be obtained. The convection equation for the temperature $T(\boldsymbol{x}, t)$ is given as

$$\frac{\partial T}{\partial t} + (\boldsymbol{v} \cdot \nabla)T = \alpha_t \nabla^2 T, \tag{3.22}$$

where α_t is the assumed constant thermal diffusivity. For the temperature boundary conditions on the top and bottom walls we have

$$\kappa \nabla T \cdot \boldsymbol{n} = 0 \quad \text{on } \partial D_T, \tag{3.23a}$$

$$\kappa \nabla T \cdot \boldsymbol{n} = 1 \quad \text{on } \partial D_B, \tag{3.23b}$$

where κ is the thermal conductivity of the fluid and $\boldsymbol{n}$ refers to the outward normal on the boundary. As for the inlet and outlet thermal boundary conditions, we first subtract a linear term from T and use a periodic boundary condition for the other contribution

$$T(\boldsymbol{x}, t; \omega) = \theta(\boldsymbol{x}, t; \omega) + \phi(\omega)x, \tag{3.24a}$$

$$\theta(x, y, t; \omega) = \theta(x + L, y, t; \omega), \tag{3.24b}$$

where the coefficient $\phi(\omega)$ can be determined by the global energy balance [10]. The numerical schemes for the aforementioned equations, which can be found in [27], are based on the deterministic splitting schemes for the Navier–Stokes equations [17, 10].

In this work we demonstrate only the k-convergence of ME-gPC by the computation of an averaged global Nusselt number. Using the same notation as in [11], we define a Nusselt number Nu as

$$Nu = L\left(1 + \frac{2a}{L}\right)^2 \Bigg/ \int_0^{L+2a} \langle \theta - \theta_b \rangle \mathrm{d}s|_{\partial D_B}, \tag{3.25}$$

TABLE 3.6
k-convergence of the ME-gPC method in terms of the variance of the Nusselt number in a grooved channel. Uniform meshes are used in the random space. The reference variance of the Nusselt number is 0.0394*, which is given by the ME-gPC method with* $N = 10$ *and* $p = 6$.

gPC	Variance	Δ_{var}	ME-gPC	Variance	Δ_{var}
$p = 1$, $n_p = 2$	0.0719	8.26e-1	$N = 1$, $p = 1$, $n_p = 2$	0.0719	8.26e-1
$p = 3$, $n_p = 4$	0.0617	5.67e-1	$N = 3$, $p = 1$, $n_p = 6$	0.0325	1.75e-1
$p = 5$, $n_p = 6$	0.0435	1.06e-1	$N = 5$, $p = 1$, $n_p = 10$	0.0355	9.84e-2
$p = 7$, $n_p = 8$	0.0430	8.41e-2	$N = 7$, $p = 1$, $n_p = 14$	0.0402	4.82e-3

where θ_b is a reference temperature taken to be the mixed-mean temperature at $x = 0$,

$$\theta_b = \left\langle \frac{3}{4} \int_{-1}^{1} u(x = 0, y, t; \omega)\theta(x = 0, y, t; \omega) \mathrm{d}y \right\rangle, \tag{3.26}$$

and $\langle \cdot \rangle$ refers to the time average over one period of the flow, $t < t' < +\Omega_F^{-1}$.

The Reynolds number is defined as $Re = 3\langle Q(t) \rangle / 4\nu$, and the Prandtl number as $Pr = \nu/\alpha$. We take $Re = 525$ and $Pr = 1$ in this work. We let $\Omega_F = 0.15$ and let ξ_H be a random variable of Beta distribution with $\alpha = 1$ and $\beta = 4$.

The convergence of gPC for the Navier–Stokes equation and the advection-diffusion equation has been studied and verified in [34, 30]. The solution of (3.18) and (3.22) converges to a steady state. We use the variance obtained by the ME-gPC method with $N = 10$, $p = 6$, and $t = 4000$ as the reference R^*. We define a relative normalized error for the variance as

$$\Delta_{var} = \left| \frac{R - R^*}{R^*} \right|, \tag{3.27}$$

where R is the variance given by gPC or ME-gPC. Let $n_p = N(p + 1)$ indicate the number of unknowns in the PC expansion. In Table 3.6, we show the k-convergence of the ME-gPC method. We note that both the mean and the variance obtained from a low-order ME-gPC are converging to the reference results as the number of elements N increases, which yields a k-convergence. The decomposition of random space accelerates the convergence significantly, but at this (relatively low) Reynolds number we need a slightly larger n_p for ME-gPC to obtain comparable accuracy with gPC. However, at higher Reynolds numbers we have larger stochastic perturbations and ME-gPC is more efficient than gPC; this is consistent with the cost comparison results of the first example presented earlier. In fact, for large values of the polynomial order p, gPC yields stochastic solutions that exhibit artificial oscillations at the tails of the corresponding PDFs; such artifacts are not present with ME-gPC [27]. Finally, compared to high-order gPC, an important cost advantage of ME-gPC for large-scale numerical simulations in fluid mechanics is that it can be easily parallelized as the local problems in each random element are independent.

In Figure 3.15, we show the standard deviation of the temperature at $t = 4000$. It can be seen that the standard deviation increases from the channel region to the groove part of the domain. The local maxima are inside the cavity and close to the top right corner of the cavity, which reflects the established flow communication between the groove and the channel. This, in turn, is the reason that gives rise to enhanced heat transfer [11].

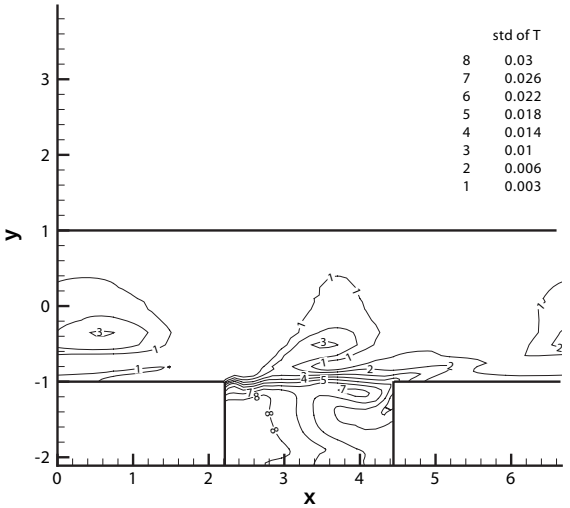


FIG. 3.15. *Standard deviation of T at $t = 4000$ obtained by ME-gPC with $N = 10$ and $p = 6$.*

4. Summary. In this paper we present an ME-gPC method for stochastic processes represented by random variables of arbitrary distributions. The ME-gPC method combines the two PC schemes of k-version and p-version by the numerical reconstruction of orthogonal polynomials, which yields a $k \times p$ version. Numerical experiments show that the numerical reconstruction of high accuracy can be obtained with negligible cost. Hence, the ME-gPC method inherits the advantages of previous PC methods and can efficiently achieve kp-convergence for stochastic differential equations. This is shown by a simple ODE model and a heat transfer problem in a grooved channel. The application of ME-gPC depends on the smoothness of the solution in the random space. Due to the fast (exponential) convergence, gPC is preferred if the p-convergence can be achieved; otherwise, the convergence can be improved by the kp-convergence of ME-gPC. The adaptive ME-gPC scheme shows a good performance in convergence for problems related to discontinuities induced by random inputs because it can weaken the influence of discontinuities by small random elements and maintain the fast (exponential) convergence of gPC in regular elements, which is clearly shown by the adaptive meshes for the K-O problem. Thus, the ME-gPC method is more flexible and efficient than previous polynomial chaos schemes in many cases.

In [1], the asymptotic numerical complexity of stochastic Galerkin finite element methods of $k \times h$ version and $p \times h$ version is compared with the Monte Carlo method. Note that the comparison is valid only for the optimal case. For some cases, especially when the random dimension is high and the desired accuracy is moderate, the Monte Carlo method is still preferred in practice because it is random dimension independent while all PC methods including the ME-gPC method depend on the random dimension. If strong discontinuities exist together with a high random dimension as shown in case (iii) of section 3.2.2, polynomial chaos methods can lose efficiency further or even fail to converge. However, if the required error is sufficiently small, the PC method, especially the ME-gPC method, will be more efficient than the Monte

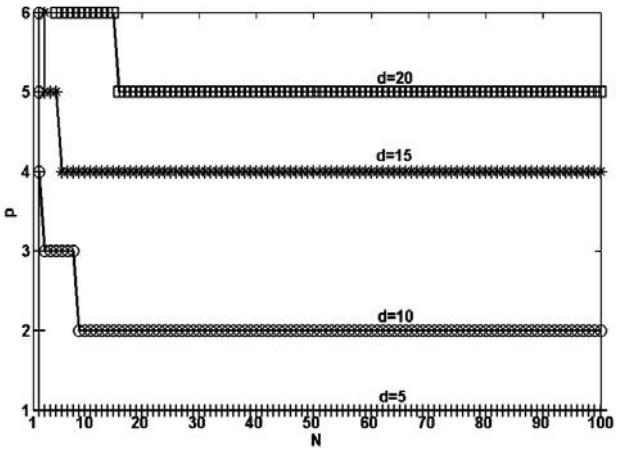


FIG. 4.1. *Cost comparison between ME-gPC and the standard Monte Carlo method for fixed accuracy. Here p is the order of gPC in each element, N is the number of elements and d is the number of dimensions. The symbol-lines are iso-cost lines.*

Carlo method due to the k-p convergence as shown in cases (i) and (ii) of section 3.2.2. In Figure 4.1, we present a comparison between the k-convergence $(N^{-2(p+1)})$ of ME-gPC and the convergence $(n^{-1/2})$ of the standard Monte Carlo method; see [1]. For the same accuracy and different random dimension numbers, the lines show the cases where the cost of the standard Monte Carlo method is equal to that of ME-gPC. For a certain random dimension number (denoted by "d"), the region below the line is where MC is more efficient; the region above the line is where ME-gPC is more efficient.

In this work we use heuristically the decay rate of relative error of variance as the indicator for the k-type refinement. A more rigorous a posteriori error estimate is still needed for the kp-adaptivity. These issues will be addressed in future publications.

Appendix (application of ME-gPC to a stochastic elliptic problem). Here we briefly elaborate on how to apply ME-gPC to solve differential equations with stochastic coefficients using the following stochastic linear boundary value problem: Find a stochastic function, $u : \overline{D} \times \Omega \to \mathbb{R}$, such that almost surely the following equation holds:

$$\begin{aligned} -\nabla \cdot (a(\boldsymbol{x};\omega)\nabla u(\boldsymbol{x};\omega)) &= f(\boldsymbol{x};\omega) \quad \text{on } D, \\ u(\boldsymbol{x};\omega) &= 0 \quad \text{on } \partial D, \end{aligned} \tag{4.1}$$

where D is an open domain in the physical space with Lipschitz boundaries, $a(\boldsymbol{x};\omega)$ and $f(\boldsymbol{x};\omega)$ are second-order random processes. We assume $0 < \alpha_1 \le a(\boldsymbol{x};\omega) \le \alpha_2 < \infty$ almost everywhere on $\overline{D} \times \Omega$. Using the K-L expansion and following the procedure in [1], we assume that $a(\boldsymbol{x};\omega) \approx a(\boldsymbol{x};\boldsymbol{\xi}(\omega))$ and $f(\boldsymbol{x};\omega) \approx f(\boldsymbol{x};\boldsymbol{\xi}(\omega))$, where the same $\mathbb{R}^d$-valued random variable $\boldsymbol{\xi}$ with i.i.d. components is used for both $a(\boldsymbol{x};\omega)$ and $f(\boldsymbol{x};\omega)$ in their K-L expansions. We note here that the "i.i.d." assumption is strong since it is sufficient for the random variables in a K-L expansion to be mutually uncorrelated.

In each random element B_k we naturally have the following problem:

$$\begin{aligned} -\nabla \cdot (a(\boldsymbol{x};\boldsymbol{\xi}_k)\nabla u(\boldsymbol{x};\boldsymbol{\xi}_k)) &= f(\boldsymbol{x};\boldsymbol{\xi}_k) \quad \text{on } D, \\ u(\boldsymbol{x};\boldsymbol{\xi}_k) &= 0 \quad \text{on } \partial D. \end{aligned} \tag{4.2}$$

All previous assumptions on $a(\boldsymbol{x};\boldsymbol{\xi}_k)$ and $f(\boldsymbol{x};\boldsymbol{\xi}_k)$ are satisfied with respect to a conditional PDF $\hat{f}_k(\boldsymbol{\xi}_k|I_{B_k}=1)$. We note that such a system is complete on each random element B_k.

We define a bilinear form $\mathscr{B}(\cdot,\cdot)$ as

$$\mathscr{B}(u,v) = \int_{B_k} \hat{f}_k(\boldsymbol{\xi}_k|I_{B_k}=1) \int_D a(\boldsymbol{x};\boldsymbol{\xi}_k)\nabla u(\boldsymbol{x};\boldsymbol{\xi}_k)\cdot\nabla v(\boldsymbol{x};\boldsymbol{\xi}_k)\mathrm{d}\boldsymbol{x}\mathrm{d}\boldsymbol{\xi}_k \tag{4.3}$$

and a linear form $\mathscr{L}(\cdot)$ as

$$\mathscr{L}(v) = \int_{B_k} \hat{f}_k(\boldsymbol{\xi}_k|I_{B_k}=1) \int_D f(\boldsymbol{x};\boldsymbol{\xi}_k)v(\boldsymbol{x};\boldsymbol{\xi}_k)\mathrm{d}\boldsymbol{x}\mathrm{d}\boldsymbol{\xi}_k. \tag{4.4}$$

Due to the assumption on $a(\boldsymbol{x};\boldsymbol{\xi}_k)$ the space

$$W(D,B_k,\hat{f}_k) = \left\{v(\boldsymbol{x};\boldsymbol{\xi}_k)|\mathscr{B}(v,v) = \|v\|_W^2 < \infty \quad \forall v \in L_2(B_k)\otimes H_0^1(D)\right\} \tag{4.5}$$

is a Hilbert space. Using the Lax–Milgram theorem, we can claim that a unique solution $\hat{u}_k(\boldsymbol{x};\boldsymbol{\xi}_k)$ exists for the problem

$$\mathscr{B}(u,v) = \mathscr{L}(v) \quad \forall v \in W. \tag{4.6}$$

To this end, the gPC procedure can be readily employed in each element B_k on a truncated subspace $V^r \otimes V^p \subset W$to solve (4.2) [14, 32], where V^r indicates the local orthogonal basis $\{\Phi_{k,i}(\boldsymbol{\xi}_k)\}$ in the random space and V^p denotes the piecewise polynomial basis in the physical space. Once the local approximation is obtained, the global approximation can be constructed by Proposition 2.2.

Using a similar procedure ME-gPC can be readily applied to any differential equations with stochastic coefficients.

REFERENCES

[1] I. Babuška, R. Tempone, and G. E. Zouraris, *Galerkin finite element approximations of stochastic elliptic partial differential equations*, SIAM J. Numer. Anal., 42 (2004), pp. 800–825.

[2] D. Boley and G. H. Golub, *A survey of matrix inverse eigenvalue problems*, Inverse Problems, 3 (1987), pp. 595–622.

[3] R. Cameron and W. Martin, *The orthogonal development of non-linear functionals in series of Fourier–Hermite functionals*, Ann. of Math. (2), 48 (1947), pp. 385–392.

[4] M. K. Deb, I. Babuška, and J. T. Oden, *Solution of stochastic partial differential equations using Galerkin finite element techniques*, Comput. Methods Appl. Mech. Engrg., 190 (2001), pp. 6359–6372.

[5] H. J. Fischer, *On the condition of orthogonal polynomials via modified moments*, Z. Anal. Anwendungen, 15 (1996), pp. 1–18.

[6] H. J. Fischer, *On generating orthogonal polynomials for discrete measures*, Z. Anal. Anwendungen, 17 (1998), pp. 183–205.

[7] P. Frauenfelder, C. Schwab, and R. A. Todor, *Finite elements for elliptic problems with stochastic coefficients*, Comput. Methods Appl. Mech. Engrg., 194 (2005), pp. 205–228.

[8] W. Gautschi, *On generating orthogonal polynomials*, SIAM J. Sci. Statist. Comput., 3 (1982), pp. 289–317.

[9] W. Gautschi, *Algorithm* 726: *ORTHPOL—A package of routines for generating orthogonal polynomials and Gauss-type quadrature rules*, ACM Trans. Math. Software, 20 (1994), pp. 21–62.
[10] N. K. Ghaddar, G. E. Karniadakis, and A. T. Patera, *A conservative isoparametric spectral element method for forced convection: Application to fully developed flow in periodic geometries*, Num. Heat Transfer, 9 (1986), pp. 277–300.
[11] N. K. Ghaddar, M. Magen, B. B. Mikic, and A. T. Patera, *Numerical investigation of incompressible flow in grooved channels.* II. *Resonance and oscillatory heat-transfer enhancement*, J. Fluid Mech., 168 (1986), pp. 541–567.
[12] R. G. Ghanem, *Ingredients for a general purpose stochastic finite element formulation*, Comput. Methods Appl. Mech. Engrg., 168 (1999), pp. 19–34.
[13] R. G. Ghanem, *Stochastic finite elements for heterogeneous media with multiple random non-Gaussian properties*, ASCE J. Engrg. Mech., 125 (1999), pp. 26–40.
[14] R. Ghanem and P. Spanos, *Stochastic Finite Elements: A Spectral Approach*, Springer-Verlag, New York, 1991.
[15] R. D. Henderson, *Dynamic refinement algorithms for spectral element methods*, Comput. Methods Appl. Mech. Engrg., 175 (1999), pp. 395–411.
[16] C. Herman and E. Kang, *Heat transfer enhancement in a grooved channel with curved vanes*, Int. J. Heat Mass Transfer, 45 (2002), pp. 3741–3757.
[17] G. Karniadakis and S. Sherwin, *Spectral/hp Element Methods for CFD*, Oxford University Press, New York, 1999.
[18] O. P. Le Maitre, H. N. Njam, R. G. Ghanem, and O. M. Knio, *Uncertainty propagation using Wiener–Haar expansions*, J. Comput. Phys., 197 (2004), pp. 28–57.
[19] O. P. Le Maitre, H. N. Njam, R. G. Ghanem, and O. M. Knio, *Multi-resolution analysis of Wiener-type uncertainty propagation schemes*, J. Comput. Phys., 197 (2004), pp. 502–531.
[20] H. G. Matthies and C. G. Bucker, *Finite elements for stochastic media problems*, Comput. Methods Appl. Mech. Engrg., 168 (1999), pp. 3–17.
[21] H. G. Matthies and A. Keese, *Galerkin methods for linear and nonlinear elliptic stochastic partial differential equations*, Comput. Methods Appl. Mech. Engrg., 194 (2005), pp. 1295–1331.
[22] C. Mavriplis, *Adaptive mesh strategies for the spectral element method*, Comput. Methods Appl. Mech. Engrg., 116 (1994), pp. 77–86.
[23] S. E. Notaris, *Interpolatory quadrature formulae with Chebyshev abscissae*, J. Comput. Appl. Math., 133 (2001), pp. 507–517.
[24] S. A. Orszag and L. R. Bissonnette, *Dynamical properties of truncated Wiener-Hermite expansions*, Physics Fluids, 10 (1967), pp. 2603–2613.
[25] C. Schwab, *p- and hp- Finite Element Methods: Theory and Applications in Solid and Fluid Mechanics*, The Clarendon Press, Oxford University Press, New York, 1998.
[26] G. Szegö, *Orthogonal Polynomials*, American Mathematical Society, Providence, RI, 1959.
[27] X. Wan, *Multi-Element Generalized Polynomial Chaos for Differential Equations with Random Imputs: Algorithms and Applications*, Ph.D. thesis, Division of Applied Mathematics, Brown University, Providence, RI, in preparation.
[28] X. Wan and G. E. Karniadakis, *An adaptive multi-element generalized polynomial chaos method for stochastic differential equations*, J. Comput. Phys., 209 (2005), pp. 617–642.
[29] X. Wan and G. E. Karniadakis, *Long-term behavior of polynomial chaos in stochastic flow simulations*, Comput. Methods Appl. Mech. Engrg., in press.
[30] X. Wan, D. Xiu, and G. E. Karniadakis, *Stochastic solutions for the two-dimensional advection-diffusion equation*, SIAM J. Sci. Comput., 26 (2004), pp. 578–590.
[31] N. Wiener, *The homogeneous chaos*, Amer. J. Math., 60 (1938), pp. 897–936.
[32] D. Xiu and G. E. Karniadakis, *The Wiener–Askey polynomial chaos for stochastic differential equations*, SIAM J. Sci. Comput., 24 (2002), pp. 619–644.
[33] D. Xiu and G. E. Karniadakis, *Modeling uncertainty in flow simulations via generalized polynomial chaos*, J. Comput. Phys., 187 (2003), pp. 137–167.
[34] D. Xiu and G. E. Karniadakis, *A new stochastic approach to transient heat conduction modeling with uncertainty*, Int. J. Heat Mass Transfer, 46 (2003), pp. 4681–4693.

SIAM J. SCI. COMPUT.
Vol. 28, No. 3, pp. 929–941
© 2006 Society for Industrial and Applied Mathematics

NUMERICAL SIMULATIONS IN 3D HEAT CONDUCTION: MINIMIZING THE QUADRATIC MEAN TEMPERATURE GRADIENT BY AN OPTIMALITY CRITERIA METHOD*

ALBERTO DONOSO†

Abstract. We analyze an optimal design problem in three-dimensional (3D) heat conduction: Given fixed amounts of two isotropic conducting materials, decide how we are to mix them in a 3D domain to minimize the quadratic mean temperature gradient. By using an optimality criteria method, we provide some numerical evidence that Tartar's result (see [in *Calculus of Variations, Homogenization and Continuum Mechanics*, G. Buttazzo, G. Bouchitte, and P. Suquet, eds., World Scientific, River Edge, NJ, 1994, pp. 279–296]) is verified in three dimensions when the target vector field is zero.

Key words. optimal design, optimality conditions, graded material

AMS subject classifications. 49J15, 49M05

DOI. 10.1137/060650453

1. Introduction.

We consider the following optimal design problem in heat conduction: Given two isotropic conducting materials, with thermal conductivities α and β, $0 < \alpha < \beta$, decide on the best way to mix them in a simply connected design domain $\Omega \subset \mathrm{R}^3$ (and the emphasis is placed on the three-dimensional (3D) situation), so as to minimize the temperature gradient, ∇u, when the amounts of both materials are fixed and the body is subjected to the thermal load, f, and the temperature boundary condition, u_0.

Let us denote by (VP) the following problem:
Minimize

$$I(\chi) = \frac{1}{2}\int_\Omega |\nabla u(x)|^2\, dx$$

subject to

$$\begin{aligned} -\mathrm{div}((\alpha\chi(x) + \beta(1-\chi(x)))\nabla u(x)) &= f(x) \quad \text{in } \Omega, \\ u &= u_0 \quad \text{on } \partial\Omega, \\ \frac{1}{|\Omega|}\int_\Omega \chi(x)\, dx &= V_0, \quad 0 < V_0 < 1. \end{aligned}$$

Here $\Omega \subset \mathrm{R}^3$ is a simply connected regular domain. Typically, the characteristic function χ indicates where the α-material is to be placed, and V_0 is the relative volume fraction of the α-material, which is given.

It is well known that such problems lack optimal solutions within the class of characteristic functions [9]. In such cases, the optimal design is obtained by placing a graded material, that is to say, a two-component composite characterized by a gradual variation from the α-material to the β-material. The analysis when the cost functional

*Received by the editors December 12, 2005; accepted for publication (in revised form) January 20, 2006; published electronically June 19, 2006. This research was supported by MTM2004-07114 (Spain) and by PAI-05-027 (Spain).
http://www.siam.org/journals/sisc/28-3/65045.html
†Departamento de Matemáticas, ETSI Industriales, Universidad de Castilla-La Mancha, 13071 Ciudad Real, Spain (Alberto.Donoso@uclm.es).

does not explicitly depend on derivatives of the temperature has been carried out by means of the theory of homogenization. Some recent nice accounts are [1] and [4]. Another interesting reference in the context of elastic structures is [2], where both problems, the minimization of the compliance for multiple surface loadings (in two and three dimensions) and the maximization of the first eigenfrequency, are addressed. Quadratic functionals on ∇u, like the one above, have been explored (see [7], [8], [11]), trying to extend the ideas of homogenization and H-convergence to treat this more general situation.

There exists another recent perspective to tackle the design problem. The idea is based on introducing a suitable potential to reformulate the problem as a vector variational problem and, afterwards, relax the reformulated problem by computing the explicit quasi convexification. For the 3D case we present, explicit formulae of the relaxed integrand have been recently computed in [5], and it has also been possible for a general quadratic functional in two dimensions (see [10]). In all of these cases, the relaxed integrands take the value $+\infty$ in a certain domain. This makes us think that it could complicate the numerical treatment of the relaxed problem. On the other hand, we also think it could be interesting to develop a tool which allow us to analyze in a systematic way this kind of design problem under this approach.

The pioneering work [11] addressed for the first time this optimal design problem. One of the main achievements was to conclude that for F in a dense G_δ, but unknown set of L^2, the optimization problem, $(\widetilde{VP})$,

Minimize:

$$I(t) = \frac{1}{2}\int_\Omega |\nabla u(x) - F(x)|^2 \, dx$$

subject to

$$\begin{aligned} -\mathrm{div}((\alpha t(x) + \beta(1-t(x)))\nabla u(x)) &= f(x) \quad \text{in } \Omega, \\ u &= u_0 \quad \text{on } \partial\Omega, \\ \frac{1}{|\Omega|}\int_\Omega t(x)\, dx &= V_0, \quad 0 < V_0 < 1, \end{aligned}$$

where the function $t(x)$ is now constrained by

$$0 \le t(x) \le 1,$$

admits optimal solutions. Moreover, if the pair $(\tilde{u},\tilde{t})$ is an optimal solution for $(\widetilde{VP})$, optimal structures for (VP) (when a target vector field, F, is considered) are locally first-order laminates with local volume fraction $\tilde{t}(x)$, the temperature gradient, $\nabla\tilde{u}$, is perpendicular to the direction of lamination, and all minimizing sequences of temperature gradients converge strongly in L^2.

Our approach has consisted of obtaining and solving optimality conditions for $(\widetilde{VP})$ when $F \equiv 0$. We have not been able to show that $(\widetilde{VP})$ has optimal solutions. This is essentially Tartar's result in [11] for such a G_δ set as explained before, but for that particular case ($F \equiv 0$) we provide some numerical evidence that the zero field belongs to that G_δ set. In all experiments we have conducted the convergence was fine and the simulations stable and robust. Under these circumstances, we suspect that the infima of $(\widetilde{VP})$ are, in fact, minima. This is our main contribution with this paper. In fact this same evidence also holds for a general quadratic functional. All

these numerical experiments have been carried out for the two-dimensional (2D) case in [6].

It is worth emphasizing that in this paper, the 3D case of a variational problem in the presence of gradients in the objective function is treated for the first time, as far as the author knows.

The motivation for solving (VP) is the design of macroscopic thermal conducting devices, made up of graded materials with desired properties under the action of the thermal load, f, and the value of the temperature, u_0, on the boundary. In this case, we measure the dissipated energy by means of minimizing the quadratic mean temperature gradient.

The work is organized as follows. We begin obtaining optimality conditions for $(\widetilde{VP})$. Later, we explain how to solve the nonlinear resulting system and, finally, show different numerical examples, changing the source term, f, the relative volume fraction, V_0, and the contrast ratio (β/α).

2. Numerical approach: Optimality conditions. There are mainly two types of iterative methods [1] in optimal design, both of which alternate evaluations of the state and the adjoint state and also update the design parameters. The first type is the so-called . Its principle is to use optimality conditions to update the design parameters. This is a very popular method commonly used in the structural engineering community (see [4] and the references therein). The second method is more classical and is based on gradient information. Although it can include several variants to improve properties concerning convergence, it is well known as the . There is a vast literature on this kind of descent method (see, for instance, [3] and the references therein). It is important to notice that in [8] a gradient-based optimization scheme was developed for the same design problem considered here but in the 2D context. In that work it is shown that the numerically optimized designs support Tartar's result [11] in two dimensions when the target vector field is zero. In this work an optimality criteria method is used to solve the design problem in the 3D situation.

Looking back to the design problem, we can state that if $(\widetilde{VP})$ admits optimal solutions, they can be approximated by using the necessary conditions of optimality.

PROPOSITION 2.1. f u_0 α β V_0 $\widetilde{VP}$

$$
\begin{aligned}
-\operatorname{div}((\alpha t(x)+\beta(1-t(x)))\nabla u(x)) &= f(x) \quad \Omega,\\
u &= u_0 \quad \partial\Omega,\\
-\operatorname{div}((\alpha t(x)+\beta(1-t(x)))\nabla p(x)) &= -\Delta u(x) \quad \Omega,\\
p &= 0 \quad \partial\Omega,\\
(\beta-\alpha)\nabla u(x)\nabla p(x) &= \lambda_1(x)-\lambda_2(x)+\lambda_3,\\
\lambda_1(x)t(x) &= 0, \quad \lambda_1(x)\geq 0, \quad t(x)\geq 0,\\
\lambda_2(x)(t(x)-1) &= 0, \quad \lambda_2(x)\geq 0, \quad t(x)\leq 1,\\
\frac{1}{|\Omega|}\int_\Omega t(x)\,dx &= V_0.
\end{aligned}
$$

The numerical algorithm used works in a simple way: Given an admissible control t_k at iteration step k, we solve the state equation to obtain the gradient ∇u_k and, using both, we get ∇p_k. Having in mind that for intermediate controls[1] $(0<t_k<1)$,

[1] In fact, the control can be as close to either of the extreme values, 0 and 1, as we wish.

$\lambda_1 = \lambda_2 = 0$ and hence

$$\frac{(\beta-\alpha)\nabla u \nabla p}{\lambda_3} = 1,$$

we choose as the iterative sequence a heuristic updating scheme [4], $t_{k+1} = 1 - G_k(1 - t_k)$ to update[2] the control, where $G_k = \frac{(\beta-\alpha)\nabla u_k \nabla p_k}{\lambda_{3,k}}$ (notice that when $G_k \to 1$, then $t_{k+1} \to t_k$). Since the update t_{k+1} depends on the present value of the Lagrange multiplier $\lambda_{3,k}$, then $\lambda_{3,k}$ must be adjusted in an inner iteration loop by using, for example, a bisection method, to ensure that since t_{k+1} verifies the volume constraint at iteration step $k+1$. In this case since

$$\int_\Omega t_{k+1}\, dx = \int_\Omega (1 - G_k(1-t_k))\, dx = \int_\Omega \left(1 - \frac{(\beta-\alpha)\nabla u_k \nabla p_k}{\lambda_{3,k}}(1-t_k)\right)\, dx = V_0|\Omega|,$$

we arrive at

$$\frac{1}{\lambda_{3,k}} \int_\Omega (\beta-\alpha)\nabla u_k \nabla p_k (1-t_k)\, dx = (1-V_0)|\Omega| > 0 \tag{1}$$

and find that

$$\lambda_{3,k} = \frac{\int_\Omega (\beta-\alpha)\nabla u_k \nabla p_k (1-t_k)\, dx}{(1-V_0)|\Omega|}.$$

As we mentioned, there are several type of gradient algorithms. The simplest would be a fixed step-size descent method. In such a case the control must be updated by (in a compacted way)

$$t_{k+1} = \max(0, \min(1, t_k - I'(t_k) s_k)),$$

where s_k is a small step such that $I(t_{k+1}) < I(t_k)$, and $I'(t_k)$ is the gradient of the objective function with respect to the control, easily computed as

$$I'(t_k) = (\alpha - \beta)\nabla u_k \nabla p_k + \lambda_{3,k}.$$

Therefore, when $G_k \to 1$, this in fact means that $I'(t_k) \to 0$, and we have converged, at least, to a local minimum. Moreover, having in mind that $I'(t_k) = \lambda_{3,k}(1 - G_k)$, it is easy to prove that the heuristic updating scheme, $t_{k+1} = 1 - G_k(1-t_k)$, and the fixed step-size descent method, $t_{k+1} = t_k - I'(t_k)s_k$, are identically the same by taking $s_k = \frac{(t_k - 1)}{\lambda_{3,k}}$.

The general iteration scheme used for obtaining all numerical results presented in this work has the following form.

START At iteration k, take an admissible control t_k, such as $0 < t_k < 1$, and let $\int_\Omega t_k\, dx = V_0|\Omega|$.

Step 1: Given t_k, obtain ∇u_k by solving the state equation,

$$\begin{aligned} -\mathrm{div}((\alpha t_k + \beta(1-t_k))\nabla u_k) &= f \quad \text{in } \Omega, \\ u_k &= u_0 \quad \text{on } \partial\Omega. \end{aligned}$$

[2]Notice that in the design problems addressed in [4] under optimality criteria the law used to update the control is $t_{k+1} = G_k t_k$, and the control t corresponds to the β-material property in the state equation. Following the same idea, we have chosen the law $1 - t_{k+1} = G_k(1-t_k)$ because $(1-t)$ now accompanies the β-material property in the state equation.

Step 2: Get ∇p_k by solving the adjoint state equation,

$$\begin{aligned} -\mathrm{div}((\alpha t_k + \beta(1-t_k))\nabla p_k) &= -\mathrm{div}(\nabla u_k) \quad \text{in } \Omega, \\ p_k &= 0 \quad \text{on } \partial\Omega. \end{aligned}$$

Step 3: Compute the Lagrange multiplier, $\lambda_{3,k}^j$, as

$$\lambda_{3,k}^j = \frac{\int_\Omega (\beta-\alpha)\nabla u_k \nabla p_k (1-t_k^j)\, dx}{(1-V_0)|\Omega|},$$

in such a way that $t_k^{j+1} = 1 - G_k^j(1-t_k^j)$ verifies the volume constraint.

Step 4: Update t_k^{j+1} as

$$t_k^{j+1} = \begin{cases} 1 - \frac{1-t_k^j}{2} & \text{if} \quad G_k^j(1-t_k^j) \le \frac{1-t_k^j}{2}, \\ 1 - G_k^j(1-t_k^j) & \text{if} \quad \frac{1-t_k^j}{2} \le G_k^j(1-t_k^j) \le \frac{1+(1-t_k^j)}{2}, \\ 1 - \frac{1+(1-t_k^j)}{2} & \text{if} \quad \frac{1+(1-t_k^j)}{2} \le G_k^j(1-t_k^j). \end{cases}$$

Step 5: Ensure that $\int_\Omega t_k^{j+1}\, dx = V_0|\Omega|$. If the volume constraint is verified, go to **Step 6** and $t_{k+1} = t_k^{j+1}$. Otherwise, go to **Step 3** ($j \leftarrow j+1$) and calculate the new $\lambda_{3,k}^{j+1}$.

Step 6: Stopping criteria: $|I(t_{k+1}) - I(t_k)| \le tol$. If it is not verified, go to **Step 1** ($k \leftarrow k+1$).

END

We have solved both the state equation and the adjoint one (Steps 1 and 2) by minimizing their associated energy functional and have approached them by using piecewise linear basis functions and finite differences.

By using the fixed point scheme for the control (Step 4), each design variable is updated independently of the update of the other design variables, except for the rescaling (Step 5) that has to take place to satisfy the volume constraint [4]. Instead of using the ("more natural") updating $t_{k+1} = \max(0, \min(1, 1 - G_k(1-t_k)))$, the adjustment process of Step 4 allows the control to converge in a more gradual way, ensuring that at iteration step $k+1$, $0 < t_{k+1} < 1$. In the inner loop (Steps 2, 3, and 4), $\lambda_{3,k}$ is adjusted in order to satisfy the volume constraint.

Finally, it is not hard to realize that the function used to adjust the volume constraint, $g(\lambda_{3,k}) = \int_\Omega t_{k+1}(\lambda_{3,k})\, dx - V_0|\Omega|$, has a continuous monotone dependence on the Lagrange multiplier, $\lambda_{3,k}$. We can convince ourselves by taking into account the expression (1). If $\int_\Omega (\beta-\alpha)\nabla u_k \nabla p_k (1-t_k)\, dx > 0$, then $\lambda_{3,k} > 0$, and the function g is strictly decreasing. By contrast, when $\int_\Omega (\beta-\alpha)\nabla u_k \nabla p_k (1-t_k)\, dx < 0$, then $\lambda_{3,k} < 0$ and the function g is strictly increasing; thus we can use the bisection method to determine when the function becomes zero in both situations (see Figure 1).

The stability and robustness of the algorithm have been tested by running the same numerical examples with different mesh sizes and different starting points, respectively, obtaining very similar results. This let us conclude that $(\widetilde{VP})$ admits, first, optimal solutions (this indicates existence) and, second, uniqueness. Concerning the convergence, it has been checked that the objective function slightly changes its value and therefore, the L^∞-norm of the difference $I(t_{k+1}) - I(t_k)$ reduces to zero, as we approach the minimum. All numerical examples belonging to this work, with the

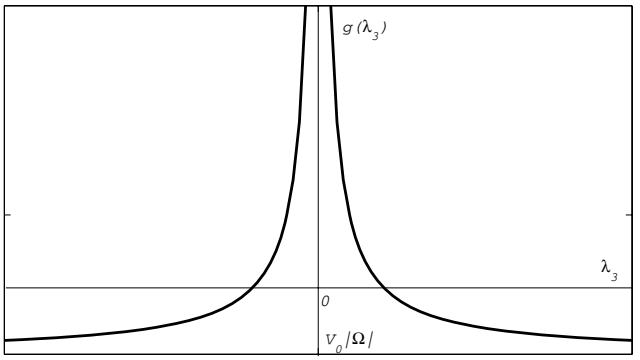


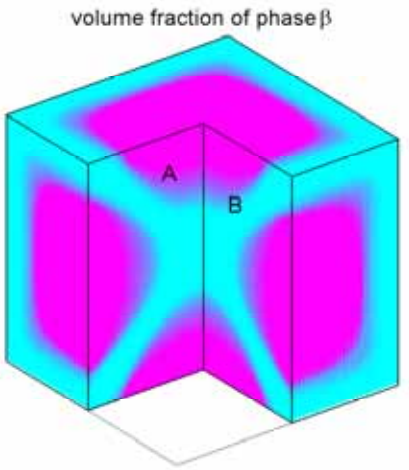
FIG. 1. *Volume adjustment using the bisection method.*

exception of the first one, have been obtained under the stopping criteria, $tol = 10^{-4}$, requiring for convergence about 100 iterations of the algorithm, on average. However, when plotting the density map, a "visual" convergence is reached long before this, in about 40 iterations.

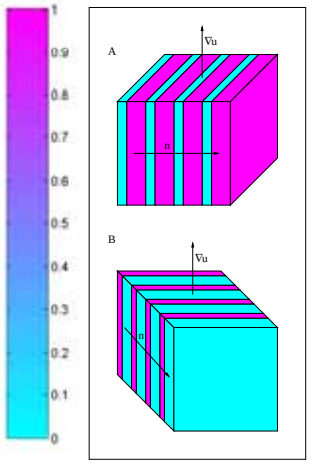
3. Numerical examples. We illustrate our approach through several numerical examples. In all of them, the design domain is chosen to be the cube $\Omega = [0,1]^3$ which is subdivided into 125,000 tetrahedric subdomains (a mesh of $50 \times 50 \times 50$ nodes). Although it is known that the optimal solution is given by a locally first-order laminate, in some areas the best choice is to place a pure phase, as we will see later. The following numerical simulations show optimal layouts when the value of the temperature is fixed on the boundary and also constant, $u_0 = 0$, and the thermal load applied over the body is always positive. Therefore, intuition tells us that the material with higher conductivity should be placed where the outward heat flux is also higher in order to drain the energy away and hence minimize the temperature gradient inside the body.

In the first example the source term is uniform in Ω, for example, $f(x) = 1$, and the material properties take the values $\alpha = 1$ and $\beta = 2$. We have constrained the amount of β (best conductor) to be 40% of the design domain (it corresponds to taking $V_0 = 0.6$). A graded scale has been used to represent the optimal solution (see Figure 2(a)) where aqua would correspond to the α-material and pink to the β-material. The regions of graded properties are given by intermediate shades. In such zones, it is necessary to know the direction of the laminates. This information is provided by Figure 2(c), where the contours are the level surfaces of the temperature. Note that the direction of the laminates is tangential to the level surfaces, hence perpendicular to the temperature gradient. Contrary to what happened in two dimensions [6], now the direction of lamination, n, is contained in the manifold of normal ∇u, so, at first, we have infinite possibilities, that is, first-order laminates with different n but with the same functional value. We have tried to indicate this microstructural behavior in the two points A and B in Figure 2(b).

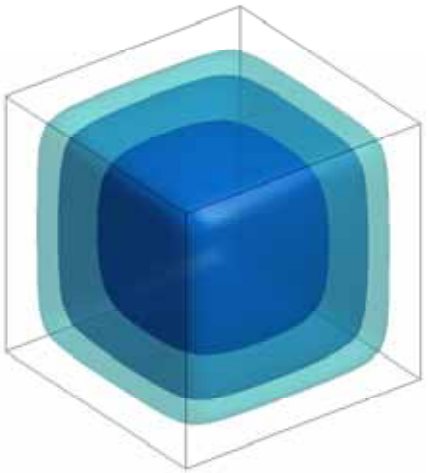
In Figure 2(a) a half-section of the body is depicted in order to see the internal details. In Figures 2(d), 2(e), and 2(f), some sections are also shown in order to understand better what happens as we look inside. For this example, the i-level indicates the i-slice in any characteristic direction because of the symmetry in the input data. The convergence history for the objective function and for the residual term is also included in Figures 3(a) and 3(b), respectively ($tol = 10^{-5}$).



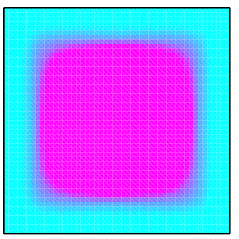
(a) Half-section



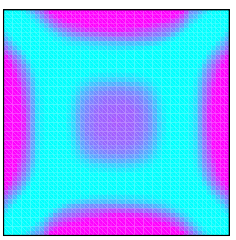
(b) Laminate 3d



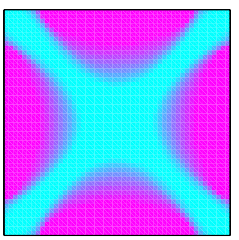
(c) Isosurfaces temperature



(d) 1-level

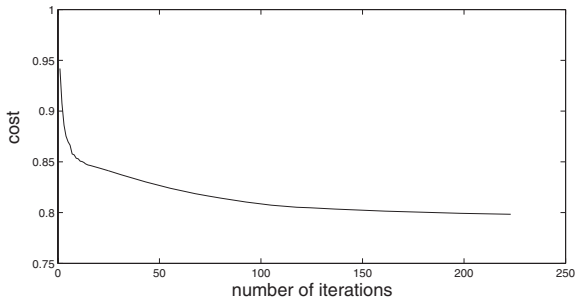


(e) 12-level

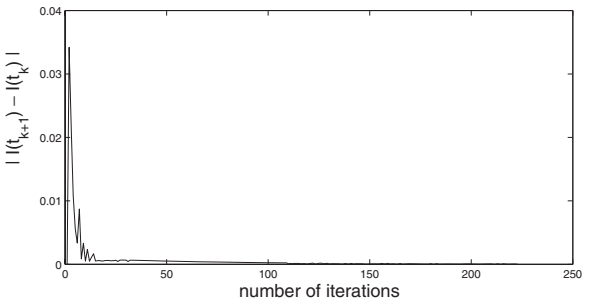


(f) 25-level

FIG. 2. *Optimal solution of Example* 1.



(a) Objective function



(b) Residual term

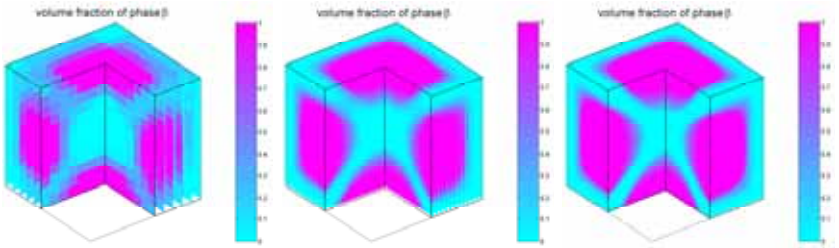
FIG. 3. *Convergence history of Example* 1.

As commented before, very similar results are obtained when different mesh sizes and different starting points are used. Although this numerical analysis has been carried out for all examples included in this work, we show such comparisons only for the previous example (Example 1 but now with $V_0 = 0.5$), while in the following examples, only the optimal design (obtained for the homogeneous distribution as a starting point and for a mesh size of $50 \times 50 \times 50$) is shown.

Concerning the existence of solutions, we run the same example for three different mesh sizes. The numerical simulations are shown in Figure 4. Judging from the numerical results, there exists clear numerical evidence that the design problem has optimal solutions.

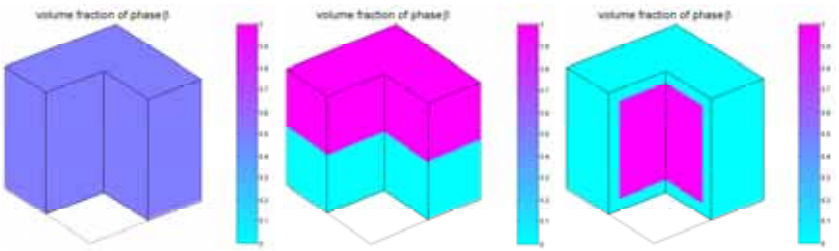
Now we run the same example by using different starting points, verifying the volume constraint (at starting point 1 there is a homogeneous distribution (Figure 5(a)), at 2 the β-material occupys the upper half of the body (Figure 5(b)), and at 3 the β-material is in a centered cube (Figure 5(c))). Judging from their respective numerical results (Figures 5(d), 5(e), and 5(f)), we can state that there exists clear numerical evidence about the uniqueness of the optimal solution. The convergence plots for the objective function and for the three starting points are shown in Figure 6.

In the second example we have taken $f(x) = e^{-100|x-\frac{1}{2}|^2}$, $V_0 = 0.6$, $\alpha = 1$, $\beta = 2$. As before, a half-section (Figure 7(a)) and the isosurfaces of temperature (Figure 7(b)) are included to understand the internal layout of the body and how the

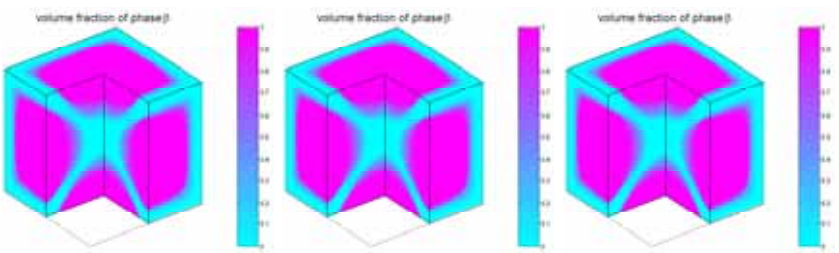


(a) $12 \times 12 \times 12$ (b) $25 \times 25 \times 25$ (c) $50 \times 50 \times 50$

FIG. 4. *Optimal solution for different mesh sizes of Example* 1.



(a) starting point 1 (b) starting point 2 (c) starting point 3



(d) optimal design (e) optimal design (f) optimal design

FIG. 5. *Optimal solution for different starting points of Example* 1.

optimal graded material must be built. Again, due to the symmetry of the example, the i-level indicates the i-slice in any characteristic direction (see Figures 7(c), 7(d), and 7(e)).

Now we see the influence of changing the relative volume fraction V_0 when heat is generated only in a centered cube; namely, the thermal load is given by

$$f(x) = \begin{cases} 1 & \text{if } x \in \left(\frac{1}{4}, \frac{3}{4}\right)^3, \\ 0 & \text{else.} \end{cases}$$

We maintain the same values in the material properties, $\alpha = 1$, $\beta = 2$. As we can ap-

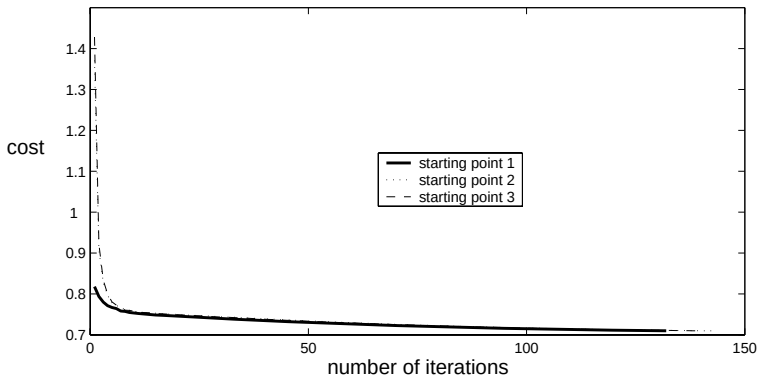
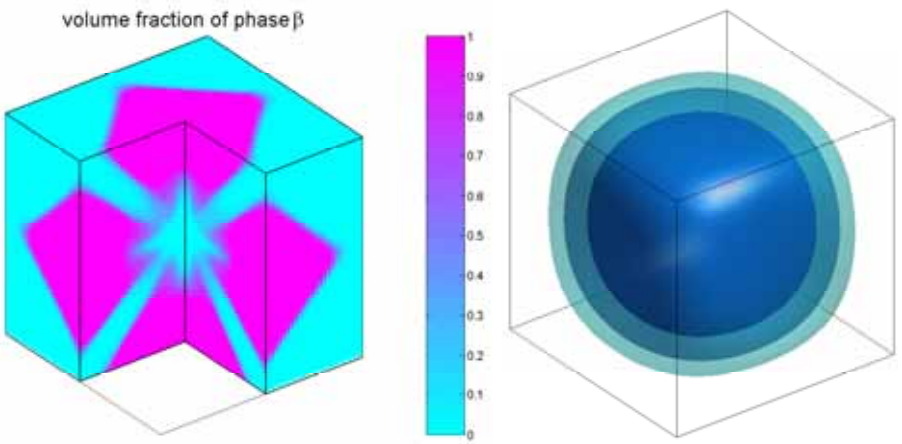
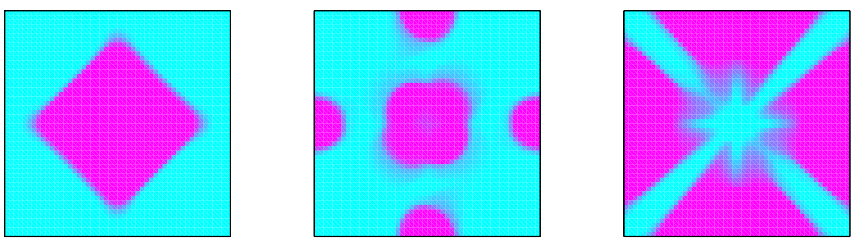


FIG. 6. *Objective function for different starting points of Example* 1.

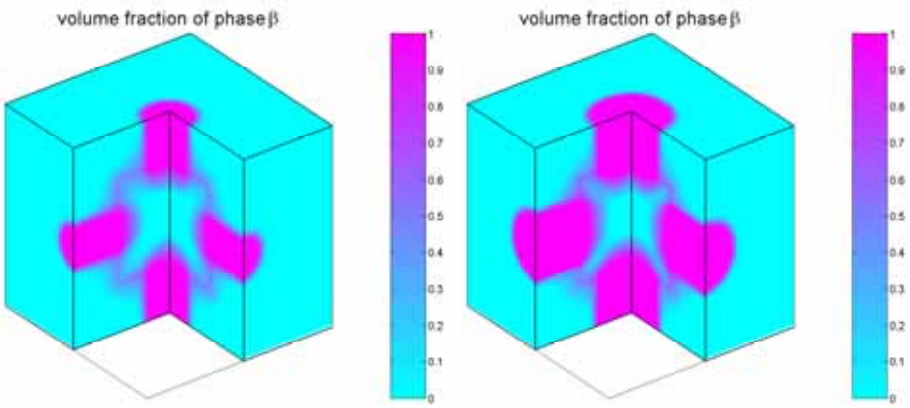


(a) Half-section (b) Isosurfaces temperature

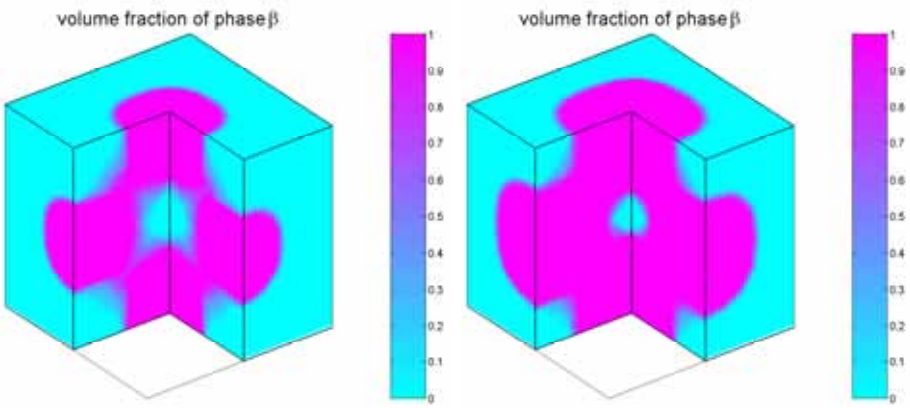


(c) 1-level (d) 12-level (e) 25-level

FIG. 7. *Optimal solution of Example* 2.



(a) $V_0 = 0.85$ (b) $V_0 = 0.75$

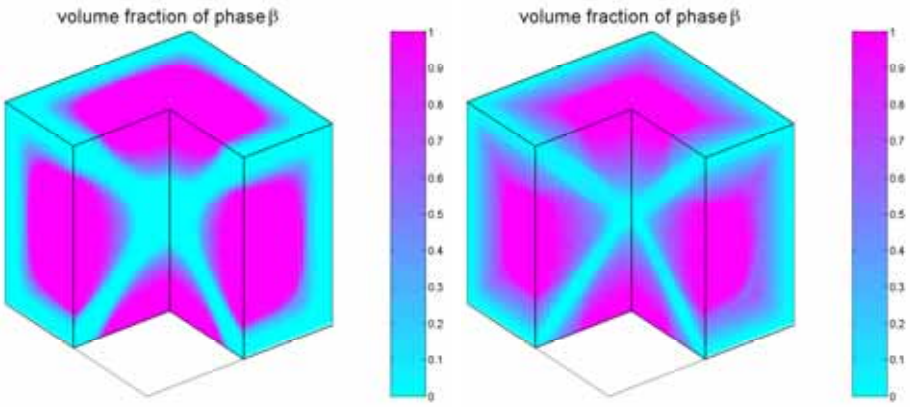


(c) $V_0 = 0.60$ (d) $V_0 = 0.40$

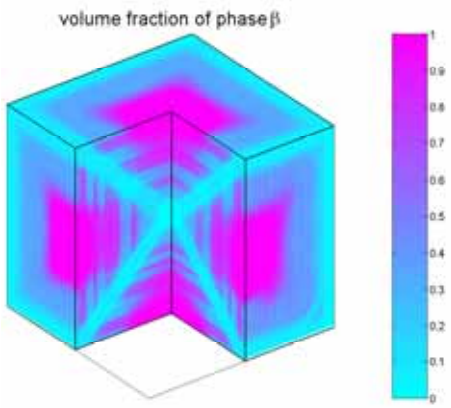
FIG. 8. *Optimal solution of Example* 3.

preciate for this particular example, less graded areas appear as the relative volume fraction decreases (see Figure 8).

Finally we see the influence of varying the contrast ratio (β/α). The source term is the same as that used in the first example. In this case, more graded areas appear as the ratio increases (see Figure 9).



(a) $(\beta/\alpha) = 2$ (b) $(\beta/\alpha) = 5$



(c) $(\beta/\alpha) = 10$

FIG. 9. *Optimal solution of Example* 4.

Acknowledgment. I would like to thank P. Pedregal for supervising this work.

REFERENCES

[1] G. ALLAIRE, *Shape Optimization by the Homogenization Method*, Springer, New York, 2002.
[2] S. AUBRY, *Etude théroique et numérique de quelques problèmes d'optimisation de forme à l'aide de méthodes d'homogénéisation*, Ph.D thesis, Paris 6 University, Paris, 1999.
[3] N. BANICHUK, *Introduction to Optimization of Structures*, Springer, New York, 1990.
[4] M. P. BENDSØE AND O. SIGMUND, *Topology Optimization: Theory, Methods and Applications*, Springer, Berlin, 2003.
[5] J. C. BELLIDO, *Explicit computation of the relaxed density coming from a three-dimensional optimal design problem*, Nonlinear Anal., 52 (2003), pp. 1709–1726.
[6] A. DONOSO AND P. PEDREGAL, *Optimal design of 2D conducting graded materials by minimizing quadratic functionals in the field*, Struct. Multidiscip. Optim., 30 (2005), pp. 360–367.
[7] Y. GRABOVSKY, *Optimal design problems for two-phase conducting composites with weakly discontinuous objective functionals*, Adv. in Appl. Math., 27 (2001), pp. 683–704.

[8] R. Lipton and A. Velo, *Optimal design of gradient fields with applications to electrostatics*, in Nonlinear Partial Differential Equations and Their Applications, College de France Seminar, D. Cioranescu, F. Murat, and J. L. Lions, eds., Stud. Math. Appl. 31, North–Holland, Amsterdam, 2002, pp. 509–532.

[9] F. Murat, *Contre-exemples pur divers problèmes où le contrôle intervient dans les coefficients*, Ann. Mat. Pura Appl. (4), 112 (1977), pp. 49–68.

[10] P. Pedregal, *Optimal design in* 2*-d conductivity for quadratic functionals in the field*, in Proceedings of the NATO Advanced Workshop in Nonlinear Homogenization, Warsaw, 2003.

[11] L. Tartar, *Remarks on optimal design problems*, in Calculus of Variations, Homogenization and Continuum Mechanics, G. Buttazzo, G. Bouchitte, and P. Suquet, eds., World Scientific, River Edge, NJ, 1994, pp. 279–296.

SIAM J. SCI. COMPUT.
Vol. 28, No. 3, pp. 942–962
© 2006 Society for Industrial and Applied Mathematics

ADAPTIVITY IN SPACE AND TIME FOR REACTION-DIFFUSION SYSTEMS IN ELECTROCARDIOLOGY*

PIERO COLLI FRANZONE†, PETER DEUFLHARD‡, BODO ERDMANN‡, JENS LANG§, AND LUCA F. PAVARINO¶

Abstract. The paper introduces and studies numerical methods that are fully adaptive in both three-dimensional (3D) space and time to challenging multiscale cardiac reaction-diffusion models. In these methods, temporal adaptivity comes via stepsize control in function space oriented linearly implicit time integration, while spatial adaptivity is realized within multilevel finite element methods controlled by a posteriori local error estimators. In contrast to other recent adaptivity approaches to cardiac modeling that discretize first in space and then in time (so-called method of lines), our method discretizes first in time and then in space (so-called Rothe method)—an approach that has already proven to be highly efficient in a number of challenging multiscale problems in science and technology (KARDOS code library). With this method, the evolution of a complete heartbeat, from the excitation to the recovery phase, is simulated both in the frame of the anisotropic monodomain models and in the more realistic anisotropic bidomain models, coupled with either a variant of the simple FitzHugh–Nagumo model or the more complex phase-I Luo–Rudy ionic model. The numerical results exhibit a rather satisfactory performance of our adaptive method for complex cardiac reaction-diffusion models on 3D domains up to moderate sizes. In particular, the method accurately resolves the evolution of the intra- and extracellular potentials, gating variables, and ion concentrations during the excitation, plateau, and recovery phases.

Key words. reaction-diffusion equations, cardiac bidomain and monodomain models, adaptive finite elements, multilevel methods, error estimates, adaptive time integration, adaptive Rothe method

AMS subject classifications. 65N55, 65M

DOI. 10.1137/050634785

1. Introduction.

Recent advances in contemporary cardiac electrophysiology are progressively revealing the complex multiscale structure of the bioelectrical activity of the heart, from the microscopic activity of ion channels of the cellular membrane to the macroscopic properties of the anisotropic propagation of excitation and recovery fronts in the whole heart; see, e.g., Zipes and Jalife [59], Kléber and Rudy [31], and Panfilov and Holden [43]. In recent years, the experimental study of these complex phenomena has been accompanied by mathematical and physical modeling and numerical simulation. The most complete of these models describes the cardiac tissue by a reaction-diffusion system of PDEs (the bidomain model), coupled with a system of ODEs (e.g., Luo–Rudy ionic models) modeling the ionic currents associated with the reaction terms.

Due to their inherent multiscale structure in both time and space, the numerical simulation of such models will be quite expensive. In space, meaningful portions of

*Received by the editors June 30, 2005; accepted for publication (in revised form) January 23, 2006; published electronically June 19, 2006.
http://www.siam.org/journals/sisc/28-3/63478.html
†Dipartimento di Matematica, Università di Pavia, Via Ferrata 1, 27100 Pavia, Italy (colli@imati.cnr.it).
‡Konrad-Zuse-Zentrum für Informationstechnik Berlin (ZIB), Takustr. 7, 14195 Berlin-Dahlem, Germany (deuflhard@zib.de, erdmann@zib.de).
§Technische Universität Darmstadt, Schlossgartenstr. 7, 64289 Darmstadt, Germany (lang@mathematik.tu-darmstadt.de).
¶Dipartimento di Matematica, Università di Milano, Via Saldini 50, 20133 Milano, Italy (pavarino@mat.unimi.it).

cardiac tissue have sizes in the cm regime, while the steep excitation front requires discretizations of the order of 0.1 mm—which results in a spatial scale spread factor of up to 10^3. In time, a normal heartbeat takes about 1 sec, while the time constants of the involved rapid kinetics range from 0.1 to 500 msec, requiring in some phases computational time steps of about 0.001 msec (or even less when currents or shocks are applied); we thus have a temporal scale spread factor of up to 10^6. Hence, realistic three-dimensional (3D) models employing , , grids will give rise to discrete problems with $\mathcal{O}(10^7)$ unknowns per time step—with simulations running over a huge number of time steps. As a remedy for the spatial spreading, algorithmic speedup by parallelism, domain decomposition, or adaptive mesh refinement is in the toolbox; for the time spreading, adaptive time-step control must be implemented. For large scale , , simulations, see, e.g., Colli Franzone and Pavarino [11], Pavarino and Colli Franzone [44], and Colli Franzone, Pavarino, and Taccardi [12]; for the use of some in parabolic reaction-diffusion systems, see, e.g., Cherry, Greenside, and Henriquez [8], Yu [57], and Trangenstein and Kim [53] in two space dimensions and Moore [40] or Lines, Grøttum, and Tveito [37] in two and three dimensions. All of these papers use a (discretization first in space and then in time). In contrast, the present authors have strong arguments, both theoretical and practical, to favor a (discretization first in time and then in space) within a multilevel finite element setting. As for the Rothe approach, see the first pioneering paper [6] by Bornemann for linear parabolic equations and its generalization to realistic nonlinear parabolic systems of PDEs by Lang [32], where the efficient solution of an impressive number of challenging problems from science and technology has also been documented. As for a detailed computational comparison between both approaches (including adaptivity), see, e.g., the survey paper [17] by Deuflhard, Lang, and Nowak. It is the purpose of the present paper to apply this elsewhere rather successful kind of method to the challenging class of cardiac models. For the mere purpose of just evaluating the algorithms in this special PDE context, we decided to start with quite simple 3D geometries. Clearly, once the method has proven to be useful, we will turn to real-world geometries from medical imaging input—see, e.g., work by the ZIB group in other fields of computational medicine [19, 33, 21, 58].

We consider both the anisotropic monodomain and bidomain models, coupled with either a variant of the simple FitzHugh–Nagumo model or the more complex phase-I Luo–Rudy ionic model. We simulate the evolution of a complete heartbeat, from the excitation (depolarization) phase to the following plateau and recovery (repolarization) phases. Our simulations are performed with the KARDOS code library [20, 2] that realizes the above Rothe method in an adaptive multilevel finite element setting. Note that most simulations in computational electrocardiology are based on the simpler monodomain model, while here we also consider the full bidomain system in three dimensions. The few other bidomain simulations presented in the literature often employ simplifications of the model, such as two-dimensional (2D) domains, simpler ionic models, operator splitting of the elliptic-parabolic formulation of the bidomain system, coarser time or space mesh size for the elliptic subproblem, etc.; out of the bulk of literature we mention Colli Franzone, Guerri, and Rovida [10], Colli Franzone and Guerri [9], Roth [48], Hooke et al. [28], Pormann [46], Henriquez [25], Henriquez, Muzikant, and Smoak [26], and Muzikant et al. [42], Vigmond, Aguel, and Trayanova [55], Sambelashvili and Efimov [49], Ashihara et al. [3], Dos Santos et al. [56], Murillo and Cai [41], Pennacchio and Simoncini [45], and Sundnes et al. [52].

The paper is organized as follows. In section 2, we introduce the standard model of cardiac tissue, including its fiber structure and conductivity tensors, the anisotropic

bidomain model for the intra- and extracellular potentials, and the simpler anisotropic monodomain model. Moreover, the basic ideas of cellular ionic models of Hodgkin–Huxley type are reviewed, focusing on a FitzHugh–Nagumo variant and the phase-I Luo–Rudy model that will be employed in our cardiac model. In section 3, we describe the adaptive numerical discretization of our models as realized within the KARDOS code library, focusing on the adaptive time discretization by linearly implicit methods and the adaptive multilevel finite elements employed in the space discretization. Finally, in section 4, we present the results of several numerical experiments in three space dimensions and discuss the gain due to adaptivity. All 3D visualizations in this paper have been created using Amira—a system for advanced visual data analysis; see [1, 50].

2. Cardiac reaction-diffusion models. In this section, we collect details about the cardiac models, including the membrane models as far as we have chosen them for our simulations.

2.1. Cardiac fibers and conductivity tensors. The cardiac ventricular tissue can be modeled as an arrangement of cardiac fibers which rotate counterclockwise from epi- to endocardium (Streeter [51]) and which has a laminar organization modeled as a set of muscle sheets running radially from epi- to endocardium (LeGrice, Smaill, and Hunter [36]). Therefore, at every point $\mathbf{x}$ it is possible to identify a triplet of orthonormal principal axes $\mathbf{a}_l(\mathbf{x})$, $\mathbf{a}_t(\mathbf{x})$, $\mathbf{a}_n(\mathbf{x})$, with $\mathbf{a}_l(\mathbf{x})$ parallel to the local fiber direction, $\mathbf{a}_t(\mathbf{x})$ and $\mathbf{a}_n(\mathbf{x})$ tangent and orthogonal to the radial laminae, respectively, and both being transversal to the fiber axis.

The macroscopic bidomain model represents the cardiac tissue as the superposition of two anisotropic continuous media, the intra- (i) and extra- (e) cellular media, coexisting at every point of the tissue and separated by a distributed continuous cellular membrane; see, e.g., Henriquez [25] and Keener and Sneyd [30]. Denoting by $\sigma_l^{i,e}$, $\sigma_t^{i,e}$, $\sigma_n^{i,e}$ the conductivity coefficients in the intra- and extracellular media measured along the corresponding directions $\mathbf{a}_l, \mathbf{a}_t, \mathbf{a}_n$, the anisotropic conductivity tensors $D_i(\mathbf{x})$ and $D_e(\mathbf{x})$ related to [illegible] of the media are given by

$$D_{i,e}(\mathbf{x}) = \sigma_l^{i,e}\, \mathbf{a}_l(\mathbf{x})\mathbf{a}_l^T(\mathbf{x}) + \sigma_t^{i,e}\, \mathbf{a}_t(\mathbf{x})\mathbf{a}_t^T(\mathbf{x}) + \sigma_n^{i,e}\, \mathbf{a}_n(\mathbf{x})\mathbf{a}_n^T(\mathbf{x}). \tag{2.1}$$

For [illegible] anisotropic media, $\sigma_n^{i,e} = \sigma_t^{i,e}$, we have

$$D_{i,e}(\mathbf{x}) = {\sigma_t}^{i,e} I + ({\sigma_l}^{i,e} - {\sigma_t}^{i,e})\mathbf{a}_l(\mathbf{x})\mathbf{a}_l^T(\mathbf{x}).$$

2.2. Anisotropic bidomain (B) model. The intra- and extracellular electric potentials u_i, u_e in the anisotropic bidomain model are described by a reaction-diffusion system coupled with a system of ODEs for ionic gating variables w and for ion concentrations c. We denote by $v = u_i - u_e$ the transmembrane potential and by

$$I_m = c_m \frac{\partial v}{\partial t} + I_{ion}(v, w, c)$$

the membrane current per unit volume, where $c_m = \chi * C_m$, $I_{ion} = \chi * i_{ion}$, with χ the ratio of membrane area per tissue volume, C_m the surface capacitance, and i_{ion} the ionic current of the membrane per unit area. Let I_{app}^e be an applied extracellular current per unit volume, satisfying the compatibility condition $\int_\Omega I_{app}^e \,\mathrm{dx} = 0$, and $\mathbf{j}_{i,e} = -D_{i,e}\nabla u_{i,e}$ the intra- and extracellular current density. Due to the current conservation law, we have

$$\operatorname{div} \mathbf{j}_i = -I_m, \qquad \operatorname{div} \mathbf{j}_e = I_m - I_{app}^e.$$

Then the anisotropic bidomain model in the unknown potentials $(u_i(\mathbf{x},t), u_e(\mathbf{x},t))$ and $v(\mathbf{x},t) = u_i(\mathbf{x},t) - u_e(\mathbf{x},t)$, gating variables $w(\mathbf{x},t)$, and ion concentrations $c(\mathbf{x},t)$ can be written as

$$
\begin{aligned}
& c_m \partial_t v - \operatorname{div}(D_i \nabla u_i) + I_{ion}(v,w,c) = 0 && \text{in } \Omega \times (0,T),\\
& -c_m \partial_t v - \operatorname{div}(D_e \nabla u_e) - I_{ion}(v,w,c) = -I^e_{app} && \text{in } \Omega \times (0,T),\\
& \partial_t w - R(v,w) = 0, \quad \partial_t c - S(v,w,c) = 0 && \text{in } \Omega \times (0,T),\\
& \mathbf{n}^T D_{i,e} \nabla u_{i,e} = 0 && \text{in } \partial\Omega \times (0,T),\\
& v(\mathbf{x},0) = v_0(\mathbf{x}), \quad w(\mathbf{x},0) = w_0(\mathbf{x}), \quad c(\mathbf{x},0) = c_0(\mathbf{x}) && \text{in } \Omega,
\end{aligned}
\tag{2.2}
$$

where we have imposed insulated boundary conditions. The system uniquely determines v, while the potentials u_i and u_e are defined only up to a same additive time-dependent constant relating to the reference potential. This potential is chosen to be the average extracellular potential in the cardiac volume by imposing $\int_\Omega u_e \, \mathrm{dx} = 0$. We refer the reader to [13] for a mathematical analysis of the bidomain model. In the following, we will consider as ionic membrane models some Luo–Rudy models which are briefly recalled in section 2.4; (see Luo and Rudy [39]). These models specify the reaction term I_{ion} and the functions $R(v,w)$ and $S(v,w,c)$ determining the evolution of the gating variables w and the ion concentrations c.

In terms of the transmembrane and extracellular potentials $v(\mathbf{x},t)$ and $u_e(\mathbf{x},t)$, we obtain the following equivalent formulation of the reaction-diffusion equations of the anisotropic bidomain model:

$$
\begin{aligned}
& c_m \partial_t v + I_{ion}(v,w,c) + \operatorname{div}(D_e \nabla u_e) = I^e_{app} && \text{in } \Omega \times (0,T),\\
& -\operatorname{div}((D_i + D_e)\nabla u_e) = \operatorname{div}(D_i \nabla v) - I^e_{app} && \text{in } \Omega \times (0,T).
\end{aligned}
\tag{2.3}
$$

2.3. Anisotropic monodomain (M) model. If we assume equal anisotropy ratio of the two media, it is well known that the bidomain system reduces to the monodomain model. We shall mention here another interesting derivation of a reduced bidomain model which does not make such an assumption and that we will still call monodomain model. Denoting by $J_{tot} = \mathbf{j}_i + \mathbf{j}_e$ the total current flowing in the two media, since $J_{tot} = -D_i \nabla u_i - D_e \nabla u_e$, substituting $u_i = v + u_e$, we get

$$
\nabla u_e = -D^{-1} D_i \nabla v - D^{-1} J_{tot}, \tag{2.4}
$$

with $D = D_i + D_e$. Therefore, the first equation in the bidomain system can be written as

$$
-c_m \partial_t v + \operatorname{div}(D_e D^{-1} D_i \nabla v) + \operatorname{div}(D_e D^{-1} J_{tot}) - I_{ion}(v,w) = -I^e_{app}. \tag{2.5}
$$

Since the tensors (2.1) can be written as

$$
D_{i,e}(\mathbf{x}) = \sigma_l^{i,e} I + (\sigma_t^{i,e} - \sigma_l^{i,e}) \mathbf{a}_t(\mathbf{x}) \mathbf{a}_t^T(\mathbf{x}) + (\sigma_n^{i,e} - \sigma_l^{i,e}) \mathbf{a}_n(\mathbf{x}) \mathbf{a}_n^T(\mathbf{x}),
$$

then

$$
D_e D^{-1} = \mu_l^e I + (\mu_t^e - \mu_l^e) \mathbf{a}_t(\mathbf{x}) \mathbf{a}_t^T(\mathbf{x}) + (\mu_n^e - \mu_l^e) \mathbf{a}_n(\mathbf{x}) \mathbf{a}_n^T(\mathbf{x}), \tag{2.6}
$$

with $\mu^e_{l,t,n} = \sigma^e_{l,t,n}/(\sigma^e_{l,t,n} + \sigma^i_{l,t,n})$. Assuming constant conductivity coefficients and taking into account that $\operatorname{div} J_{tot} = -I^e_{app}$, we have

$$\begin{aligned}\operatorname{div}(D_e D^{-1} J_{tot}) &= \mu^e_l \operatorname{div} J_{tot} + (\mu^e_t - \mu^e_l)\operatorname{div}(\mathbf{a}_t(\mathbf{x})\mathbf{a}^T_t(\mathbf{x})J_{tot}) \\ &\quad + (\mu^e_n - \mu^e_l)\operatorname{div}(\mathbf{a}_n(\mathbf{x})\mathbf{a}^T_n(\mathbf{x})J_{tot}) \\ &= -\mu^e_l I^e_{app} + (\mu^e_t - \mu^e_l)\operatorname{div}(\mathbf{a}_t(\mathbf{x})\mathbf{a}^T_t(\mathbf{x})J_{tot}) \\ &\quad + (\mu^e_n - \mu^e_l)\operatorname{div}(\mathbf{a}_n(\mathbf{x})\mathbf{a}^T_n(\mathbf{x})J_{tot}.\end{aligned} \tag{2.7}$$

From (2.4) it follows that $-D_e D^{-1} D_i \nabla v = D_e D^{-1} J_{tot} + D_e \nabla u_e$, and hence the flux relationship

$$\mathbf{n}^T D_e D^{-1} D_i \nabla v = \mathbf{n}^T D_e D^{-1} J_{tot} + \mathbf{n}^T D_e \nabla u_e. \tag{2.8}$$

Using (2.6), the first term on the right-hand side can be written as

$$\mathbf{n}^T (D_e D^{-1} J_{tot}) = \mu^e_l \mathbf{n}^T J_{tot} + (\mu^e_t - \mu^e_l)(\mathbf{n}^T \mathbf{a}_t)(\mathbf{a}^T_t J_{tot}) + (\mu^e_\nu - \mu^e_l)(\mathbf{n}^T \mathbf{a}_\nu)(\mathbf{a}^T_\nu J_{tot}).$$

The insulating conditions $\mathbf{n}^T \mathbf{j}_i = \mathbf{n}^T \mathbf{j}_e = 0$ imply $\mathbf{n}^T J_{tot} = 0$, i.e., J_{tot} is tangent to $\partial\Omega$, and assuming that the fibers are also tangent to $\partial\Omega$, we have $\mathbf{n}^T \mathbf{a}_\nu = 0$ and $\mathbf{a}^T_t J_{tot} = 0$. Substituting these conditions into (2.8) it follows the monodomain boundary condition

$$\mathbf{n}^T D_e D^{-1} D_i \nabla v = 0. \tag{2.9}$$

Disregarding the two additional terms in (2.7) related to the projections of J_{tot} onto the directions across fiber $\mathbf{a}_t$ and $\mathbf{a}_n$ (which disappear for media having equal anisotropic ratio $\mu^e_l = \mu^e_t = \mu^e_n$), it results that $\operatorname{div}(D_e D^{-1} J_{tot}) \approx -\mu^e_l I^e_{app}$. Substituting this approximation into (2.5) and considering the boundary condition (2.9), we obtain the anisotropic monodomain model consisting of a single parabolic reaction-diffusion equation for v with the conductivity tensor $D_m = D_e D^{-1} D_i$, $I^m_{app} = I^e_{app}\sigma^i_l/(\sigma^e_l + \sigma^i_l)$ and coupled with the same gating system

$$\begin{aligned}&c_m \partial_t v - \operatorname{div}(D_m \nabla v) + I_{ion}(v, w, c) = I^m_{app} && \text{in } \Omega \times (0,T), \\ &\partial_t w - R(v, w) = 0, \quad \partial_t c - S(v, w, c) = 0 && \text{in } \Omega \times (0,T), \\ &\mathbf{n}^T D_m \nabla v = 0 && \text{in } \partial\Omega \times (0,T), \\ &v(\mathbf{x}, 0) = v_0(\mathbf{x}), \quad w(\mathbf{x}, 0) = w_0(\mathbf{x}), \quad c(\mathbf{x}, 0) = c_0(\mathbf{x}) && \text{in } \Omega.\end{aligned} \tag{2.10}$$

From the knowledge of the distribution of $v(\mathbf{x}, t)$ the extracellular potential distribution u_e is derived by solving the elliptic boundary value problem

$$-\operatorname{div}(D\nabla u_e) = \operatorname{div}(D_i \nabla v), \qquad -\mathbf{n}^T D \nabla u_e = \mathbf{n}^T D_i \nabla v.$$

2.4. Membrane models and ionic currents. Following the celebrated work by Hodgkin and Huxley [27] (1963 Nobel Prize in Medicine) on nerve action potential, many models of Hodgkin–Huxley type have later been developed for the cardiac action potential. In these models, the ionic current through channels of the membrane, due to the transmembrane potential v and the M gating and Q ionic concentration variables $w := (w_1, \ldots, w_M), c := (c_1, \ldots, c_Q)$, is given by

$$I_{ion}(v, w, c) = \sum_{k=1}^{N} G_k(v, c) \prod_{j=1}^{M} w_j^{p_{j_k}} (v - v_k(c)) + H(v, c),$$

where $G_k(v,c)$ is the membrane conductance, $H(v,c)$ accounts for pump and exchanger currents, v_k is the Nernst equilibrium potential for the kth current, and p_{j_k} are integers. The dynamics of the gating and concentration variables is described by a system of ODEs as in (2.2); when w_j is a gating variable ($0 \le w_j \le 1$) these equations have the form

$$\partial_t w_j = \alpha_j(v)(1-w_j) - \beta_j(v) w_j, \tag{2.11}$$

with coefficients $\alpha_j > 0$ and $\beta_j > 0$ and appropriate initial conditions.

Many refinements of the original Hodgkin–Huxley model have been proposed by fitting improved experimental data with more complex models; for example, we recall here the following models: Beeler–Reuter (1977, N = 4, M = 6, Q = 1), phase-I Luo–Rudy (1991, N = 6, M = 6, Q = 1), phase-II Luo–Rudy (1994, N = 10, M = 6, Q = 5); see Hund and Rudy [29] for current developments.

Simplified models of lower complexity (with 1 or 2 gating variables) have been proposed too. The simplest and most used is the FitzHugh–Nagumo (FHN) model (N = 1, M = 1), which yields only a coarse approximation of a typical cardiac action potential, particularly in the plateau and repolarization phases. A better approximation is given by the following FHN variant by Rogers and McCulloch [47]:

$$\begin{aligned} I_{ion}(v,w) &= Gv(1-v/v_{th})(1-v/v_p) + \eta_1 v w, \\ \partial_t w &= \eta_2 \left(v/v_p - \eta_3 w\right), \end{aligned}$$

where $G, \eta_1, \eta_2, \eta_3$ are positive real coefficients, v_{th} is a threshold potential, and v_p is the peak potential. The gating variable w satisfies

$$\partial_t w = \eta v - \gamma w,$$

with $\eta, \gamma > 0$. We will consider this FHN variant as our simplest gating model.

In this paper, we will also consider the more detailed phase-I Luo–Rudy (LR1) model (see [39]), where I_{ion} is the sum of $N = 6$ currents

$$I_{ion} = I_{Na} + I_{si} + I_K + I_{K1} + I_{Kp} + I_b,$$

two inwards (I_{Na} and I_{si}) and four outwards (I_K, I_{K1}, I_{Kp}, I_b). The first three currents depend on $M = 6$ gating variables satisfying differential equations of the form (2.11) and $Q = 1$ ion (calcium) concentration c satisfying a different differential equation. The last three currents are time-independent. For the detailed equations of this model, we refer the reader to [39].

3. Adaptive Rothe method (code KARDOS). The code family KARDOS [20, 2] is characterized by a combination of discretization in time with finite elements in space in the setting of a , i.e., first time discretization and then spatial discretization. In this setting, both time-step control and dynamic mesh refinement on the basis of a posteriori error estimation can be simultaneously realized. For linear parabolic scalar PDEs, this so-called adaptive Rothe method has been suggested, theoretically investigated, and worked out in algorithmic detail by Bornemann [6]. On this basis, Lang and Walter [35] have generalized the approach to systems of reaction-diffusion equations. A rigorous analysis for nonlinear parabolic systems has been given by Lang [32], where challenging examples

from other fields of science and technology are also included. For a detailed comparison with an adaptive method of lines (first in space and then time discretization), we refer the reader to Deuflhard, Lang, and Nowak [17]. In what follows, we want to collect the main ingredients of the adaptive Rothe method as needed for the above described anisotropic monodomain and bidomain cardiac models.

Throughout this section, our reaction-diffusion problems will be written as

$$B(x,t,u,\nabla u)\partial_t u = \nabla \cdot (D(x,t,u,\nabla u)\nabla u) + F(x,t,u,\nabla u), \tag{3.1}$$

supplemented with suitable boundary and initial conditions. For simplicity of the presentation, we assume that the vector-valued solution $u = (u_1, \ldots, u_m)^T$ is unique.

3.1. Linearly implicit methods. Mathematically speaking, the above cardiac models are nonlinear parabolic PDEs, with the bidomain model even degenerate. The additional mildly stiff ODEs stemming from the membrane models (FHN or LR1) are naturally treated within the frame of the Rothe approach. This means that after discretization we are faced with a large scale [illegible] or even [illegible] system. From an ODE point of view, an optimal treatment of this stiffness structure would mean to apply some L-stable implicit time discretization; for general reference, see the textbooks by Hairer and Wanner [24] or by Deuflhard and Bornemann [16]. From the PDE point of view, however, the avoidance of order reduction (which occurs above order 2) is even more important for the overall efficiency of the time integrator. As shown in [34], the time discretization behind the code ROS3P is a good compromise which avoids order reduction, exhibits [illegible] A [illegible] (with $R(\infty) \sim 0.7$ instead of $R(\infty) = 0$ as for L-stability), and damps higher frequency oscillations to a sufficient extent. Note that the popular Crank–Nicolson scheme is not strongly A-stable (it goes with $R(\infty) = -1$), which results in an insufficient filtering of spurious modes. Fully implicit schemes require the iterative solution of finite dimensional [illegible] systems of algebraic equations by some Newton-like method. In contrast, as pointed out by Deuflhard [14, 15], linearly implicit methods realize a (simplified) Newton method in function space and require only one or a few [illegible] systems per time step. The code KARDOS contains linearly implicit methods of Rosenbrock type. According to their one-step nature, they allow for a rapid change of step sizes and an efficient adaptation of the spatial discretization in each time step. Moreover, a simple embedding technique can be used to estimate the error part arising from time discretization.

In order to give a rough idea of this kind of time discretization, we set $B = I$ in (3.1) and consider only the autonomous case. In the Rothe approach, we look at (3.1) as an abstract Cauchy problem of the form

$$\partial_t u = f(u), \quad u(t_0) = u_0, \quad t > t_0, \tag{3.2}$$

where the differential operators and the boundary conditions are incorporated into the nonlinear function $f(u)$. Its time discretization by an s-stage Rosenbrock method has the recursive form

$$(I - \tau\gamma_{ii}\, J_n)\, K_{ni} = \tau f\left(u_n + \sum_{j=1}^{i-1} \alpha_{ij}\, K_{nj}\right) + \tau J_n \sum_{j=1}^{i-1} \gamma_{ij}\, K_{nj},\ i = 1(1)s, \tag{3.3}$$

$$u_{n+1} = u_n + \sum_{i=1}^{s} b_i K_{ni}, \tag{3.4}$$

where the step number s and the defining formula coefficients b_i, α_{ij}, and γ_{ij} are chosen such that a desired order of consistency and good stability properties for stiff equations (see, e.g., [24, Chapter IV.7]) are obtained. We assume $\gamma_{ii}=\gamma>0$ for all i, which is the standard simplification to derive Rosenbrock methods with one and the same operator on the left-hand side of (3.3). The simplest candidate of this class of methods is the linearly implicit Euler method ($s=1$ and $\gamma=1$).

For the general system

$$B(t,u)\partial_t u = f(t,u), \quad u(t_0)=u_0, \quad t>t_0, \tag{3.5}$$

an efficient implementation that avoids matrix-vector multiplications with the Jacobian was given by Lubich and Roche [38]. In the case of a time- or solution-dependent matrix B, an approximation of $\partial_t u$ has to be taken into account, leading to the generalized Rosenbrock method of the form

$$\begin{aligned}\left(\frac{1}{\tau\gamma}B(t_n,u_n)-J_n\right)U_{ni} &= f(t_i,U_i)-B(t_n,u_n)\sum_{j=1}^{i-1}\frac{c_{ij}}{\tau}U_{nj}+\tau\gamma_i C_n \\ &\quad + \left(B(t_n,u_n)-B(t_i,U_i)\right)Z_i, \quad i=1(1)s,\end{aligned} \tag{3.6}$$

where the internal values are given by

$$t_i=t_n+\alpha_i\tau, \quad U_i=u_n+\sum_{j=1}^{i-1}a_{ij}\,U_{nj}, \quad Z_i=(1-\sigma_i)z_n+\sum_{j=1}^{i-1}\frac{s_{ij}}{\tau}\,U_{nj},$$

and the Jacobians are defined by

$$\begin{aligned}J_n &:= \partial_u\left(f(t,u)-B(t,u)z\right)\big|_{u=u_n,t=t_n,z=z_n},\\ C_n &:= \partial_t\left(f(t,u)-B(t,u)z\right)\big|_{u=u_n,t=t_n,z=z_n}.\end{aligned}$$

This yields the new solution

$$u_{n+1}=u_n+\sum_{i=1}^{s}m_i\,U_{ni}$$

and an approximation of the temporal derivative $\partial_t u$

$$z_{n+1}=z_n+\sum_{i=1}^{s}m_i\left(\frac{1}{\tau}\sum_{j=1}^{i}(c_{ij}-s_{ij})U_{nj}+(\sigma_i-1)z_n\right).$$

The new coefficients can be derived from α_{ij}, γ_{ij}, and b_i [38]. In the special case $B(t,u)=I$, we get (3.3) by setting $U_{ni}=\tau\sum_{j=1,\dots,i}\gamma_{ij}K_{nj}$, $i=1,\dots,s$.

Among the many Rosenbrock solvers suitable to integrate index-1 systems of the form (3.5) we selected the code ROS3P that has been constructed recently by Lang and Verwer [34]. This choice had also turned out to be favorable in preliminary tests on the monodomain model.

For linearly implicit methods of Rosenbrock type a second solution of inferior order, say $\hat{p}$, can be computed by a so-called embedded formula

$$\hat{u}_{n+1} = u_n + \sum_{i=1}^{s} \hat{m}_i U_{ni},$$

$$\hat{z}_{n+1} = z_n + \sum_{i=1}^{s} \hat{m}_i \left(\frac{1}{\tau} \sum_{j=1}^{i} (c_{ij} - s_{ij}) U_{nj} + (\sigma_i - 1) z_n \right),$$

where the original weights m_i are simply replaced by $\hat{m}_i$. If p is the order of u_{n+1}, we call such a pair of formulas to be of order $p(\hat{p})$. Introducing an appropriately scaled norm $\|\cdot\|$, the local error estimator

$$r_{n+1} = \|u_{n+1} - \hat{u}_{n+1}\| + \|\tau(z_{n+1} - \hat{z}_{n+1})\| \tag{3.7}$$

can be used to propose a new time step by

$$\tau_{n+1} = \frac{\tau_n}{\tau_{n-1}} \left(\frac{TOL_t \, r_n}{r_{n+1} \, r_{n+1}} \right)^{1/(\hat{p}+1)} \tau_n. \tag{3.8}$$

Here TOL_t is a desired tolerance prescribed by the user. This formula is related to a discrete PI controller first established in the pioneering papers [23, 22] by Gustafsson, Lundh, and Söderlind.

Note that the above time discretization scheme has been applied to the abstract Cauchy problem (3.2), i.e., to the initial value problem in function space. This means that the Rosenbrock discretization scheme (3.3) is a sequence of linear elliptic PDE boundary value problems. The evaluation of the terms used in the time-step control (3.8) is done via a finite element approximation to be discussed next.

3.2. Multilevel finite elements. In section 4, we show only results for rather simple geometries, which would well permit the use of spectral methods for spatial discretization. However, since the whole scheme is intended to work for realistic geometries of individual patients' hearts, a choice of linear finite elements within the multilevel scheme seems to be in order. The main idea of multilevel techniques consists of replacing the solution space, an infinite dimensional function space, by a (nested) sequence of finite dimensional spaces with successively increasing dimension—see, e.g., Bank and Smith [5], Deuflhard, Leinen, and Yserentant [18], and Bornemann, Erdmann, and Kornhuber [7].

For the construction of the adaptive multilevel mesh, the edge-oriented hierarchical error estimator from [18] is used in the form extended to Rosenbrock schemes in [32]. Such estimators are well established to control the adaptive multilevel process producing successively finer meshes and, in spatial multiscale cases, drastically reducing the size of the arising linear algebraic systems with respect to a prescribed tolerance.

For a more formal description, let T_h be an admissible finite element mesh at $t = t_n$ and S_h^q be the associated finite dimensional space consisting of all continuous functions which are polynomials of order q on each finite element $T \in T_h$. Then the standard Galerkin finite element approximation $U_{ni}^h \in S_h^q$ of the intermediate values U_{ni} satisfies the equation

$$(L_n \, U_{ni}^h, \phi) = (r_{ni}, \phi) \quad \text{for all } \phi \in S_h^q, \tag{3.9}$$

where L_n is the weak representation of the differential operator on the left-hand side in (3.6) and r_{ni} stands for the entire right-hand side in (3.6). Since the operator L_n is independent of i, its calculation is required only once within each time step.

The resulting linear systems can be solved by either direct or iterative methods. While direct methods work quite satisfactorily in 1D and even 2D applications, iterative solvers such as Krylov subspace methods perform considerably better with respect to CPU time and memory requirements for large 2D and 3D problems. For the problem class in question here, i.e., the cardiac models, we mainly used the BI-CGSTAB algorithm [54] with ILU preconditioning.

After computing the approximate intermediate values U_{ni}^h, a posteriori error estimates can be used to give specific assessment of the error distribution. Consider a hierarchical decomposition

$$S_h^{q+1} = S_h^q \oplus Z_h^{q+1}, \tag{3.10}$$

where Z_h^{q+1} is the subspace that corresponds to the span of additional basis functions, typically of higher order. Defining an a posteriori error estimator $E_{n+1}^h \in Z_h^{q+1}$ by

$$E_{n+1}^h = E_{n0}^h + \sum_{i=1}^{s} m_i E_{ni}^h, \tag{3.11}$$

with E_{n0}^h approximating the projection error of the initial value u_n in Z_h^{q+1} and E_{ni}^h estimating the spatial error of the intermediate value U_{ni}^h, the local spatial error for a finite element $T \in T_h$ can be estimated by $\eta_T := \|E_{n+1}^h\|_T$. The error estimator E_{n+1}^h is computed by linear systems which can be derived from (3.9). We get for $i = 1, \ldots, s$

$$(L_n\, E_{ni}^h, \phi) = (r_{ni}(E_{n1}^h + U_{n1}^h, \ldots, E_{n,i-1}^h + U_{n,i-1}^h), \phi) - (L_n\, U_{ni}^h, \phi) \quad \text{for all } \phi \in Z_h^{q+1}.$$

With linear finite elements as approximation space, quadratic bubble functions living on edges are used for local error estimation. The scaling of the solution components is specified in [32, page 66]. The whole process of multilevel mesh adaptation is controlled by a local extrapolation technique due to Babuška and Rheinboldt [4].

The authors are well aware of the fact that in the field of numerical cardiac modeling other spatial adaptivity techniques are also in use, but none of them use a posteriori error estimators as the ones we advocate here. As an example, Tveito and his group use wave front tracking [37], a method well known from [illegible] PDEs and based on monitoring local gradients (or a finite difference approximation of them). Such techniques may be less costly (depending on the necessary overhead for the mesh data structure) but are typically regarded as less reliable in nonlinear [illegible] PDEs.

4. Numerical simulations. In order to study the performance of our adaptive Rothe method, we have performed various simulations on the simple 3D test domain $\Omega = [1, 2] \times [1, 2] \times [1, 1.1]$ of size $1 * 1 * 0.1$ cm^3, including both the monodomain (M) and the bidomain (B) model. Results are presented for the FHN and the LR1 ionic models. The parameters used in our computations are listed in Table 4.1—for details compare to section 2 above. The fibers lie in the (x, y)-plane and have constant direction, forming a $-45°$ angle with the [illegible]-axis.

TABLE 4.1
Parameters chosen in the cardiac (M, B) and membrane (FHN, LR1) models.

General	$\chi = 10^3\ \mathrm{cm}^{-1},\ \ C_m = 10^{-3}\ mF/\mathrm{cm}^2$
M	$\sigma_l = 1.2 \cdot 10^{-3}\ \Omega^{-1}\mathrm{cm}^{-1},\ \ \sigma_t = 2.5562 \cdot 10^{-4}\ \Omega^{-1}\mathrm{cm}^{-1}$
B	$\sigma_l^e = 2 \cdot 10^{-3}\ \Omega^{-1}\mathrm{cm}^{-1},\ \ \sigma_l^i = 3 \cdot 10^{-3}\ \Omega^{-1}\mathrm{cm}^{-1}$ $\sigma_t^e = 1.3514 \cdot 10^{-3}\ \Omega^{-1}\mathrm{cm}^{-1}$ $\sigma_t^i = 3.1525 \cdot 10^{-4}\ \Omega^{-1}\mathrm{cm}^{-1}$ $\sigma_n^e = \sigma_t^e/\mu_1,\ \ \sigma_n^i = \sigma_t^i/\mu_2$ $\mu_1 = \mu_2 = 1$ axial isotropic case $\mu_1 = 2,\ \mu_2 = 10$ orthotropic case
FHN	$G = 1.5\ \Omega^{-1}\mathrm{cm}^{-2},\ \ v_{th} = 13\ mV,\ \ v_p = 100\ mV$ $\eta_1 = 4.4\ \Omega^{-1}\mathrm{cm}^{-2},\ \ \eta_2 = 0.012,\ \ \eta_3 = 1$
LR1	as in original paper [39] except G_{si} reduced by a factor 2/3

(3.5) For the (M) we set

$$B(t,u) = \begin{pmatrix} c_m & 0 \\ 0 & I \end{pmatrix}, \qquad f(t,u) = \begin{pmatrix} I_{app} - I_{ion}(u) + \nabla \cdot (D_m \nabla u_1) \\ P(u) \end{pmatrix},$$

where

$$u = (v, w)^T, \quad I = 1, \quad P = \eta_2 \left(\frac{v}{v_p} - \eta_3 w \right)$$

for the ( ) and

$$u = (v, h, j, m, d, f, X, [C_a]_i)^T, \quad I = \mathrm{diag}(1,1,1,1,1,1,1,1),$$
$$P = (R_1, \ldots, R_6, S)^T, \quad R_k = \alpha_k(u_1)(1 - u_{k+1}) - \beta_k(u_1) u_{k+1}, \quad k = 1, \ldots, 6,$$
$$S = 0.07(10^{-4} - u_8) - 10^{-4} I_{si}(u)$$

for the (1). In this case, the matrix B is so that (under mild additional conditions) the solution is unique.

For the ( ), we have

$$B(t,u) = \begin{pmatrix} c_m & -c_m & 0 \\ -c_m & c_m & 0 \\ 0 & 0 & I \end{pmatrix}, \qquad f(t,u) = \begin{pmatrix} -I_{ion} + \nabla \cdot (D_m \nabla u_1) \\ -I_{app} + I_{ion} + \nabla \cdot (D_m \nabla u_2) \\ P(u) \end{pmatrix},$$

where

$$u = (u_i, u_e, w)^T, \quad I = 1, \quad v = u_1 - u_2, , \quad P = \eta_2 \left(\frac{v}{v_p} - \eta_3 w \right)$$

for the ( ) and

$$u = (u_i, u_e, h, j, m, d, f, X, [C_a]_i)^T, \quad v = u_1 - u_2, I = \mathrm{diag}(1,1,1,1,1,1,1,1),$$
$$P = (R_1, \ldots, R_6, S)^T, \quad R_k = \alpha_k(v)(1 - u_{k+2}) - \beta_k(v) u_{k+2}, \ k = 1, \ldots, 6,$$
$$S = 0.07(10^{-4} - u_9) - 10^{-4} I_{si}(u)$$

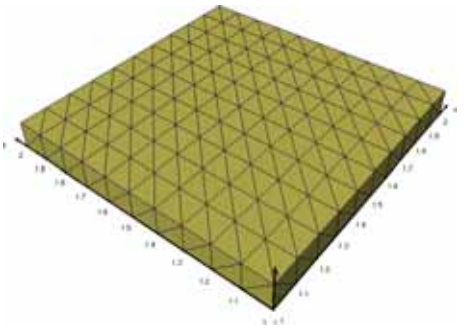


FIG. 4.1. *Initial mesh on simple test domain* Ω.

for the [illegible] ([illegible] 1).

In this case, the matrix B is [illegible], due to the rank-1 upper 2×2 diagonal block, thus exhibiting the already mentioned [illegible] character of the bidomain model. As a consequence, special attention towards a possible uniqueness of the solution must be paid. For μ constant, the vector $(\mu, \mu, 0)^T$ is an eigenfunction corresponding to the zero eigenvalue of B and of $\mathcal{M} := B/(\tau\gamma) - J(u)$ as well. Therefore, the solvability of the bidomain problem is assured for right-hand sides orthogonal to this eigenfunction, i.e., for $(\mu, \mu, 0)f(t, u) = 0$. It is easy to verify that for (3.6) the orthogonality condition holds. Thus, whenever $\int_\Omega I_{app}\, dx = 0$, the system has a unique solution up to an additive constant in the components u_i and u_e. Throughout the computation, we fixed this constant by imposing the condition $\int_\Omega u_e\, dx = 0$.

In all models, the Jacobians C_n and J_n reduce to

$$C_n = 0, \qquad J_n = \partial_u f(u)_{|u=u_n}.$$

Setting $u = (u_1, u_2, q)^T$ the nonlinear terms I_{ion} and P depend only on the difference $v = u_1 - u_2$, which means

$$I = I_{ion}(u_1 - u_2, q), \qquad P = P(u_1 - u_2, q).$$

In our simulations documented here, we present the results for a complete heartbeat in its three main phases at each point $\mathbf{x} \in \Omega$:

(a) the excitation phase (depolarization), where the potential v undergoes an abrupt temporal change within 2 msec, followed by an exponential decay; in our numerical simulations we applied a stimulus of 200 $\mu A/\text{cm}^3$ for 1 msec in a small area close to the center of the domain;
(b) the plateau phase, lasting from $40 - 50$ msec to about 400 msec according to the ionic model chosen and the type of normal and pathological heartbeat considered; v varies very little and slowly compared to the excitation phase, and the cardiac tissue is refractory; i.e., any applied stimulus does not elicit another action potential;
(c) the recovery phase (repolarization), where v returns to the rest value during a period of 20–50 msec, after which the tissue becomes excitable again.

Throughout the computations, we used an initial coarse mesh consisting of 600 tetrahedral finite elements; see Figure 4.1. As tolerances for the accuracies in time and in space we normally chose $TOL_t = TOL_x = 0.01$. This sometimes led to an overflow of our memory resources. Therefore, we restricted mesh refinement to level 3 or 4 (where level 0 was the coarse initial grid).

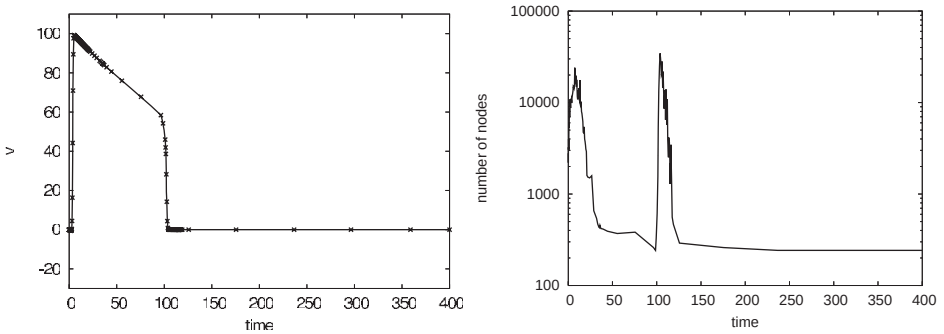


FIG. 4.2. *M-FHN model: Left: potential v at point* $(1.65, 1.65, 1.10)$ *versus time. Right: dynamically adapted number of spatial nodes.*

4.1. Monodomain model. We start with results for the M-FHN model.

In Figure 4.2(left), a typical profile of the solution component v observed at the spatial point $\mathbf{x} = (1.65, 1.65, 1.10)$ is presented. The automatically chosen time integration points nicely condense at sharp changes (exponential increase and decrease of the potential) and relax in the plateau and the recovery phases.

The time steps selected by the stepsize control in the code ROS3P range between the maximum $\tau_{\max} = 62.52$ and the minimum $\tau_{\min} = 0.0548$ msec, which means a spreading factor of about 10^3. Overall, only 130 time integration points were needed. Figure 4.2(right) documents the dynamically varying number of spatial nodes. Figure 4.3 shows the adaptive movement of the spatial grid as a whole.

For the M-LR1 model, Figure 4.4 presents plots of the action potential, the gating variables, and the calcium concentration as a function of time. Again, small time steps arise only in the depolarization and repolarization phases.

In order to illustrate the performance of our spatial adaptation procedure, we present in Figure 4.5 temporal evolutions of the action potential and of one of the gating variables for two different spatial tolerances $TOL_x = 0.02, 0.01$ using a common time tolerance $TOL_t = 0.01$. The maxima of achieved numbers of nodes at intermediate time steps were 41,156 versus 160,047 with four additional refinement steps. For $TOL_x = 0.03$, which is not depicted here, unphysical oscillations around the steep solution fronts would occur.

4.2. Bidomain model. We start with the B-FHN model.

Figure 4.6 plots the intra- and extracellular electric potentials u_i, u_e at two different points as a function of time. Observe that there is not only a moving front but a much more complex dynamics. In particular, the extracellular potentials vary a lot depending on the spatial position, both in the first wave complex (the QRS) and the last little wave (the T wave).

Next, we present the results for the B-LR1 model. Even in this most complex of our selected models the adaptive strategy of the code KARDOS works properly: the adaptivity devices both in time and in space cope nicely with the scale spreading. Figure 4.7 shows the extra- and intracellular electric potentials u_e and u_i over the whole period of 400 msec. Here the time steps selected by the stepsize control in the code ROS3P range between the maximum $\tau_{\max} = 38.60$ and the minimum $\tau_{\min} = 0.0334$ msec, which means a spreading factor of about 10^3 as in the simpler models. Overall, only 195 time integration points were needed.

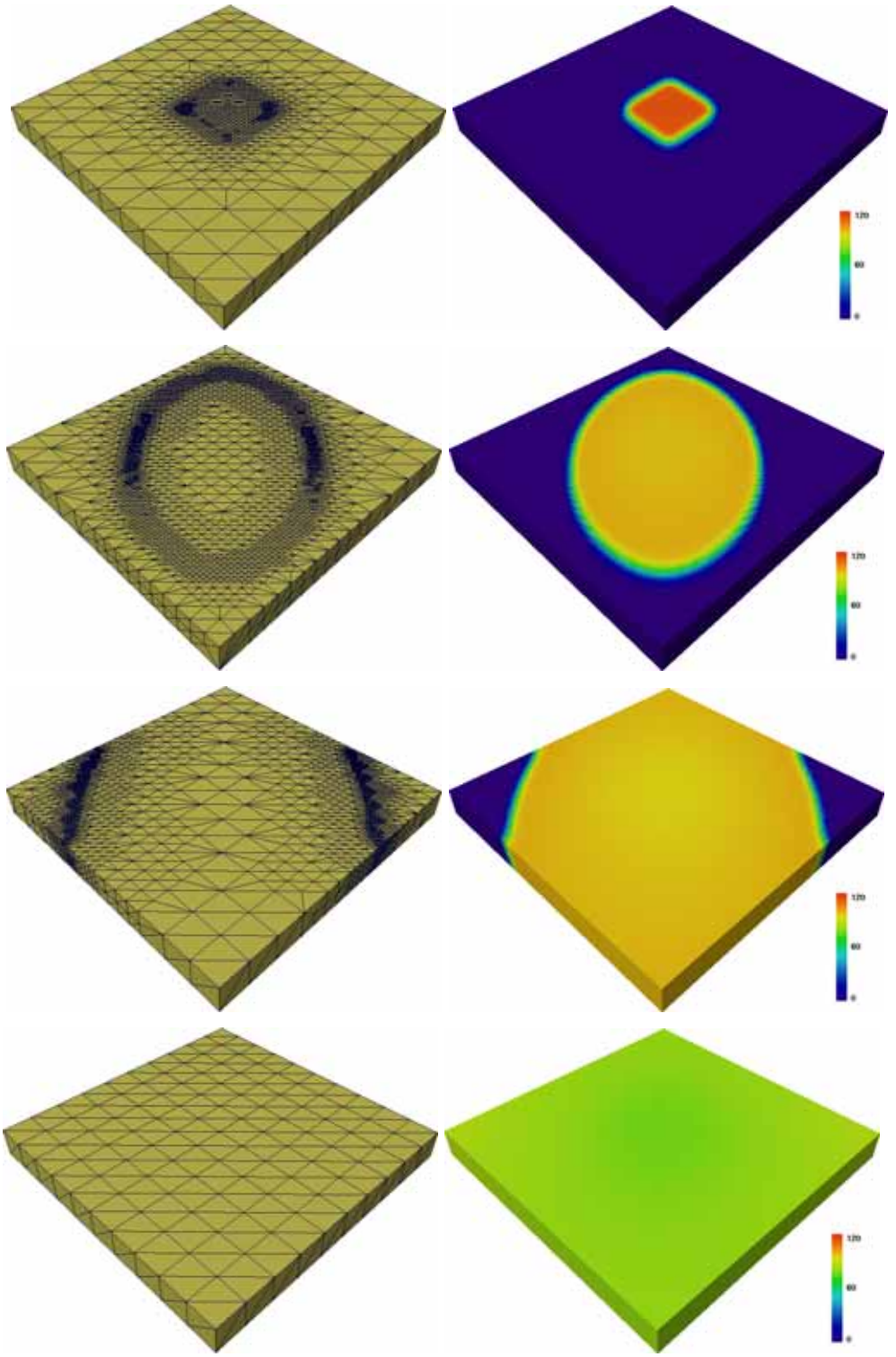


FIG. 4.3. *M-FHN model. Behavior at times* $1, 7, 13, 50$ msec. *Left: dynamical mesh adaptation. Right: potential* v.

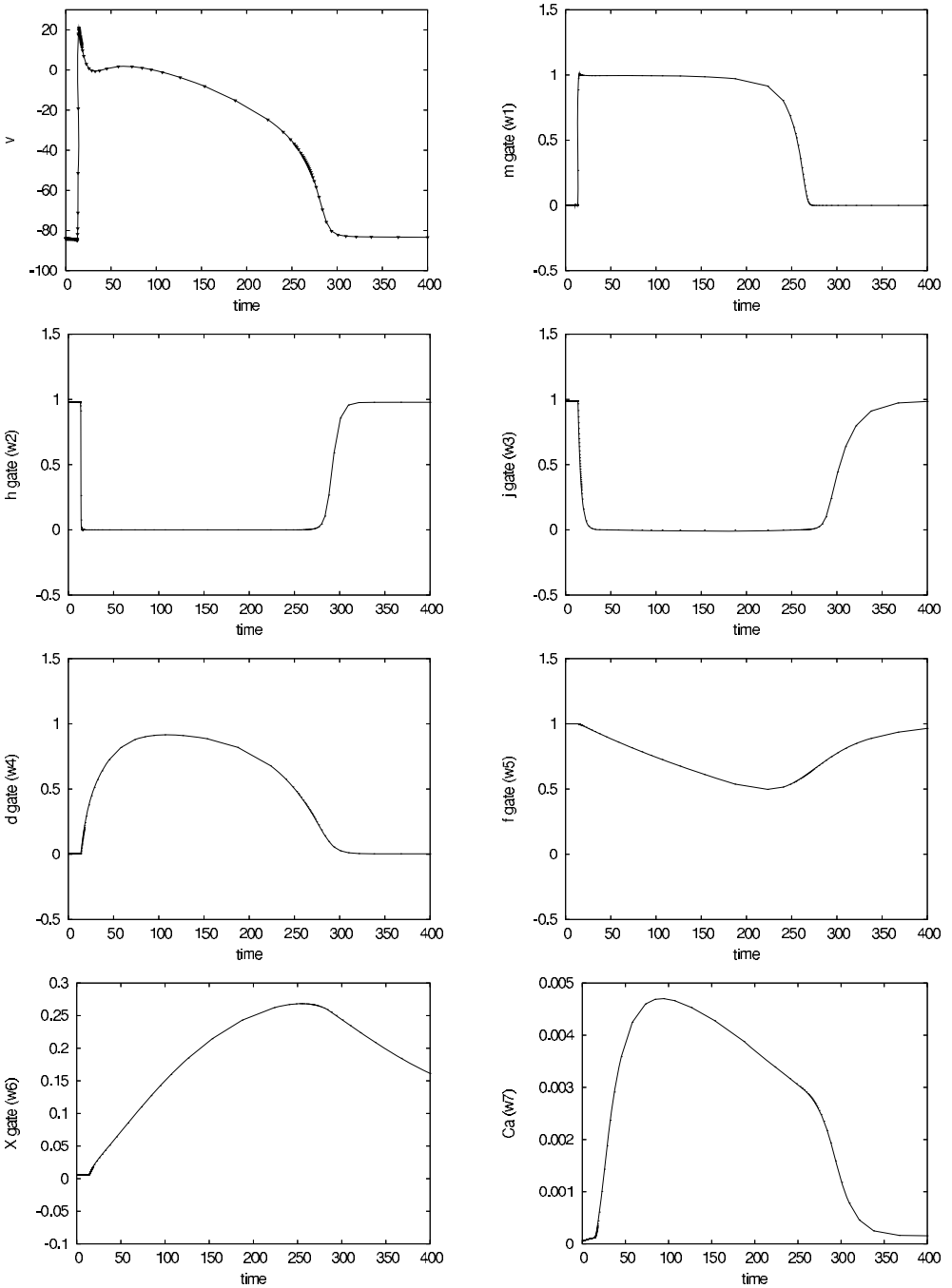


FIG. 4.4. *M-LR1 model* ($TOL_x = TOL_t = 0.01$)*: action potential* v*, gating variables* $w_1, \ldots, w_6$*, and calcium concentration* w_7 *at point* $(1.9, 1.9, 1.0)$ *versus time.*

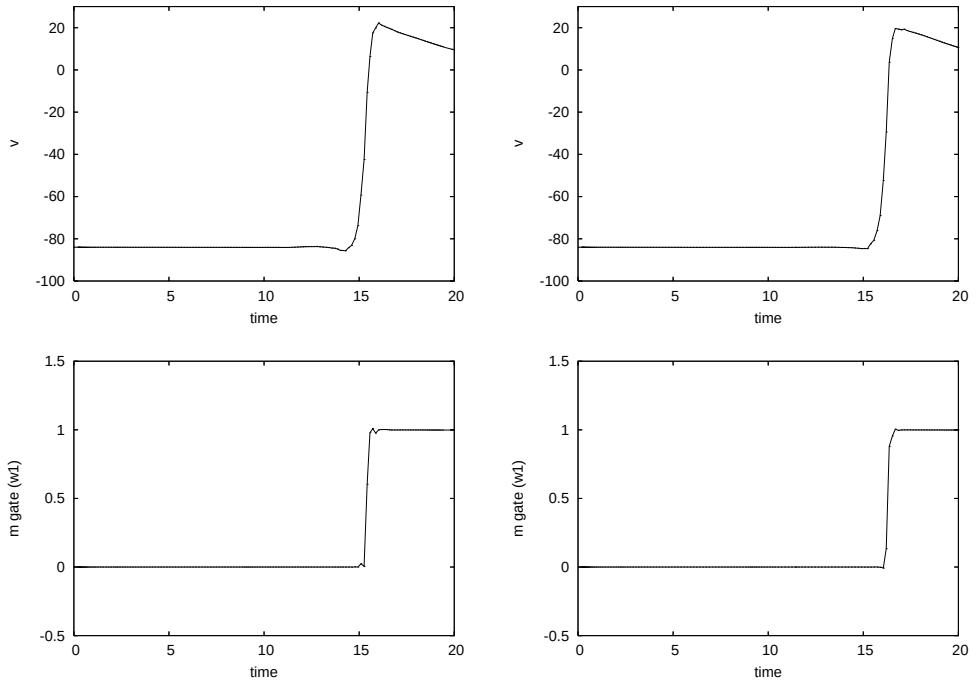


FIG. 4.5. *M-LR*1 *membrane model* ($TOL_t = 0.01$)*: action potential* v *and gating variable* w_1 *at point* $(1.9, 1.9, 1.0)$ *versus time. Left: spatial tolerance* $TOL_x = 0.02$*. Right:* $TOL_x = 0.01$*.*

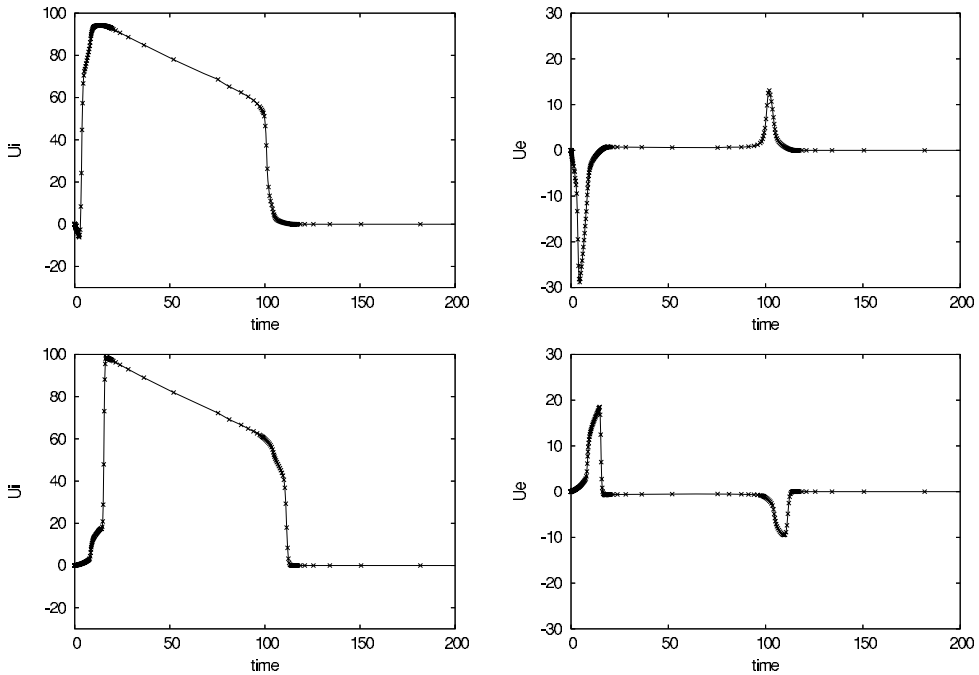


FIG. 4.6. *B-FHN model: intra- and extracellular electric potentials* u_i*,* u_e *versus time. Top: at point* (1.65,1.65,1.1)*. Bottom: at point* (1.9,1.9,1.0)*.*

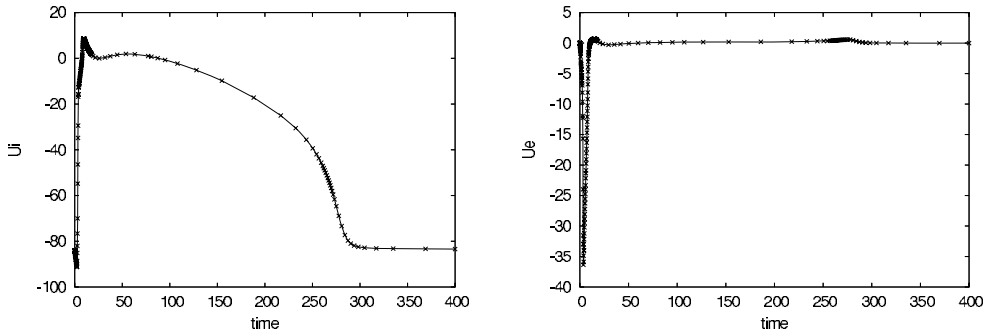


FIG. 4.7. *B-LR1 model: intra- and extracellular electric potentials* u_i, u_e *at point* $(1.65, 1.65, 1.1)$ *versus time.*

4.3. Adaptivity gain. Within our code KARDOS, we realize adaptivity both in time and in space. In , , any constant time stepping procedure would need to select a time step $\sim \tau_{\min}$ as suggested from the time-step control. In the simulations over a time span of 400 msec, this would lead to roughly $\sim 10,000$ integration steps to be compared with ~ 200 steps in the adaptive code. This gain factor is expected if the same integrator is used. However, quite often, strongly A-stable integrators are not applied (e.g., Crank–Nicolson or even explicit schemes). This might increase the number of necessary integration points even further.

As for the , , adaptation, the construction principle within KARDOS is to equilibrate the discretization errors at each time step. To be more precise, let $\tilde{v}(x,t)$, $t \in [t_n, t_{n+1}]$, be the , solution of (3.2) with initial values taken from the approximate solution $v_h(x,t)$, i.e., $\tilde{v}(x,t_n) = v_h(x,t_n)$. Using variable time-step sizes and adaptive mesh refinement as described in section 3, we obtain an error estimate Est as the sum of temporal and spatial error estimates. From theory [32], we get the error bounds

$$c\,Est \le \|(\tilde{v} - v_h)(x,t_{n+1})\| \le C\,Est,$$

where $0 < c < C$ are unknown, problem-dependent constants. In this way, local errors are kept at moderate size if $Est \le TOL$, which can be guaranteed by our adaptive method. This technique also controls the growth of global errors for dynamically unstable systems.

We illustrate the effect of spatial adaptivity by comparing uniform versus adaptive meshes, both within the time integrator ROS3P. As a simpler test example, we solve the Nagumo equation with a diffusion operator and a cubic reaction term $I(v)$, which reads

$$\partial_t v - \sum_{i=1}^{3} \sigma_i \partial_{x_i} \partial_{x_i} v(x,t) + I(v(x,t)) = f(x,t,v(x,t)),$$

$$I(v) = 1.5v\left(1 - \frac{v}{13}\right)\left(1 - \frac{v}{100}\right),$$

where $\sigma_1 = 1.2 \cdot 10^{-5}$ and $\sigma_2 = \sigma_3 = 2.556 \cdot 10^{-4}$. The function f and the boundary conditions are determined so that the travelling wave reference solution is

$$v(x,t) = w(\alpha(t - \phi(x)))$$

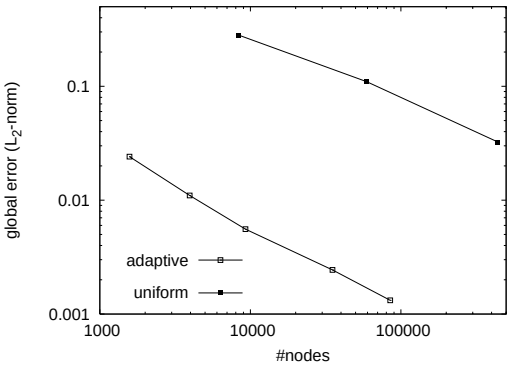


FIG. 4.8. *Impact of adaptive refinement in space at time* $t = 38$.

with

$$w(\tau) = \frac{100}{1 + e^{-6.85\tau}}, \qquad \phi = \sqrt{\Big(\frac{x_1}{0.0616}\Big)^2 + \Big(\frac{x_2}{0.0284}\Big)^2}.$$

The constant fibers are assumed to be parallel to the x_1- and the x_2-axis. The computational domain is the same as for the previous examples. Note that the solution v has the same structure of action potential during the excitation phase as the potentials in the bioelectrical models.

Figure 4.8 shows the relative global L_2-error $\|v - v_h\|/\|v\|$ at time $t = 38$ achieved for a varying number of spatial nodes. At this point in time the early excitation occurs which is crucial for the travelling wave front later on. For the uniform meshes, we chose successive mesh sizes $h = 1/40, 1/80,$ and $1/160$, whereas the adaptive meshes were obtained automatically. Obviously, for finer and finer meshes, both approaches reduce the global error, but uniform meshes need about 200 times more nodes for a comparable accuracy in the depicted L_2-norm. In the maximum norm, this factor would even be about 400.

For realistic heart geometries, the impact of spatial adaptivity is expected to play an even stronger role. In view of future computations with real life patient data, we here did not include any comparisons in terms of CPU time, since a comparison on the chosen simple geometry would not be meaningful.

Conclusion. We have presented a fully adaptive approach to solve various 3D reaction-diffusion models arising in computational electrocardiology. The model complexity ranged from the simple monodomain model coupled with a FitzHugh–Nagumo ionic membrane model (M-FHN) to the quite complex bidomain model coupled with the phase-I Luo–Rudy ionic membrane model (B-LR1). Due to the arising steep solution gradients and travelling fronts, an automatic local adaptation of spatial and temporal discretizations appeared to be crucial—not only to save computing time and storage but also to obtain accurate solutions with tolerable effort. The computations in this paper have concentrated on adaptivity and models and chosen a simple geometry. Future work will have to focus on more realistic geometries and to compare the efficiency of the advocated method with other computational approaches.

Acknowledgments. We want to thank Rainer Roitzsch (ZIB) for providing a lot of ideas to improve the code and the layout of the paper. We also are grateful to

Ulrich Nowak (ZIB) for fruitful discussions about questions of stable integration of the LR1 model.

REFERENCES

[1] *Amira homepage*, http://amira.zib.de/.
[2] *KARDOS homepage*, http://www.zib.de/Numerik/numsoft/kardos.
[3] T. Ashihara, T. Namba, M. Ito, T. Ikeda, K. Nakazawa, and N. Trayanova, *Spiral wave control by a localized stimulus: A bidomain model study*, J. Cardiovasc. Electr., 15 (2004), pp. 226–233.
[4] I. Babuška and W. C. Rheinboldt, *Error estimates for adaptive finite element computations*, SIAM J. Numer. Anal., 15 (1978), pp. 736–754.
[5] R. E. Bank and R. K. Smith, *A posteriori error estimates based on hierarchical bases*, SIAM J. Numer. Anal., 30 (1993), pp. 921–935.
[6] F. Bornemann, *An adaptive multilevel approach to parabolic equations.* III. 2*D error estimation and multilevel preconditioning*, Impact Comput. Sci. Engrg., 4 (1992), pp. 1–45.
[7] F. A. Bornemann, B. Erdmann, and R. Kornhuber, *A posteriori error estimates for elliptic problems in two and three space dimensions*, SIAM J. Numer. Anal., 33 (1996), pp. 1188–1204.
[8] E. Cherry, H. Greenside, and C. Henriquez, *A space-time adaptive method for simulating complex cardiac dynamics*, Phys. Rev. Lett., 84 (2000), pp. 1343–1346.
[9] P. Colli Franzone and L. Guerri, *Spread of excitation in* 3*-D models of the anisotropic cardiac tissue,* I. *Validation of the eikonal approach*, Math. Biosci., 113 (1993), pp. 145–209.
[10] P. Colli Franzone, L. Guerri, and S. Rovida, *Wavefront propagation in an activation model of the anisotropic cardiac tissue: Asymptotic analysis and numerical simulations*, J. Math. Biol., 28 (1990), pp. 121–176.
[11] P. Colli Franzone and L. Pavarino, *A parallel solver for reaction-diffusion systems in computational electrocardiology*, Math. Models Methods Appl. Sci., 14 (2004), pp. 883–911.
[12] P. Colli Franzone, L. Pavarino, and B. Taccardi, *Simulating patterns of excitation, repolarization and action potential duration with cardiac bidomain and monodomain models*, Math. Biosci., 197 (2005), pp. 35–66.
[13] P. Colli Franzone and G. Savaré, *Degenerate evolution systems modeling the cardiac electric field at micro and macroscopic level*, in Evolution Equations, Semigroups and Functional Analysis, A. Lorenzi and B. Ruf, eds., Birkhäuser, Basel, 2002, pp. 49–78.
[14] P. Deuflhard, *Uniqueness theorems for stiff ode initial value problems*, in Numerical Analysis, D. Griffiths and G. A. Watson, eds., Longman Scientific and Technical, Harlow, UK, 1990, pp. 74–205.
[15] P. Deuflhard, *Newton Methods for Nonlinear Problems. Affine Invariance and Adaptive Algorithms*, Springer Ser. Comput. Math. 35, Springer-Verlag, Berlin, 2004.
[16] P. Deuflhard and F. Bornemann, *Scientific Computing with Ordinary Differential Equations*, Texts Appl. Math. 42, Springer-Verlag, New York, 2002.
[17] P. Deuflhard, J. Lang, and U. Nowak, *Adaptive algorithms in dynamical process simulation*, in Progress in Industrial Mathematics at ECMI'94, H. Neunzert, ed., Wiley–Teubner, Stuttgart, 1996, pp. 122–137.
[18] P. Deuflhard, P. Leinen, and H. Yserentant, *Concepts of an adaptive hierarchical finite element code*, Impact Comput. Sci. Engrg., 1 (1989), pp. 3–35.
[19] P. Deuflhard, M. Weiser, and M. Seebass, *A new nonlinear elliptic multilevel FEM in clinical cancer therapy planning*, Comput. Visual Sci., 3 (2000), pp. 115–120.
[20] B. Erdmann, J. Lang, and R. Roitzsch, *Kardos User's Guide*, Technical Report ZR-02-42, Zuse Institute Berlin (ZIB), Berlin-Dahlem, Germany, 2002.
[21] E. Gladilin, S. Zachow, P. Deuflhard, and H.-C. Hege, *Anatomy- and physics-based facial animation for craniofacial surgery simulations*, Med. Biol. Eng. Comput., 42 (2004), pp. 167–170.
[22] K. Gustafsson, *Control-theoretic techniques for stepsize selection in implicit Runge-Kutta methods*, ACM Trans. Math. Software, 20 (1994), pp. 496–517.
[23] K. Gustafsson, M. Lundh, and G. Söderlind, *A PI stepsize control for the numerical solution of ordinary differential equations*, BIT, 28 (1988), pp. 270–287.
[24] E. Hairer and G. Wanner, *Solving Ordinary Differential Equations* II*, Stiff and Differential-Algebraic Problems*, 2nd revised ed., Springer-Verlag, Berlin, Heidelberg, New York, 1996.

[25] C. Henriquez, *Simulating the electrical behavior of cardiac tissue using the bidomain model*, Crit. Rev. Biomed. Eng., 21 (1993), pp. 1–77.
[26] C. Henriquez, A. Muzikant, and C. Smoak, *Anisotropy, fiber curvature, and bath loading effects on activation in thin and thick cardiac tissue preparations: Simulations in a three-dimensional bidomain model*, J. Cardiovasc. Electrophysiol., 7 (1996), pp. 424–444.
[27] A. Hodgkin and A. Huxley, *A quantitative description of membrane current and its application to conduction and excitation in nerve*, J. Physiol., 117 (1952), pp. 500–544.
[28] N. Hooke, C. Henriquez, P. Lanzkron, and D. Rose, *Linear algebraic transformations of the bidomain equations: Implications for numerical methods*, Math. Biosci., 120 (1994), pp. 127–145.
[29] T. Hund and Y. Rudy, *Rate transient and regulation of action potential and calcium transient in a canine cardiac ventricular cell model*, Circulation, 110 (2004), pp. 3168–3174.
[30] J. Keener and J. Sneyd, *Mathematical Physiology*, Springer-Verlag, New York, 1998.
[31] A. Kléber and Y. Rudy, *Basic mechanisms of cardiac impulse propagation and associated arrhythmias*, Physiol. Rev., 184 (2004), pp. 431–488.
[32] J. Lang, *Adaptive Multilevel Solution of Nonlinear Parabolic PDE Systems. Theory, Algorithm, and Applications.*, Lect. Notes Comput. Sci. Eng. 16, Springer-Verlag, Berlin, 2000.
[33] J. Lang, B. Erdmann, and M. Seebass, *The impact of a nonlinear heat transfer model for temperature control in regional hyperthermia*, IEEE Trans. Biomed. Engrg., 46 (1999), pp. 1129–1138.
[34] J. Lang and J. Verwer, *Ros3p: An accurate third-order Rosenbrock solver designed for parabolic problems*, BIT, 41 (2001), pp. 730–737.
[35] J. Lang and A. Walter, *A finite element method adaptive in space and time for nonlinear reaction–diffusion systems*, Impact Comput. Sci. Engrg., 4 (1992), pp. 269–314.
[36] I. LeGrice, B. Smaill, and P. Hunter, *Laminar structure of the heart: A mathematical model*, Amer. J. Physiol. (Heart Circ. Physiol.), 272 (1997), pp. 2466–2476.
[37] G. T. Lines, P. Grøttum, and A. Tveito, *Modeling the electrical activity of the heart: A bidomain model of the ventricles embedded in a torso*, Comput. Visual. Sci., 5 (2003), pp. 195–213.
[38] C. Lubich and M. Roche, *Rosenbrock methods for differential-algebraic systems with solution-dependent singular matrix multiplying the derivative*, Computing, 43 (1990), pp. 325–342.
[39] C. Luo and Y. Rudy, *A model of the ventricular cardiac action potential: Depolarization, repolarization, and their interaction*, Circ. Res., 68 (1991), pp. 1501–1526.
[40] P. K. Moore, *An adaptive finite element method for parabolic differential systems: Some algorithmic considerations in solving in three space dimensions*, SIAM J. Sci. Comput., 21 (2000), pp. 1567–1586.
[41] M. Murillo and X.-C. Cai, *A fully implicit parallel algorithm for simulating the non-linear electrical activity of the heart*, Numer. Linear Algebra Appl., 11 (2004), pp. 261–277.
[42] A. Muzikant, E. W. Hsu, P. D. Wolf, and C. S. Henriquez, *Region specific modeling of cardiac muscle: Comparison of simulated and experimental potentials*, Ann. Biomed. Eng., 30 (2003), pp. 867–883.
[43] A. Panfilov and A. Holden, eds., *Computational Biology of the Heart*, Wiley, Chichester 1997.
[44] L. Pavarino and P. Colli Franzone, *Parallel solution of cardiac reaction-diffusion models*, in Domain Decomposition Methods in Science and Engineering, Lect. Notes Comput. Sci. Eng. 40, R. Kornhuber, R. Hoppe, J. Périaux, O. Pironneau, O. Widlund, and J. Xu, eds., Springer-Verlag, Berlin, 2004, pp. 669–676.
[45] M. Pennacchio and V. Simoncini, *Efficient algebraic solution of reaction-diffusion systems for the cardiac excitation process*, J. Comput. Appl. Math., 145 (2002), pp. 49–70.
[46] J. Pormann, *A Simulation System for the Bidomain Equations*, Ph.D. thesis, Duke University, Durham, NC, 1999.
[47] J. Rogers and A. McCulloch, *A collocation-Galerkin finite element model of cardiac action potential propagation*, IEEE Trans. Biomed. Engrg., 41 (1994), pp. 743–757.
[48] B. Roth, *How the anisotropy of the intracellular and extracellular conductivities influence stimulation of cardiac muscle*, J. Math. Biol., 30 (1992), pp. 633–646.
[49] A. Sambelashvili and I. Efimov, *Dynamics of virtual electrode-induced scroll-wave reentry in a 3D bidomain model*, Amer. J. Physiol. (Heart Circ. Physiol.), 287 (2004), pp. 1570–1581.
[50] D. Stalling, M. Westerhoff, and H.-C. Hege, *Amira: A highly interactive system for visual data analysis*, in The Visualization Handbook, C. Hansen and C. Johnson, eds., Elsevier, Amsterdam, 2005, pp. 749–767.
[51] D. Streeter, *Gross morphology and fiber geometry in the heart*, in Handbook of Physiology, Vol. 1, R. Berne, ed., Williams & Wilkins, Baltimore, MD, 1979, pp. 61–112.

[52] J. Sundnes, G. T. Lines, P. Grøttum, and A. Tveito, *Electrical activity in the human heart*, in Advanced Topics in Computational Partial Differential Equations, Lect. Notes Comput. Sci. Eng. 33, H. Langtangen and A. Tveito, eds., Springer-Verlag, 2004, pp. 401–449.

[53] J. Trangenstein and C. Kim, *Operator splitting and adaptive mesh refinement for the Luo-Rudy* I *model*, J. Comput. Phys., 196 (2004), pp. 645–679.

[54] H. A. van der Vorst, *BI-CGSTAB: A fast and smoothly converging variant of BI-CG for the solution of nonsymmetric linear systems*, SIAM J. Sci. Statist., 13 (1992), pp. 631–644.

[55] E. Vigmond, F. Aguel, and N. Trayanova, *Computational techniques for solving the bidomain equations in three dimensions*, IEEE Trans. Biomed. Engrg., 49 (2002), pp. 1260–1269.

[56] R. W. Dos Santos, G. Plank, S. Bauer, and E. Vigmond, *Preconditioning techniques for the Bidomain equations*, in Domain Decomposition Methods in Science and Engineering, Lect. Notes Comput. Sci. Eng. 40, R. Kornhuber, R. Hoppe, J. Périaux, O. Pironneau, O. Widlund, and J. Xu, eds., Springer-Verlag, Berlin, 2004, pp. 571–580.

[57] H. Yu, *A local space-time adaptive scheme in solving two-dimensional parabolic problems based on domain decomposition methods*, SIAM J. Sci. Comput., 23 (2001), pp. 304–322.

[58] S. Zachow, M. Weiser, H.-C. Hege, and P. Deuflhard, *Soft tissue prediction in computer assisted maxillofacial surgery planning: A quantitative evaluation of histomechanical modeling using pre- and postoperative CT-Data*, in Biomechanics Applied to Computer Assisted Surgery, Research Signpost, Trivandrum, Kerala, India 2005, pp. 277–298.

[59] D. Zipes and J. Jalife, *Cardiac Electrophysiology*, 4th ed., W.B. Saunders, Philadelphia, 2004.

SIAM J. SCI. COMPUT.
Vol. 28, No. 3, pp. 963–983
© 2006 Society for Industrial and Applied Mathematics

ON LARGE-SCALE DIAGONALIZATION TECHNIQUES FOR THE ANDERSON MODEL OF LOCALIZATION*

OLAF SCHENK†, MATTHIAS BOLLHÖFER‡, AND RUDOLF A. RÖMER§

Abstract. We propose efficient preconditioning algorithms for an eigenvalue problem arising in quantum physics, namely the computation of a few interior eigenvalues and their associated eigenvectors for large-scale sparse real and symmetric indefinite matrices of the Anderson model of localization. We compare the Lanczos algorithm in the 1987 implementation by Cullum and Willoughby with the shift-and-invert techniques in the implicitly restarted Lanczos method and in the JACOBI–DAVIDSON method. Our preconditioning approaches for the shift-and-invert symmetric indefinite linear system are based on maximum weighted matchings and algebraic multilevel incomplete LDL^T factorizations. These techniques can be seen as a complement to the alternative idea of using more complete pivoting techniques for the highly ill-conditioned symmetric indefinite Anderson matrices. We demonstrate the effectiveness and the numerical accuracy of these algorithms. Our numerical examples reveal that recent algebraic multilevel preconditioning solvers can accelerate the computation of a large-scale eigenvalue problem corresponding to the Anderson model of localization by several orders of magnitude.

Key words. Anderson model of localization, large-scale eigenvalue problem, Lanczos algorithm, JACOBI–DAVIDSON algorithm, Cullum–Willoughby implementation, symmetric indefinite matrix, multilevel preconditioning, maximum weighted matching

AMS subject classifications. 65F15, 65F50, 82B44, 65F10, 65F05, 05C85

DOI. 10.1137/050637649

1. Introduction.

One of the hardest challenges in modern eigenvalue computation is the numerical solution of large-scale eigenvalue problems, in particular those arising from quantum physics such as, e.g., the Anderson model of localization (see section 3 for details). Typically, these problems require the computation of some eigenvalues and eigenvectors for systems which have up to several million unknowns due to their high spatial dimensions. Furthermore, their underlying structure involves random perturbations of matrix elements which invalidates simple preconditioning approaches based on the graph of the matrices. Moreover, one is often interested in finding some eigenvalues and associated eigenvectors in the interior of the spectrum. The classical Lanczos approach [51] has led to eigenvalue algorithms [16, 17] that are, in principle, able to compute these eigenvalues using only a small amount of memory. More recent work on implicitly restarted Lanczos techniques [42] has accelerated these methods significantly, yet to be fast one needs to combine this approach with shift-and-invert techniques; i.e., in every step one has to solve a shifted system of type $A - \sigma I$, where σ is a shift near the desired eigenvalues and $A \in \mathbb{R}^{n,n}$,

*Received by the editors August 5, 2005; accepted for publication (in revised form) January 31, 2006; published electronically June 19, 2006. This work was supported by the Swiss Commission for Technology and Innovation (CTI) under grant 7036 ENS-ES and by the EPSRC under grant EP/C007042/1.

http://www.siam.org/journals/sisc/28-3/63764.html

†Department of Computer Science, University of Basel, Klingelbergstrasse 50, CH-4056 Basel, Switzerland (olaf.schenk@unibas.ch).

‡Department of Mathematics, MA 4-5, TU Berlin, Str. des 17. Juni, 10623 Berlin, Germany (bolle@math.tu-berlin.de). The work of this author was supported by the DFG research center MATHEON "Mathematics for Key Technologies" in Berlin.

§Centre for Scientific Computing and Department of Physics, University of Warwick, Coventry CV4 7AL, UK (r.roemer@warwick.ac.uk).

$A = A^T$ is the associated matrix. In general, shift-and-invert techniques converge rather quickly, which is in line with the theory [51]. Still, a linear solver is required to solve systems $(A - \sigma I)x = b$ efficiently with respect to time and memory. While implicitly restarted Lanczos techniques [42] usually require the solution of the system $(A - \sigma I)x = b$ to maximum precision, and thus are mainly suited for sparse direct solvers, the JACOBI–DAVIDSON method has become an attractive alternative [61], in particular when dealing with preconditioning methods for linear systems.

Until recently, sparse symmetric indefinite direct solvers were not as efficient as symmetric positive definite solvers, and this might have been one major reason why shift-and-invert techniques were not able to compete with traditional Lanczos techniques [27], in particular because of memory constraints. With the invention of fast matchings-based algorithms [49], which improve the diagonal dominance of linear systems, the situation has dramatically changed and the impact on preconditioning methods [7], as well as the benefits for sparse direct solvers [58], has been recognized. Furthermore, these techniques have been successfully transferred to the symmetric case [22, 24], allowing modern state-of-the-art direct solvers [57] to be orders of magnitudes faster and more memory efficient than ever, finally leading to symmetric indefinite sparse direct solvers that are almost as efficient as their symmetric positive definite counterparts. Recently this approach has also been utilized to construct incomplete factorizations [38] with similar dramatic success. For a detailed survey on preconditioning techniques for large symmetric indefinite linear systems, the interested reader should consult [5, 6].

2. Numerical approach for large systems. In the present paper we combine the above mentioned advances with inverse-based preconditioning techniques [8]. This allows us to find interior eigenvalues and eigenvectors for the Anderson problem several orders of magnitudes faster than traditional algorithms [16, 17] while still keeping the amount of memory reasonably small.

Let us briefly outline our strategy. We will consider recent novel approaches in preconditioning methods for symmetric indefinite linear systems and eigenvalue problems and apply them to the Anderson model. Since the Anderson model is a large-scale sparse eigenvalue problem in three spatial dimensions, the eigenvalue solvers we deal with are designed to compute only a few interior eigenvalues and eigenvectors, thus avoiding a complete factorization. In particular we will use two modern eigenvalue solvers, which we will briefly introduce in section 5. The first one is ARPACK [42], which is a Lanczos-type method using implicit restarts (cf. section 5.1). We use this algorithm together with a shift-and-invert technique; i.e., eigenvalues and eigenvectors of $(A - \sigma I)^{-1}$ are computed instead of those of A. ARPACK is used in conjunction with a direct factorization method and a multilevel incomplete factorization method for the shift-and-invert technique.

First, we use the shift-and-invert technique with the novel symmetric indefinite sparse direct solver that is part of PARDISO [57], and we report extensive numerical results on the performance of this method. Section 6 will give a short overview of the main concepts that form the PARDISO solver. Second, we use ARPACK in combination with the multilevel incomplete LU factorization package ILUPACK [9]. Here we present a new version of this preconditioner that is devoted to symmetric indefinite problems and combines two basic ideas, namely (i) symmetric maximum weighted matchings [22, 24] and (ii) inverse-based decomposition techniques [8]. These will be described in sections 6.2 and 8.

As a second eigenvalue solver we use the symmetric version of the JACOBI–

DAVIDSON method, in particular the implementation JDBSYM [32]. This Newton-type method (see section 5.2) is used together with ILUPACK [9]. As we will see in several further numerical experiments, the synergy of both approaches will form an extremely efficient preconditioner for the Anderson model that is memory efficient while at the same time accelerates the eigenvalue computations significantly; i.e., system sizes that resulted in weeks of computing time [27] can now be computed within an hour.

3. The Anderson model of localization. The Anderson model of localization is a paradigmatic model describing the electronic transport properties of disordered quantum systems [41, 54]. It has been used successfully in amorphous materials such as alloys [52], semiconductors, and even DNA [53]. Its hallmark is the prediction of a spatial confinement of the electronic motion upon increasing the disorder—the so-called Anderson [illegible] [2]. When the model is used in three spatial dimensions, it exhibits a metal-insulator transition in which the disorder strength w mediates a change of transport properties from metallic behavior at small w via critical behavior at the transition w_c to insulating behavior and strong localization at larger w [41, 54]. Mathematically, the quantum problem corresponds to a Hamilton operator in the form of a real symmetric matrix A, with quantum mechanical energy levels given by the eigenvalues $\{\lambda\}$, and the respective wave functions are simply the eigenvectors of A, i.e., vectors x with real entries. With $N = M \times M \times M$ sites, the quantum mechanical (stationary) Schrödinger equation is equivalent to the eigenvalue equation $Ax = \lambda x$, which in site representation reads as

(3.1)
$$x_{i-1;j;k} + x_{i+1;j;k} + x_{i;j-1;k} + x_{i;j+1;k} + x_{i;j;k-1} + x_{i;j;k+1} + \varepsilon_{i;j;k} x_{i;j;k} = \lambda x_{i;j;k},$$

with $i, j, k = 1, 2, \ldots, M$, denoting the Cartesian coordinates of a site. The disorder enters the matrix on the diagonal, where the entries $\varepsilon_{i;j;k}$ correspond to a spatially varying disorder potential and are selected randomly according to a suitable distribution [40]. Here, we shall use the standard box distribution $\varepsilon_{i;j;k} \in [-w/2, w/2]$ such that the w parameterizes the aforementioned disorder strength. Clearly, the eigenvalues of A then lie within the interval $[-6 - w/2, 6 + w/2]$ due to the Gershgorin circle theorem. In most studies of the disorder-induced metal-insulator transition, w ranges from 1 to 30 [54]. But these values also depend on whether generalizations to random off-diagonal elements [26, 63] (the so-called random-hopping problem), anisotropies [44, 47], or other lattice graphs [36, 60] are being considered.

The intrinsic physics of the model is quite rich. For disorders $w \ll 16.5$, the eigenvectors are extended, i.e., $x_{i;j;k}$ is fluctuating from site to site, but the envelope $|x|$ is approximately a nonzero constant. For large disorders $w > 16.5$, all eigenvectors are localized such that the envelope $|x_n|$ of the nth eigenstate may be approximately written as $\exp -|\vec{r} - \vec{r}_n|/l_n(w)$ with $\vec{r} = (i, j, k)^T$ and $l_n(w)$ denoting the [illegible] of the eigenstate. In Figure 1, we show examples of such states. Note that $|x|^2$ and not x corresponds to a physically measurable quantity and is therefore the observable quantity of interest to physicists. Directly at $w = w_c \approx 16.5$, the extended states at $\lambda = 0$ vanish and no current can flow. The wave function vector x appears simultaneously extended and localized, as shown in Figure 2.

In order to numerically distinguish these three regimes, namely, localized, critical, and extended behaviors, one needs to (i) go to extremely large system sizes of order 10^6 to 10^8 and (ii) average over many different realizations of the disorder, i.e., compute eigenvalues or eigenvectors for many matrices with different diagonals. In the present paper, we concentrate on the computation of a few eigenvalues and corresponding

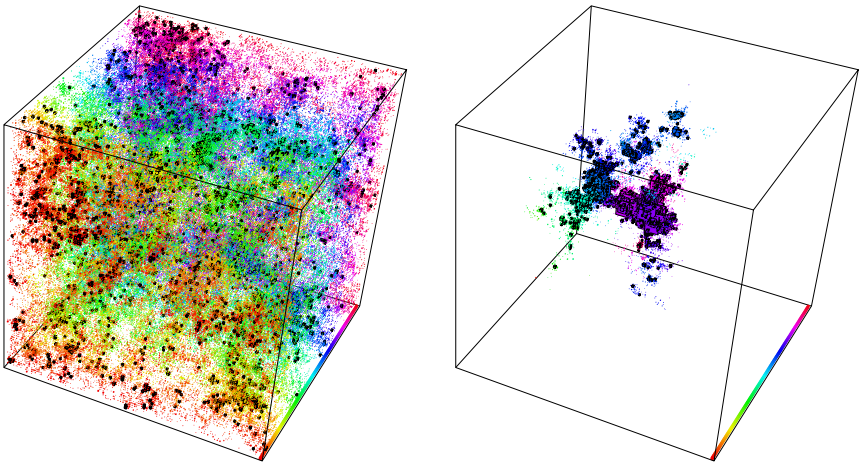


FIG. 1. *Extended (left) and localized (right) wave function probabilities for the 3D Anderson model with periodic boundary conditions at $\lambda = 0$ with $N = 100^3$ and $w = 12.0$ and 21.0, respectively. Every site with probability $|x_j|^2$ larger than the average $1/N^3$ is shown as a box with volume $|x_j|^2 N$. Boxes with $|x_j|^2 N > \sqrt{1000}$ are plotted with black edges. The color scale distinguishes between different slices of the system along the axis into the page. The eigenstates have been constructed using* ARPACK *in shift-and-invert mode with* PARDISO *as a direct solver. See section 9 for details.*

eigenvectors for the physically most interesting case of critical disorder w_c and in the center of $\sigma(A)$, i.e., at $\lambda = 0$, for large system sizes [3, 10, 46, 64]. Since there is a high density of states for $\sigma(A)$ at $\lambda = 0$ in all cases, we have the further numerical challenge of clearly distinguishing the eigenstates in this high density region.

4. The Lanczos algorithm and the Cullum–Willoughby implementation. Since the mid 1980s, the preferred numerical tool for studying the Anderson matrix and computing a selected set of eigenvectors, e.g., as needed for a multifractal analysis at the transition, was the Cullum–Willoughby implementation (CWI) [16, 17, 18] of the Lanczos algorithm. Since both CWI and the algorithm itself are well known, let us here just briefly recall the algorithm's main benefits, mostly to define our notation. The algorithm iteratively generates a sequence of orthogonal vectors v_i, $i = 1, \ldots, K$, such that $V_K^T A V_K = T_K$, with $V = [v_1, v_2, \ldots, v_K]$ and T_K a symmetric tridiagonal $K \times K$ matrix. The recursion $\beta_{i+1} v_{i+1} = A v_i - \alpha_i v_i - \beta_i v_{i-1}$ defines the diagonal and subdiagonal entries of T_K, $\alpha_i = v_i^T A v_i$, and $\beta_{i+1} = v_{i+1} A v_i$, respectively. Its associated (Ritz) eigenvalues and eigenvectors then yield those for the A's.

The CWI avoids reorthogonalization of the v_i's and hence is very memory efficient. The sparsity of the matrix A can be used to full advantage. However, one needs to construct many Ritz vectors of T_K, which is computationally intensive. Nevertheless, in 1999 CWI was still significantly faster than more modern iterative schemes [27]. The main reason for this surprising result lies in the indefiniteness of the sparse matrix A, which led to severe difficulties with solvers more accustomed to standard Laplacian-type problems.

5. Modern approaches for solving symmetric indefinite eigenvalue problems. When dealing with eigenvalues near a given real shift σ, the Lanczos algorithm [51] is usually accelerated when being applied to the shifted inverse $(A - \sigma I)^{-1}$

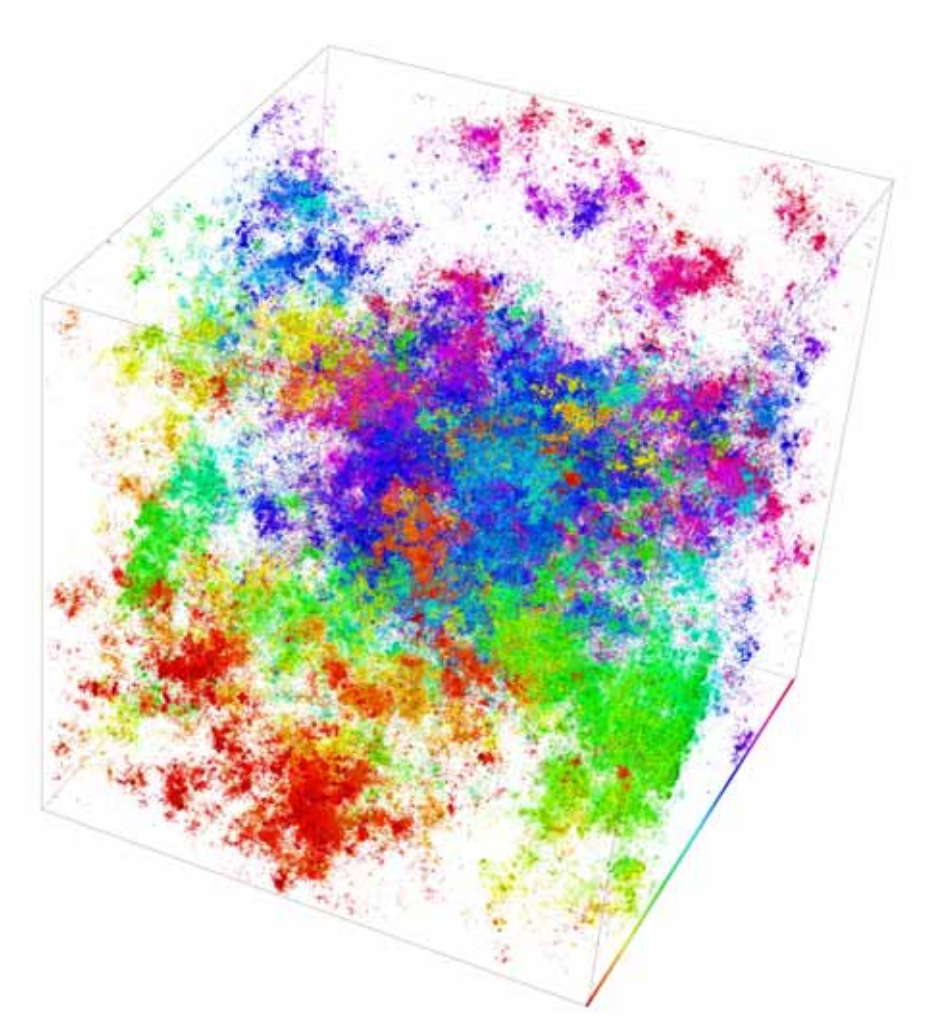


FIG. 2. *Plot of the electronic eigenstate at the metal-insulator transition with* $\lambda = 0$, $w = 16.5$, *and* $N = 350^3$. *The box-and-color scheme is as in Figure* 1. *Note how the state* extends *nearly everywhere while at the same time exhibiting certain* localized *regions of higher* $|x_j|^2$ *values. The eigenstate has been constructed using* ILUPACK*-based* JACOBI–DAVIDSON. *See section* 9 *for details.*

instead of A directly. This approach relies on the availability of a fast solution method for linear systems of type $(A - \sigma I)x = b$. However, the limited amount of available memory allows only for a small number of solution steps, and sparse direct solvers also need to be memory efficient to turn this approach into a practical method.

The limited number of Lanczos steps has led to modern implicitly restarted methods [42, 62] which ensure that the information about the desired eigenvalues is inherited when being restarted. With an increasing number of preconditioned iterative methods for linear systems [55], Lanczos-type algorithms have become less attractive mainly because in every iteration step the systems of type $(A - \sigma I)x = b$ have to be solved to „ ,, accuracy in order to avoid false eigenvalues. In contrast to this, JACOBI–DAVIDSON-like methods [61] allow using a crude approximation of the under-

lying linear system. From the point of view of linear solvers as part of the eigenvalue computation, modern direct and iterative methods need to inherit the symmetric structure $A = A^T$ while maintaining both time and memory efficiency. Symmetric matching algorithms [22, 24, 57] have significantly improved these methods.

5.1. The shift-and-invert mode of the restarted Lanczos method. The Lanczos method for real symmetric matrices A near a shift σ is based on computing successively orthonormal vectors $[v_1, \ldots, v_k, v_{k+1}]$ and a tridiagonal $(k+1) \times k$ matrix

$$\underline{T_k} = \left(\begin{array}{cccc} \alpha_1 & \beta_1 & & \\ \beta_1 & \alpha_2 & \ddots & \\ & \ddots & \ddots & \beta_{k-1} \\ & & \beta_{k-1} & \alpha_k \\ \hline & & & \beta_k \end{array} \right) \equiv \left(\begin{array}{c} T_k \\ \hline \beta_k e_k^T \end{array} \right), \tag{5.1}$$

where e_k is the kth unit vector in $\mathbb{R}^k$, such that

$$(A - \sigma I)^{-1}[v_1, \ldots, v_k] = [v_1, \ldots, v_k, v_{k+1}]\underline{T_k}. \tag{5.2}$$

Since only a limited number of Lanczos vectors $v_1, \ldots, v_k$ can be stored, and since this Lanczos sequence also consists of redundant information about undesired small eigenvalues, implicitly restarted Lanczos methods have been proposed [62, 42] that use implicitly shifted QR [35], exploiting the small eigenvalues of T_k to remove them from this sequence without ever forming a single matrix vector multiplication with $(A - \sigma I)^{-1}$. The new transformed Lanczos sequence

$$(A - \sigma I)^{-1}[\tilde{v}_1, \ldots, \tilde{v}_l] = [\tilde{v}_1, \ldots, \tilde{v}_l, \tilde{v}_{l+1}]\underline{\tilde{T}_l} \tag{5.3}$$

with $l \ll k$ then allows one to compute further $k - l$ approximations. This approach is at the heart of the symmetric version of ARPACK [42].

5.2. The symmetric JACOBI–DAVIDSON method. One of the major drawbacks of shift-and-invert Lanczos algorithms is the fact that the multiplication with $(A - \sigma I)^{-1}$ requires solving a linear system to full accuracy. In contrast to this, JACOBI–DAVIDSON-like algorithms [61] are based on a Newton-like approach to solve the eigenvalue problem. Like the Lanczos method, the search space is expanded step by step, solving the correction equation

$$(I - uu^T)(A - \theta I)(I - uu^T)z = -r \quad \text{such that} \quad z = (I - uu^T)z, \tag{5.4}$$

where (u, θ) is the given approximate eigenpair and $r = Au - \theta u$ is the associated residual. Then the search space based on $V_k = [v_1, \ldots, v_k]$ is expanded by reorthogonalizing z with respect to $[v_1, \ldots, v_k]$, and a new approximate eigenpair is computed from the Ritz approximation $[V_k, z]^T A[V_k, z]$. When computing several right eigenvectors, the projection $I - uu^T$ has to be replaced with $I - [Q, u][Q, u]^T$ using the already computed approximate eigenvectors Q. This ensures that the new approximate eigenpair is orthogonal to those that have already been computed.

The most important part of the JACOBI–DAVIDSON approach is to construct an approximate solution for (5.4) such that

$$(I - uu^T)K(I - uu^T)c = d \quad \text{with} \quad u^T z = 0 \tag{5.5}$$

and $K \approx A - \theta I$ that allows for a fast solution of the system $Kx = b$. Here, there is a strong need for robust preconditioning methods that preserve symmetry and efficiently solve sequences of linear systems with K. If K is itself symmetric and indefinite, then the simplified QMR method [29, 30] using the preconditioner $\left(I - \frac{uw^T}{w^T u}\right) K^{-1}$, where $Kw = u$ and the system matrix $\left(I - uu^T\right)(A - \theta I)$, can be used as an iterative method. Note that here the accuracy of the solution of (5.4) is uncritical until the approximate eigenpair converges [28]. This fact has been exploited in JDBSYM [4, 32]. For an overview on JACOBI–DAVIDSON methods for symmetric matrices see [33].

6. On recent algorithms for solving symmetric indefinite systems of equations. We now report on recent improvements in solving symmetric indefinite systems of linear equations that have significantly changed sparse direct as well as preconditioning methods. One key to the success of these approaches is the use of symmetric matchings, which we review in section 6.2.

6.1. Sparse direct factorization methods. For a long time, dynamic pivoting has been a central tool by which nonsymmetric sparse linear solvers gain stability. Therefore, improvements in speeding up direct factorization methods were limited to the uncertainties that have arisen from using pivoting. Certain techniques, like the column elimination tree [19, 34], have been useful for predicting the sparsity pattern despite pivoting. However, in the symmetric case the situation becomes more complicated since only symmetric reorderings, applied to both columns and rows, are required, and no a priori choice of pivots is given. This makes it almost impossible to predict the elimination tree in a sensible manner, and the use of cache-oriented level-3 BLAS [20, 21] is impossible.

With the introduction of symmetric maximum weighted matchings [22] as an alternative to complete pivoting, it is now possible to treat symmetric indefinite systems similarly to how we treat symmetric positive definite systems. This allows us to predict fill using the elimination tree [31], and thus allows us to set up the data structures that are required to predict dense submatrices (also known as supernodes). This in turn means that one is able to exploit level-3 BLAS applied to the supernodes. Consequently, the classical Bunch–Kaufman pivoting approach [12] needs to be performed only inside the supernodes.

This approach has recently been successfully implemented in the sparse direct solver PARDISO [57]. As a major consequence of this novel approach, the sparse indefinite solver has been improved to become almost as efficient as its symmetric positive analogy. Certainly for the Anderson problem studied here, PARDISO is about two orders of magnitude more efficient than previously used direct solvers [27]. We also note that the idea of symmetric weighted matchings can be carried over to incomplete factorization methods with similar success [38].

6.2. Symmetric weighted matchings as an alternative to complete pivoting techniques. Symmetric weighted matchings [22, 24], which will be explained in detail in section 7.2, can be viewed as a preprocessing step that rescales the original matrix and at the same time improves the block diagonal dominance. By this strategy, all entries are at most one in modulus, and, in addition, the diagonal blocks are either 1×1 scalars a_{ii} such that $|a_{ii}| = 1$ (in exceptional cases we will have $a_{ii} = 0$) or 2×2 blocks

$$\begin{pmatrix} a_{ii} & a_{i,i+1} \\ a_{i+1,i} & a_{i+1,i+1} \end{pmatrix} \text{ such that } |a_{ii}|, |a_{i+1,i+1}| \leqslant 1, \text{ and } |a_{i+1,i}| = |a_{i,i+1}| = 1.$$

Although this strategy does not necessarily ensure that symmetric pivoting, as in [12], is unnecessary, it is nevertheless likely to waive dynamic pivoting during the factorization process. It has been shown in [24] that, based on symmetric weighted matchings, the performance of the sparse symmetric indefinite [illegible] direct solver MA57 is improved significantly, although a dynamic pivoting strategy by Duff and Reid [25] was still present. Recent results in [57] have shown that the absence of dynamic pivoting does not harm the method anymore and that, therefore, symmetric weighted matchings can be considered as an alternative to complete pivoting.

7. Symmetric reorderings to improve the results of pivoting on restricted subsets. In this section we will discuss weighted graph matchings as an additional preprocessing step. The motivation for weighted matching approaches is to identify large entries in the coefficient matrix A that, if permuted close to the diagonal, permit the factorization process to identify more acceptable pivots and proceed with fewer pivot perturbations. These methods are based on maximum weighted matchings $\mathcal{M}$ and improve the quality of the factor in a way complementary to the alternative idea of using more complete pivoting techniques. The idea of using a permutation $P_{\mathcal{M}}$ associated with a weighted matching $\mathcal{M}$ as an approximation of the pivoting order for nonsymmetric linear systems was first introduced by Olschowka and Neumaier [49] and extended by Duff and Koster [23] to the sparse case. Permuting the rows $A \leftarrow P_{\mathcal{M}}A$ of the sparse system to ensure a zero-free diagonal or to maximize the product of the absolute values of the diagonal entries are techniques that are now often regularly used for nonsymmetric matrices [7, 45, 58, 59].

7.1. Matching algorithms for nonsymmetric matrices. Let $A = (a_{ij}) \in \mathbb{R}^{n\times n}$ be a general matrix. The nonzero elements of A define a graph with edges $\mathcal{E} = \{(i,j) : a_{ij} \neq 0\}$ of ordered pairs of row and column indices. A subset $\mathcal{M} \subset \mathcal{E}$ is called a matching, or a transversal, if every row index i and every column index j appears at most once in $\mathcal{M}$. A matching $\mathcal{M}$ is called perfect if its cardinality is n. For a nonsingular matrix, at least one perfect matching exists and can be found with well known algorithms. With a perfect matching $\mathcal{M}$, it is possible to define a permutation matrix $P_{\mathcal{M}} = (p_{ij})$ with

$$p_{ij} = \begin{cases} 1 & (j,i) \in \mathcal{M}, \\ 0 & \text{otherwise.} \end{cases} \tag{7.1}$$

As a consequence, the permutation matrix $P_{\mathcal{M}}A$ has nonzero elements on its diagonal. This method takes only the nonzero structure of the matrix into account. There are other approaches which maximize the diagonal values in some sense. One possibility is to look for a matrix $P_{\mathcal{M}}$ such that the product of the diagonal values of $P_{\mathcal{M}}A$ is maximal. In other words, a permutation σ has to be found, which maximizes

$$\prod_{i=1}^{n} |a_{\sigma(i)i}|. \tag{7.2}$$

This maximization problem is solved indirectly. It can be reformulated by defining a matrix $C = (c_{ij})$ with

$$c_{ij} = \begin{cases} \log a_i - \log |a_{ij}| & a_{ij} \neq 0, \\ \infty & \text{otherwise,} \end{cases} \tag{7.3}$$

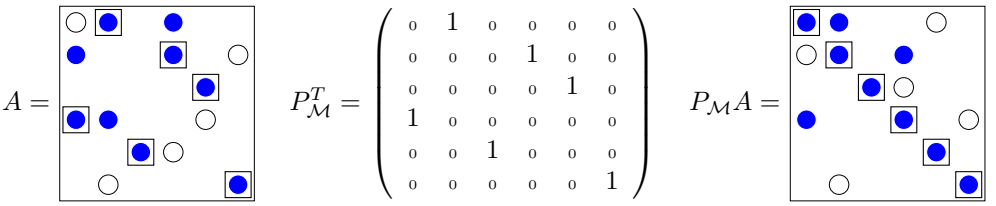


FIG. 3. *Illustration of the row permutation. A small numerical value is indicated by ∘ and a large numerical value by •. The matched entries $\mathcal{M}$ are marked with squares, and* $P_{\mathcal{M}} = (e_4; e_1; e_5; e_2; e_3; e_6)$.

where $a_i = \max_j |a_{ij}|$, i.e., the maximum element in row i of matrix A. A permutation σ, which minimizes $\sum_{i=1}^{n} c_{\sigma(i)i}$, also maximizes the product (7.2).

The minimization problem is known as the linear sum assignment problem or the bipartite weighted matching problem in combinatorial optimization. The problem is solved by a sparse variant of the Kuhn–Munkres algorithm. The complexity is $O(n^3)$ for full $n \times n$ matrices and $O(n\tau \log n)$ for sparse matrices with τ entries. For matrices whose associated graph fulfills special requirements, this bound can be reduced further to $O\left(n^{\alpha}(\tau + n \log n)\right)$ with $\alpha < 1$. All graphs arising from finite-difference or finite-element discretizations meet these conditions [37]. As before, we finally get a perfect matching $\mathcal{M}$ that in turn defines a nonsymmetric permutation $P_{\mathcal{M}}$.

The effect of nonsymmetric row permutations using a permutation associated with a matching $\mathcal{M}$ is shown in Figure 3. It is clearly visible that the matrix $P_{\mathcal{M}}A$ is now nonsymmetric, but has the largest nonzeros on the diagonal.

7.2. Symmetric 1×1 and 2×2 block weighted matchings. In the case of symmetric indefinite matrices, we are interested in symmetrically permuting PAP^T. The problem is that zero or small diagonal elements of A remain on the diagonal when we use a symmetric permutation PAP^T. Alternatively, instead of permuting a large[1] off-diagonal element a_{ij} nonsymmetrically to the diagonal, we can try to devise a permutation $P_{\mathcal{S}}$ such that $P_{\mathcal{S}}AP_{\mathcal{S}}^T$ permutes this element close to the diagonal. As a result, if we form the corresponding 2×2 block to $\left[\begin{smallmatrix} a_{ii} & a_{ij} \\ a_{ij} & a_{jj} \end{smallmatrix}\right]$, we expect the off-diagonal entry a_{ij} to be large, and thus the 2×2 block would form a suitable 2×2 pivot for the supernode Bunch–Kaufman factorization. An observation on how to build $P_{\mathcal{S}}$ from the information given by a weighted matching $\mathcal{M}$ was presented by Duff and Gilbert [22]. They noticed that the cycle structure of the permutation $P_{\mathcal{M}}$ associated with the nonsymmetric matching $\mathcal{M}$ can be exploited to derive such a permutation $P_{\mathcal{S}}$. For example, the permutation $P_{\mathcal{M}}$ from Figure 3 can be written in cycle representation as $P_{\mathcal{C}} = (e_1; e_2; e_4)(e_3; e_5)(e_6)$. This is shown in the upper graphics in Figure 4. The left graphic displays the cycles (1 2 4), (3 5), and (6). If we modify the original permutation $P_{\mathcal{M}} = (e_4; e_1; e_5; e_2; e_3; e_6)$ into this cycle permutation $P_{\mathcal{C}} = (e_1; e_2; e_4)(e_3; e_5)(e_6)$ and permute A symmetrically with $P_{\mathcal{C}}AP_{\mathcal{C}}^T$, it can be observed that the largest elements are permuted to diagonal blocks. These diagonal blocks are shown by filled boxes in the upper right matrix. Unfortunately, a long cycle would result in a large diagonal block, and the fill-in of the factor for $P_{\mathcal{C}}AP_{\mathcal{C}}^T$ may be prohibitively large. Therefore, long cycles corresponding to $P_{\mathcal{M}}$ must be broken down into disjoint 2×2 and 1×1 cycles. These smaller cycles are used to define a symmetric permutation $P_{\mathcal{S}} = (c_1, \ldots, c_m)$, where m is the total number of 2×2 and

[1] Large in the sense of the weighted matching $\mathcal{M}$.

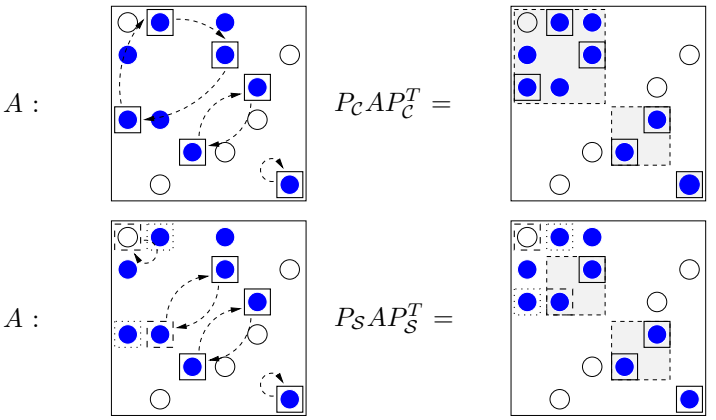


FIG. 4. *Illustration of a cycle permutation with* $P_C = (e_1; e_2; e_4)(e_3; e_5)(e_6)$ *and* $P_{\mathcal{S}} = (e_1)(e_2; e_4)(e_3; e_5)(e_6)$. *The symmetric matching* $P_{\mathcal{S}}$ *has two additional elements (indicated by dashed boxes), while one element of the original matching fell out (dotted box). The two* 2*-cycles are permuted into* 2×2 *diagonal blocks to serve as initial* 2×2 *pivots.*

1×1 cycles.

The rule for choosing the 2×2 and 1×1 cycles from $P_{\mathcal{C}}$ to build $P_{\mathcal{S}}$ is straightforward. One has to distinguish between cycles of even and odd length. It is always possible to break down even cycles into cycles of length 2. For each even cycle, there are two possible ways to break it down. We use a structural metric [24] to decide which one to take. The same metric is also used for cycles of odd length, but the situation is slightly different. Cycles of length $2l + 1$ can be broken down into l cycles of length 2 and one cycle of length 1. There are $2l + 1$ possible ways to do this. The resulting 2×2 blocks will contain the matched elements of $\mathcal{M}$. However, there is no guarantee that the remaining diagonal element corresponding to the cycle of length 1 will be nonzero. Our implementation will randomly select one element as a 1×1 cycle from an odd cycle of length $2l + 1$.

A selection of $P_{\mathcal{S}}$ from a weighted matching $P_{\mathcal{M}}$ is illustrated in Figure 4. The permutation associated with the weighted matching, which is sorted according to the cycles, consists of $P_{\mathcal{C}} = (e_1; e_2; e_4)(e_3; e_5)(e_6)$. We now split the full cycle of odd length 3 into two cycles (1)(24)—resulting in $P_{\mathcal{S}} = (e_1)(e_2; e_4)(e_3; e_5)(e_6)$. If $P_{\mathcal{S}}$ is symmetrically applied to $A \leftarrow P_{\mathcal{S}} A P_{\mathcal{S}}^T$, we see that the large elements from the nonsymmetric weighted matching $\mathcal{M}$ will be permuted close to the diagonal, and these elements will have more chances to form good initial 1×1 and 2×2 pivots for the subsequent (incomplete) factorization.

Good fill-in reducing orderings P_{Fill} are equally important for symmetric indefinite systems. The following section introduces two strategies for combining these reorderings with the symmetric graph matching permutation $P_{\mathcal{S}}$. This will provide good initial pivots for the factorization as well as a good fill-in reduction permutation.

7.3. Combination of orderings P_{Fill} for fill reduction with orderings $P_{\mathcal{S}}$ based on weighted matchings. In order to construct the factorization efficiently, care has to be taken that not too much fill-in is introduced during the elimination process. We now examine two algorithms for the combination of a permutation $P_{\mathcal{S}}$ based on weighted matchings to improve the numerical quality of the coefficient matrix A with a fill-in reordering P_{Fill} based on a nested dissection from METIS [39]. The first method is based on compressed subgraphs and has also been used by Duff and

Pralet in [24] in order to find good scalings and orderings for symmetric indefinite systems.

In order to combine the permutation $P_{\mathcal{S}}$ with a fill-in reducing permutation, we compress the graph of the reordered system $P_{\mathcal{S}}AP_{\mathcal{S}}^T$ and apply the fill-in reducing reordering to the compressed graph. In the compression step, the union of the structure of the two rows and columns corresponding to a 2×2 diagonal block is built and used as the structure of a single, compressed row and column representing the original ones.

If $G_A = (V; E)$ is the undirected graph of A and a cycle consists of two vertices $(s, t) \in V$, then graph compression will be done on the 1×1 and 2×2 cycles, which have been found using a weighted matching $\mathcal{M}$ on the graph. The vertices (s, t) are replaced with a single supervertex $u = \{s, t\} \in V_c$ in the compressed graph $G_c = (V_c, E_c)$. An edge $e_c = (s, t) \in E_c$ between two supervertices $s = \{s_1, s_2\} \in V_c$ and $t = \{t_1, t_2\} \in V_c$ exists if at least one of the following edges exists in E: (s_1, t_1), (s_1, t_2), (s_2, t_1), or (s_2, t_2). The fill-in reducing ordering is found by applying METIS on the compressed graph $G_c = (V_c, E_c)$. Expansion of P_{Fill} to the original numbering yields the final permutation. Hence all 2×2 cycles that correspond to a suitable 2×2 pivot block are reordered consecutively in the factor.

8. Symmetric multilevel preconditioning techniques. We now present a new symmetric indefinite approximate multilevel factorization that is mainly based on three parts which are repeated in a multilevel framework in each subsystem. The components consist of (i) reordering of the system, (ii) approximate factorization using inverse-based pivoting, and (iii) recursive application to the system of postponed updates.

8.1. Reordering the given system. The key ingredient for turning this approach into an efficient multilevel solver consists of the symmetric maximum weight matching presented in section 6.2. After the system is reordered into a representation

$$P_s^T DADP_s = \hat{A}, \tag{8.1}$$

where $D, P_s \in \mathbb{R}^{n,n}$, D is a diagonal matrix, and P_s is a permutation matrix, $\hat{A}$ is expected to have many diagonal blocks of size 1×1 or 2×2 that are well conditioned. Once the diagonal blocks of size 1×1 and 2×2 are built, the associated block graph of $\hat{A}$ is reordered by a symmetric reordering, e.g., AMD [1] or METIS [39], i.e.,

$$\Pi^T P_s^T DADP_s \Pi = \tilde{A}, \tag{8.2}$$

where $\Pi \in \mathbb{R}^{n,n}$ refers to the associated symmetric block permutation.

8.2. Inverse-based pivoting. Given $\tilde{A}$ we compute an incomplete factorization $LDL^T = \tilde{A} + E$ of $\tilde{A}$. To do this at step k of the algorithm we have

$$\tilde{A} = \begin{pmatrix} B & F^T \\ F & C \end{pmatrix} = \begin{pmatrix} L_B & 0 \\ L_F & I \end{pmatrix} \begin{pmatrix} D_B & 0 \\ 0 & S_C \end{pmatrix} \begin{pmatrix} L_B^T & L_F^T \\ 0 & I \end{pmatrix}, \tag{8.3}$$

where $L_B \in \mathbb{R}^{k-1,k-1}$ is lower triangular with unit diagonal and $D_B \in \mathbb{R}^{k-1,k-1}$ is block diagonal with diagonal blocks of sizes 1×1 and 2×2. Also, $S_C = C - L_F D_B L_F^T = (s_{ij})_{i,j}$ denotes the approximate Schur complement. To proceed with the incomplete factorization we perform either a 1×1 update or a 2×2 block update. One possible choice could be to use Bunch's algorithm [11]. This approach has been used in [38].

Here we use a simple criterion based on block diagonal dominance of the leading block column. Depending on the values

$$d_1 = \sum_{j>1} \frac{|s_{j1}|}{|s_{11}|}, \quad d_2 = \sum_{j>2} \left\| (s_{j1}, s_{j2}) \begin{pmatrix} s_{11} & s_{12} \\ s_{12} & s_{22} \end{pmatrix}^{-1} \right\|, \tag{8.4}$$

we perform a 2×2 update only if $d_2 < d_1$. The two leading columns of S_C can be efficiently computed using linked lists [43], and it is not required to have all entries of S_C available.

When applying the (incomplete) factorization LDL^T to $\tilde{A}$ we may still encounter a situation where at step k either $1/|s_{11}|$ or $\|(s_{ij})^{-1}_{i,j\leqslant 2}\|$ is large or even infinite. Since we are dealing with an incomplete factorization we propose to use inverse-based pivoting [8]. Therefore, we require in every step that

$$\left\| \begin{pmatrix} L_B & 0 \\ L_F & I \end{pmatrix}^{-1} \right\| \leqslant \kappa \tag{8.5}$$

for a prescribed bound κ. If after the update using a 1×1 pivot (or 2×2 pivot) the norm of the inverse lower triangular factor fails to be less than κ, the update is postponed and the leading rows/columns of L_F are permuted to the end of S_C. Otherwise, depending on whether a 1×1 or a 2×2 pivot has been selected, the entries

$$(s_{j1}/s_{11})_{j>1}, \quad \left((s_{j1},\ s_{j2}) \begin{pmatrix} s_{11} & s_{12} \\ s_{12} & s_{22} \end{pmatrix}^{-1} \right)_{j>2} \tag{8.6}$$

become the next (block) column of L, and we drop these entries whenever their absolute value is less than ε/κ for some threshold ε. For a detailed description see [8]. The norm of the inverse can be cheaply estimated using a refined strategy of [15] and is part of the software package ILUPACK that is now extended to the symmetric indefinite case [9].

8.3. Recursive application. After the inverse-based ILU we have an approximate factorization

$$Q^T \tilde{A} Q = \begin{pmatrix} L_{11} & 0 \\ L_{21} & I \end{pmatrix} \begin{pmatrix} D_{11} & 0 \\ 0 & S_{22} \end{pmatrix} \begin{pmatrix} L_{11}^T & L_{21}^T \\ 0 & I \end{pmatrix}, \tag{8.7}$$

and it typically does not pay off to continue the factorization for the remaining matrix S_{22} which consists of the previously postponed updates. Thus S_{22} is now explicitly computed and the strategies for reordering, scaling, and factorization are recursively applied to S_{22}, leading to a multilevel factorization.

Note that in order to save memory, L_{21} is not stored but implicitly approximated by $\tilde{A}_{21}(L_{11}D_{11}L_{11}^T)^{-1}$. In addition we use a technique called [illegible] that sparsifies the triangular factor L a posteriori. To do this observe that when applying a perturbed triangular factor $\tilde{L}^{-1}$ for preconditioning, instead of L^{-1} we have

$$\tilde{L}^{-1} = (I + E_L)L^{-1}, \text{ where } E_L = \tilde{L}^{-1}(L - \tilde{L}).$$

We can expect that $\tilde{L}^{-1}$ serves as a good approximation to L^{-1} as long as $\|E_L\| \ll 1$. If we obtain $\tilde{L}$ from L by dropping some entry, say l_{ij} from L, then we have to ensure that

$$\|\tilde{L}^{-1}e_i\| \cdot |l_{ij}| \leqslant \tau \ll 1$$

for some moderate constant $\tau < 1$, e.g., $\tau = 0.1$. To do this requires having a good estimate for $\nu_i \approx \|\tilde{L}^{-1}e_i\|$ available for any $i = 1, \ldots, n$. In principle it can be computed [8, 15] using $\tilde{L}^T$ instead of $\tilde{L}$. Finally, knowing how many entries exist in column j, we could drop any l_{ij} such that

$$|l_{ij}| \leqslant \tau/(\nu_i \cdot \#\{l_{kj} : \ l_{kj} \neq 0, k = j+1, \ldots, n\}).$$

8.4. Iterative solution. By construction, the computed incomplete multilevel factorization is symmetric but indefinite. For the iterative solution of linear systems using the multilevel factorization, in principle different Krylov subspace solvers could be used, such as general methods that do not explicitly use symmetry (e.g., GMRES [56]) or methods like SYMMLQ [50] which preserve the symmetry of the original matrix but which are devoted only to symmetric positive definite preconditioners. To fully exploit both symmetry and indefiniteness at the same time, here the simplified QMR method [29, 30] is chosen.

9. Numerical experiments. Here we present numerical experiments that show that the previously outlined advances in symmetric indefinite sparse direct solvers as well as in preconditioning methods significantly accelerate modern eigenvalue solvers and allow us to gain orders of magnitude in speed compared to more conventional methods.

9.1. Computing environments and software. All large-scale numerical experiments for the Anderson model of localization were performed on an SGI Altix 3700/BX2 with 56 Intel Itanium2 1.6 GHz processors and 112 GB of memory. If not explicitly stated, we always used only one processor of the system and all algorithms were implemented in either C or Fortran77. All codes were compiled by the Intel V8.1 compiler suite using [illegible] and [illegible] with the `-O3` optimization option and linked with basic linear algebra subprograms optimized for Intel architectures. The computations for $M = 250,350$ and $w = 16.5$ required 64-bit long integers and `-i8` flag for [illegible]. From comparison with smaller examples we observed an overhead of approximately 30% with respect to memory and computation time. For completeness, let us recall the main software packages used:

- ARPACK is a collection of Fortran77 subroutines designed to solve large-scale eigenvalue problems. The eigenvalue solver has been developed at the Department of Computational and Applied Mathematics at Rice University. It is available at http://www.caam.rice.edu/software/ARPACK.
- JDBSYM is a C library implementation of the JACOBI–DAVIDSON method optimized for symmetric eigenvalue problems. It solves eigenproblems of the form $Ax = \lambda x$ and $Ax = \lambda Bx$ with or without preconditioning, where A is symmetric and B is symmetric positive definite. It has been developed at the Computer Science Department of the ETH Zürich. It is available at http://people.web.psi.ch/geus/software.html.
- PARDISO is a fast direct solver package, developed at the Computer Science Department of the University of Basel. It is available at http://www.computational.unibas.ch/cs/scicomp/software/pardiso.
- ILUPACK is an algebraic multilevel preconditioning software package. This iterative solver has been developed at the Mathematics Department of the Technical University of Berlin. It is available at http://www.math.tu-berlin.de/ilupack.

9.2. CWI compared to shift-and-invert Lanczos with implicit restarts and PARDISO as direct solver. Let us first briefly compare the classical CWI with the shift-and-invert Lanczos method using implicit restarts. The latter is part of ARPACK [42]. For the solution of the symmetric indefinite system $A - \theta I$ we use the most recent version of sparse direct solver PARDISO [57]. This version is based on symmetric weighted matchings and uses METIS as a symmetric reordering strategy. The numerical results deal with the computation of five eigenvalues of the Anderson matrix A near $\lambda = 0$. Here we state the results for the physically most interesting critical disorder strength $w_c = 16.5$. We have measured the CPU times in seconds and memory requirements in GB to compute five eigenvalues closest to $\lambda = 0$ of an Anderson matrix of size $N = M^3 \times M^3$ up to $M = 100$ with CWI and ARPACK–PARDISO. We observe from this initial numerical experiment that the combination of the shift-and-invert Lanczos with PARDISO is faster when compared to the CWI by about a factor of 10 for systems with $M > 50$. Despite this success, with increasing problem size the amount of memory consumed by the sparse direct solver becomes significant[2] and numerical results with N larger than 1000000 are skipped. Figure 1 shows two different eigenstates computed with the help of PARDISO.

9.3. Using the ILUPACK-based preconditioner. We now switch to the ILUPACK-based preconditioner that is also based on symmetric weighted matchings and in addition uses inverse-based pivoting. In particular, for our experiments we use $\kappa = 5$ as a bound for the norm $\|L^{-1}\|$ of the inverse triangular factor and AMD for the symmetric reordering. We also tried to use METIS, but for this particular matrix problem we find that AMD is clearly more memory efficient. Next we compare the shift-and-invert Lanczos (ARPACK) with ILUPACK and the simplified QMR as the inner iterative solver. Here we use $\varepsilon = 1/\sqrt{N}$ with aggressive dropping, and the QMR method is stopped once the norm of residual satisfies $\|Ax - b\| \leqslant 10^{-10}\|b\|$. In order to illustrate the benefits of using symmetric weighted matchings we also tried ILUPACK without matching, but the numerical results are disappointing, as can be seen from the †'s in Table 9.1. We emphasize that the multilevel approach is crucial; a simple use of incomplete factorization methods without multilevel preconditioning [38] does not give the desired results. Besides the effect of matchings we also compare how the performance of the methods changes when varying the value w from the critical value $w = w_c = 16.5$ to $w = 12.0$ and $w = 21.0$. We find that these changes do not affect the sparse direct solver at all while the multilevel ILU significantly varies in its performance. Up to now our explanation for this effect is the observation that with increasing w the diagonal dominance of the system also increases and the ILUPACK preconditioner gains from higher diagonal dominance. As we can see from Table 9.1, ILUPACK still uses significantly less memory than the direct solver PARDISO for all values of w, and it is the only method we were able to use for larger N due to the memory constraints. Also, the computation time is best.

9.4. Using JACOBI–DAVIDSON. When using preconditioning methods inside shift-and-invert Lanczos we usually have to solve the inner linear system for $A - \theta I$ up to machine precision to make sure that the eigenvalues and eigenvectors are sufficiently correct. In contrast to this the JACOBI–DAVIDSON method allows us to solve the associated correction equation less accurately, and only when convergence takes place is a more accurate solution required. In order to show the significant

[2]The current standard memory of 2GB RAM for a desktop computer is exceeded for sizes beyond $M > 64$.

TABLE 9.1

CPU times in seconds and memory requirements in GB to compute five eigenvalues closest to $\lambda = 0$ of an Anderson matrix of size $M^3 \times M^3$ with ARPACK–PARDISO, ARPACK–ILUPACK, *and* ARPACK–ILUPACK–SYMMATCH. *The symbol — indicates that a memory consumption was larger than* 25 *GB, and † indicates memory problems with respect to the fill-in.*

M	W	ARPACK					
		PARDISO		ILUPACK		ILUPACK–SYMMATCH	
		Time	Mem.	Time	Mem.	Time	Mem.
70	12.0	1359	3.00	5117	1.09	2140	0.95
100	12.0	20639	14.34	39222	5.62	13583	3.20
130	12.0	—	—	†	†	65722	8.20
70	16.5	1305	3.00	504	0.33	477	0.31
100	16.5	20439	14.34	2349	0.95	2177	0.89
130	16.5	—	—	6320	2.09	6530	1.95
160	16.5	—	—	23663	3.95	13863	3.63
70	21.0	1225	3.00	371	0.22	310	0.22
100	21.0	20239	14.34	1513	0.64	1660	0.65
130	21.0	—	—	3725	1.41	3527	1.44
160	21.0	—	—	15302	2.63	20120	2.68

TABLE 9.2

Number of inner/outer interaction steps inside ARPACK *and* JACOBI–DAVIDSON. *The symbol — indicates that the computations were not performed anymore for* ARPACK.

M	W	ILUPACK–SYMMATCH					
		ARPACK			JACOBI–DAVIDSON		
		Outer	Total	Inner average	Outer	Total	Inner average
70	12.0	42	871	20.7	20	218	10.9
100	12.0	43	1101	25.6	17	228	13.4
130	12.0	42	1056	25.1	25	272	10.9
70	16.5	43	611	14.2	18	167	9.3
100	16.5	43	857	19.9	19	193	10.2
130	16.5	42	1058	25.2	19	271	14.3
160	16.5	42	968	23.1	15	223	14.9
190	16.5	—	—	—	19	297	15.6
220	16.5	—	—	—	22	446	20.3
250	16.5	—	—	—	17	463	27.2
350	**16.5**	—	—	—	**16**	**457**	**28.6**
70	21.0	43	585	13.6	18	167	9.3
100	21.0	42	1004	23.9	18	268	14.9
130	21.0	44	914	20.8	16	243	15.2
160	21.0	25	896	35.8	20	398	19.9
190	21.0	—	—	—	21	527	25.1
220	21.0	—	—	—	17	639	37.6
250	21.0	—	—	—	12	502	41.8

difference between the iterative parts of ARPACK and JACOBI–DAVIDSON we state the number of iteration steps in Table 9.2. If we were to aim for more eigenpairs, we would expect that eventually the JDBSYM would become less efficient and should again be replaced by ARPACK. As stopping criteria for the inner iteration inside the JACOBI–DAVIDSON method we use recent results from [48, 65]. Given the eigenvector residual

$$r_{eig} = Au - u\lambda, \text{ where } \|u\| = 1,\ \lambda = u^T Au,$$

and given an approximate solution z of the correction equation (5.4) with the associated linear system residual

$$r_{lin} = -r_{eig} - (I - uu^T)(A - \theta I)(I - uu^T)z,$$

one could define a new approximate eigenvector via $u_{eignew} = (u + z)/\|u + z\|$. Following [48] the associated new eigenvector residual r_{eignew} can be bounded by

$$\frac{|\|r_{lin}\| - \beta\|z\||}{1 + \|z\|^2} \leqslant \|r_{eignew}\| \leqslant \frac{\sqrt{(\|r_{lin}\| + \beta\|z\|)^2 + \|r_{lin}\|^2\|z\|^2}}{1 + \|z\|^2},$$

where $\beta = |\lambda - \theta + r_{eig}^T z|$. Numerical experiments in [48] indicate that initially

$$\beta\|z\| \ll \|r_{lin}\| \Rightarrow \|r_{eignew}\| \approx \frac{\|r_{lin}\|}{\sqrt{1 + \|z\|^2}},$$

while asymptotically we expect r_{lin} to converge to zero leading to

$$\beta\|z\| \gg \|r_{lin}\| \Rightarrow \|r_{eignew}\| \approx \frac{\beta\|z\|}{1 + \|z\|^2}.$$

When z is obtained from the simplified QMR algorithm as in our case, it has been shown in [65] that $\|r_{eignew}\|$ and $r_{eig}^T z$ can be cheaply computed as a by-product of the simplified QMR algorithm. In addition, in practice $\|z\|$ need not be recomputed throughout the iteration, since after a few steps, $\|z\|$ typically does not vary too much anymore [48]. This motivates our stopping criterion, where we stop the inner iteration inside QMR whenever

$$\frac{\|r_{lin}\|}{\sqrt{1 + \|z\|^2}} \leqslant \min\left\{\|r_{eignew}\|, \frac{\beta\|z\|}{1 + \|z\|^2}, \frac{\tau}{2}\right\}.$$

Here τ is the desired tolerance of the eigenvector residual (10^{-10} in our experiments). Note also that $\|r_{lin}\|$ is not explicitly available in the QMR method. Thus we use the quasi residual as an estimate and check only the true residual on exit to safeguard the process.

In what follows we compare the traditional CWI method with the JACOBI–DAVIDSON code JDBSYM [33] using ILUPACK as a preconditioner. Table 9.3 shows that switching from ARPACK to JACOBI–DAVIDSON in this case improves the total time by another factor of 6 or greater. For this reason JACOBI–DAVIDSON together with ILUPACK will be used as a default solver in the following. The numerical results in Table 9.3 show a dramatic improvement in the computation time by using ILUPACK-based JACOBI–DAVIDSON. Although this new method slows down for smaller w due to poorer diagonal dominance, a gain by orders of magnitude can still be observed. For $w = 16.5$ and larger, even more than three orders of magnitude in the computation time can be observed. Hence the new method drastically outperforms the CWI method while the memory requirement is still moderate. Figure 2 shows an eigenstate computed within three days with the help of the ILUPACK-based JACOBI–DAVIDSON. The construction of the ILUPACK preconditioner needed 14 hours at a fill-in factor of 18 compared with the fill of the original matrix.

One key to the success of the preconditioner is based on the threshold κ which bounds the growth of L^{-1}. Already, for a small example such as $M = 70$ significant differences can be observed. As we show in Table 9.4, increasing the bound by a factor of 2 from $\kappa = 5$ up to $\kappa = 10$ and $\kappa = 20$ leads to an enormous increase in fill. Here we measure the fill of the incomplete LDL^T factorization relative to the nonzeros of the original matrix. By varying the drop tolerance ε we also see that the dependence

TABLE 9.3

CPU times in seconds and memory requirements in GB to compute five eigenvalues closest to $\lambda = 0$ *with* CWI *and* JACOBI–DAVIDSON *using* ILUPACK–SYMMATCH *for the shift-and-invert technique.* ‡ *indicates that the convergence of the method was too slow. For* CWI *and* $M = 100$*, not all five eigenvalues converged successfully, so the eigenvector reconstruction finished more quickly, leading to variances in the CPU times* (*)*. Computations for* $M = 250,350$*, and* $w = 16.5$ *were computed with* 64 *bit long integers* (^{64})*.*

M	W	CWI		JACOBI–DAVIDSON ILUPACK–SYMMATCH	
		Time	Mem.	Time	Mem.
70	12.0	20228	0.11	1138	0.9
100	12.0	148843	0.32	7238	3.1
130	12.0	‡	‡	52774	9.0
70	16.5	15100	0.11	161	0.3
100	16.5	255842*	0.32	661	1.0
130	16.5	‡	‡	2000	2.4
160	16.5	‡	‡	3961	4.8
190	16.5	‡	‡	10955	8.1
220	16.5	‡	‡	25669	12.3
250	16.5	‡	‡	57203^{64}	26.0^{64}
350	**16.5**	‡	‡	**182276**^{64}	**88.0**^{64}
70	21.0	14371	0.11	99	0.3
100	21.0	331514*	0.32	484	0.8
130	21.0	‡	‡	1069	1.6
160	21.0	‡	‡	3070	3.2
190	21.0	‡	‡	8564	5.6
220	21.0	‡	‡	17259	8.5
250	21.0	‡	‡	24802	12.6

TABLE 9.4

The influence of the inverse bound κ *on the amount of memory. For* $M = 70$*, compare for different thresholds how the fill-in* $nnz(LDL^T)/nnz(A)$ *varies depending on* κ *and state the computation time in seconds.*

ε	$\kappa = 5$			$\kappa = 10$			$\kappa = 20$		
	Fill	Time LDL^T	Total time	Fill	Time LDL^T	Total time	Fill	Time LDL^T	Total time
0.01	5.4	37	870	8.7	67	500	15.2	160	480
0.005	6.8	54	440	11.0	100	380	19.1	230	500
0.0025	8.6	81	310	13.8	150	360	24.1	340	600
0.001	11.7	130	300	18.0	230	410	32.1	540	780

on κ is much more significant than the dependence on ε. Roughly speaking, the ILU decomposition becomes twice as expensive when κ is replaced with 2κ, as does the fill-in. The latter is crucial since memory constraints severely limit the size of the application that can be computed.

9.5. Hard wall boundaries and randomness in the off-diagonal matrix elements. In Table 9.5 we show how JDBSYM and ILUPACK–SYMMATCH perform when, instead of periodic boundary conditions, we use hard wall boundaries, i.e., $x_{0;j;k} = x_{i;0;k} = x_{i;j;0} = x_{M+1;j;k} = x_{i;M+1;k} = x_{i;j;M+1} = 0$ for all i, j, k. This is sometimes of interest in the Anderson problem, and, generally, it is expected that for large M, the difference in eigenvalues and eigenvectors becomes small when compared to the standard periodic boundaries. In addition, we also show results for the so-called off-diagonal Anderson problem [14]. Here, we shift the diagonal to a constant $\sigma = 1.28$ and incorporate the randomness by setting the off-diagonal elements of

TABLE 9.5

Difference in performance for our standard problem with periodic boundary conditions, the problem with hard wall conditions, and the inverse problem with random numerical entries in the off-diagonal elements. Memory requirement (in GB) and CPU times (in seconds) to compute at the transition the eigenvectors corresponding to the five eigenvalues closest to $\lambda = 0$ with shift-and-invert JACOBI–DAVIDSON *and the* ILUPACK–SYMMATCH *solver using symmetric weighted matchings.*

N	Periodic		Hard wall		Inverse	
	Time	Memory	Time	Memory	Time	Memory
70	161	0.3	169	0.3	251	0.2
100	661	1.0	704	0.9	1055	0.8
130	2000	2.4	1566	2.4	3203	1.8
160	3961	4.8	4078	4.6	11614	3.3
190	10955	8.1	10922	7.8	29455	6.5

A to be uniformly distributed in $[-1/2, 1/2]$. The graph of the matrix A remains the same. These values correspond—similarly to $w_c = 16.5$ used before for purely diagonal randomness—to the physically most interesting localization transition in this model [14]. We note that using hard wall boundary conditions instead of periodic boundary conditions leads to slightly less fill but increases the number of iteration steps, as can be seen in Table 9.5. This conclusions carries over to the off-diagonal Anderson problem, where the memory consumption is less but the iterative part takes even longer. In principle our results could be improved if we were to switch to a smaller threshold ε than the uniformly applied $\varepsilon = 1/\sqrt{N}$ here.

10. Conclusion. We have shown that modern approaches to preconditioning based on symmetric matchings and multilevel preconditioning methods lead to an astonishing increase in performance and available system sizes for the Anderson model of localization. This approach is not only several orders of magnitudes faster than the traditional CWI approach, but it also consumes only a moderate amount of memory, thus allowing us to study the Anderson eigenproblem for significantly larger scales than ever before.

Let us briefly recall the main ingredients necessary for this progress. At the heart of the new approach lies the use of symmetric matchings [38] in the preconditioning stage of the inverse-based incomplete factorization preconditioning iterative method [9]. Furthermore, the preconditioning itself is of a multilevel type, complementary to the often used full-pivoting strategies. Next, the inverse-based approach is also of paramount importance to keep the fill-in at a manageable level (see Table 9.4). Finally, we emphasize that these results, of course, reflect our selected problem class: to compute a few of the interior eigenvalues and associated eigenvectors for a highly indefinite symmetric matrix defined by the Anderson model of localization.

The performance increase by several orders of magnitude (see Table 9.3) is solely due to our use of new and improved algorithms. Combined with advances in the performance-to-cost ratio of computing hardware during the past six years, current preconditions methods make it possible to solve those problems quickly and easily, which have been considered by far too large until recently [13].

Even for $N \times N$ matrices as large as $N = 64 \cdot 10^6$, it is now possible to compute within a few days the interior eigenstates of the Anderson problem.

The success of this method indicates that it might also be successfully applied to other large-scale problems arising in (quantum) physics.

Acknowledgments. We gratefully acknowledge discussions with and implementation help from A. Croy and C. Sohrmann in the initial stages of the project.

REFERENCES

[1] P. R. Amestoy, T. A. Davis, and I. S. Duff, *An approximate minimum degree ordering algorithm*, SIAM J. Matrix Anal. Appl., 17 (1996), pp. 886–905.

[2] P. W. Anderson, *Absence of diffusion in certain random lattices*, Phys. Rev., 109 (1958), pp. 1492–1505.

[3] H. Aoki, *Fractal dimensionality of wave functions at the mobility edge: Quantum fractal in the Landau levels*, Phys. Rev. B, 33 (1986), pp. 7310–7313.

[4] P. Arbenz and R. Geus, *Multilevel preconditioned iterative eigensolvers for Maxwell eigenvalue problems*, Appl. Numer. Math., 54 (2005), pp. 107–121.

[5] M. Benzi, *Preconditioning techniques for large linear systems: A survey*, J. Comput. Phys., 182 (2002), pp. 418–477.

[6] M. Benzi, G. H. Golub, and J. Liesen, *Numerical solution of saddle point problems*, Acta Numer., 14 (2005), pp. 1–137.

[7] M. Benzi, J. C. Haws, and M. Tůma, *Preconditioning highly indefinite and nonsymmetric matrices*, SIAM J. Sci. Comput., 22 (2000), pp. 1333–1353.

[8] M. Bollhöfer and Y. Saad, *Multilevel preconditioners constructed from inverse-based ILUs*, SIAM J. Sci. Comput., 27 (2006), pp. 1627–1650.

[9] M. Bollhöfer and O. Schenk, *ILUPACK Volume* 2.0*—Preconditioning Software Package for Symmetrically Structured Problems*, http://www.math.tu-berlin.de/ilupack/ (2005).

[10] T. Brandes, B. Huckestein, and L. Schweitzer, *Critical dynamics and multifractal exponents at the Anderson transition in* 3*d disordered systems*, Ann. Phys. (Leipzig), 5 (1996), pp. 633–651.

[11] J. R. Bunch, *Partial pivoting strategies for symmetric matrices*, SIAM J. Numer. Anal., 11 (1974), pp. 521–528.

[12] J. R. Bunch and L. Kaufman, *Some stable methods for calculating inertia and solving symmetric linear systems*, Math. Comp., 31 (1977), pp. 163–179.

[13] P. Cain, F. Milde, R. A. Römer, and M. Schreiber, *Use of cluster computing for the Anderson model of localization*, Comput. Phys. Comm., 147 (2002), pp. 246–250.

[14] P. Cain, R. A. Römer, and M. Schreiber, *Phase diagram of the three-dimensional Anderson model of localization with random hopping*, Ann. Phys. (Leipzig), 8 (1999), pp. SI33–SI38.

[15] A. K. Cline, C. B. Moler, G. W. Stewart, and J. H. Wilkinson, *An estimate for the condition number of a matrix*, SIAM J. Numer. Anal., 16 (1979), pp. 368–375.

[16] J. Cullum and R. A. Willoughby, *Lanczos Algorithms for Large Symmetric Eigenvalue Computations, Volume* 1: *Theory*, Birkhäuser Boston, Boston, MA, 1985.

[17] J. Cullum and R. A. Willoughby, *Lanczos Algorithms for Large Symmetric Eigenvalue Computations, Volume* 2: *Programs*, Birkhäuser Boston, Boston, MA, 1985. Available online at http://www.netlib.org/lanczos/.

[18] P. Dayal, M. Troyer, and R. Villiger, *The Iterative Eigensolver Template Library*, ETH Zürich, Zürich, Switzerland, 2004. Available online at http://www.comp-phys.org:16080/software/ietl/.

[19] J. W. Demmel, S. C. Eisenstat, J. R. Gilbert, X. S. Li, and J. W. H. Liu, *A supernodal approach to sparse partial pivoting*, SIAM J. Matrix Anal. Appl., 20 (1999), pp. 720–755.

[20] D. Dodson and J. G. Lewis, *Issues relating to extension of the basic linear algebra subprograms*, ACM SIGNUM Newslett., 20 (1985), pp. 19–22.

[21] J. J. Dongarra, J. Du Croz, S. Hammarling, and R. J. Hanson, *A proposal for an extended set of Fortran basic linear algebra subprograms*, ACM SIGNUM Newslett., 20 (1985), pp. 2–18.

[22] I. S. Duff and J. R. Gilbert, *Symmetric weighted matching for indefinite systems*, talk presented at the Householder Symposium XV, 2002.

[23] I. S. Duff and J. Koster, *The design and use of algorithms for permuting large entries to the diagonal of sparse matrices*, SIAM J. Matrix Anal. Appl., 20 (1999), pp. 889–901.

[24] I. S. Duff and S. Pralet, *Strategies for Scaling and Pivoting for Sparse Symmetric Indefinite Problems*, Technical Report TR/PA/04/59, CERFACS, Toulouse, France, 2004.

[25] I. S. Duff and J. K. Reid, *The multifrontal solution of indefinite sparse symmetric linear equations*, ACM Trans. Math. Software, 9 (1983), pp. 302–325.

[26] A. Eilmes, R. A. Römer, and M. Schreiber, *The two-dimensional Anderson model of localization with random hopping*, Eur. Phys. J. B, 1 (1998), pp. 29–38.

[27] U. Elsner, V. Mehrmann, F. Milde, R. A. Römer, and M. Schreiber, *The Anderson model of localization: A challenge for modern eigenvalue methods*, SIAM J. Sci. Comput., 20 (1999), pp. 2089–2102.

[28] R. D. Fokkema, G. L. G. Sleijpen, and H. A. Van der Vorst, *Jacobi–Davidson style QR and QZ algorithms for the reduction of matrix pencils*, SIAM J. Sci. Comput., 20 (1998), pp. 94–125.
[29] R. Freund and F. Jarre, *A QMR-based interior-point algorithm for solving linear programs*, Math. Programming Ser. B, 76 (1997), pp. 183–210.
[30] R. Freund and N. Nachtigal, *Software for simplified Lanczos and QMR algorithms*, Appl. Numer. Math., 19 (1995), pp. 319–341.
[31] A. George and E. Ng, *An implementation of Gaussian elimination with partial pivoting for sparse systems*, SIAM J. Sci. Statist. Comput., 6 (1985), pp. 390–409.
[32] R. Geus, *JDBSYM Version* 0.14, http://www.inf.ethz.ch/personal/geus/software.html (2006).
[33] R. Geus, *The Jacobi–Davidson Algorithm for Solving Large Sparse Symmetric Eigenvalue Problems with Application to the Design of Accelerator Cavities*, Ph.D. thesis, Computer Science Department, ETH Zürich, Zürich, Switzerland, 2002. Available online at http://www.inf.ethz.ch/personal/geus/publications/diss-online.pdf.
[34] J. R. Gilbert and E. Ng, *Predicting structure in nonsymmetric sparse matrix factorizations*, in Graph Theory and Sparse Matrix Computation, J. A. George, J. R. Gilbert, and J. W. H. Liu, eds., Springer, New York, 1993, pp. 107–139.
[35] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed., The Johns Hopkins University Press, Baltimore, MD, 1996.
[36] U. Grimm, R. A. Römer, and G. Schliecker, *Electronic states in topologically disordered systems*, Ann. Phys. (Leipzig), 7 (1998), pp. 389–393.
[37] A. Gupta and L. Ying, *A Fast Maximum-Weight-Bipartite-Matching Algorithm for Reducing Pivoting in Sparse Gaussian Elimination*, Tech. Report RC 21576 (97320), IBM T. J. Watson Research Center, Yorktown Heights, NY, 1999.
[38] M. Hagemann and O. Schenk, *Weighted matchings for preconditioning symmetric indefinite linear systems*, SIAM J. Sci. Comput., 28 (2006), pp. 403–420.
[39] G. Karypis and V. Kumar, *A fast and high quality multilevel scheme for partitioning irregular graphs*, SIAM J. Sci. Comput., 20 (1998), pp. 359–392.
[40] B. Kramer, A. Broderix, A. MacKinnon, and M. Schreiber, *The Anderson transition: New numerical results for the critical exponents*, Phys. A, 167 (1990), pp. 163–174.
[41] B. Kramer and A. MacKinnon, *Localization: Theory and experiment*, Rep. Progr. Phys., 56 (1993), pp. 1469–1564.
[42] R. B. Lehoucq, D. C. Sorensen, and C. Yang, *ARPACK Users' Guide: Solution of Large-Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods*, SIAM, Philadelphia, 1998. Available online at http://www.caam.rice.edu/software/ARPACK/.
[43] N. Li, Y. Saad, and E. Chow, *Crout versions of ILU for general sparse matrices*, SIAM J. Sci. Comput., 25 (2003), pp. 716–728.
[44] Q. Li, S. Katsoprinakis, E. N. Economou, and C. M. Soukoulis, *Scaling properties in highly anisotropic systems*, Phys. Rev. B, 56 (1997), pp. R4297–R4300,
[45] X. S. Li and J. W. Demmel, *SuperLU_DIST: A scalable distributed-memory sparse direct solver for unsymmetric linear systems*, ACM Trans. Math. Software, 29 (2003), pp. 110–140.
[46] F. Milde, R. A. Römer, and M. Schreiber, *Multifractal analysis of the metal-insulator transition in anisotropic systems*, Phys. Rev. B, 55 (1997), pp. 9463–9469.
[47] F. Milde, R. A. Römer, and M. Schreiber, *Energy-level statistics at the metal-insulator transition in anisotropic systems*, Phys. Rev. B, 61 (2000), pp. 6028–6035.
[48] Y. Notay, *Inner Iterations in Eigenvalue Solvers*, Technical Report GANMN 05-01, Université Libre de Bruxelles, Brussels, Belgium, 2005.
[49] M. Olschowka and A. Neumaier, *A new pivoting strategy for Gaussian elimination*, Linear Algebra Appl., 240 (1996), pp. 131–151.
[50] C. C. Paige and M. A. Saunders, *Solution of sparse indefinite systems of linear equations*, SIAM J. Numer. Anal., 12 (1975), pp. 617–629.
[51] B. N. Parlett, *The Symmetric Eigenvalue Problem*, Prentice-Hall, Englewood Cliffs, NJ, 1980.
[52] I. Plyushchay, R. A. Römer, and M. Schreiber, *Three-dimensional Anderson model of localization with binary random potential*, Phys. Rev. B, 68 (2003), 064201.
[53] D. Porath, G. Cuniberti, and R. Di Felice, *Charge transport in DNA-based devices*, in Long-Range Charge Transfer in DNA II, Topics in Current Chemistry 237, G. B. Schuster, ed., Springer, New York, 2004, p. 183.
[54] R. A. Römer and M. Schreiber, *Numerical investigations of scaling at the Anderson transition*, in The Anderson Transition and Its Ramifications—Localisation, Quantum Interference, and Interactions, T. Brandes and S. Kettemann, eds., Springer, Berlin, 2003, pp. 3–19.

[55] Y. Saad, *Iterative Methods for Sparse Linear Systems*, 2nd ed., SIAM, Philadelphia, 2003.
[56] Y. Saad and M. H. Schultz, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 856–869.
[57] O. Schenk and K. Gärtner, *On Fast Factorization Pivoting Methods for Symmetric Indefinite Systems*, Technical report, Computer Science Department, University of Basel, Basel, Switzerland, 2004. Electron. Trans. Numer. Anal., submitted.
[58] O. Schenk and K. Gärtner, *Solving unsymmetric sparse systems of linear equations with PARDISO*, J. Future Generation Computer Systems, 20 (2004), pp. 475–487.
[59] O. Schenk, S. Röllin, and A. Gupta, *The effects of unsymmetric matrix permutations and scalings in semiconductor device and circuit simulation*, IEEE Trans. Computer-Aided Design Integrated Circuits Systems, 23 (2004), pp. 400–411.
[60] M. Schreiber and M. Ottomeier, *Localization of electronic states in* $2D$ *disordered systems*, J. Phys.: Condens. Matter, 4 (1992), pp. 1959–1971.
[61] G. L. G. Sleijpen and H. A. Van der Vorst, *A Jacobi–Davidson iteration for linear eigenvalue problems*, SIAM J. Matrix Anal. Appl., 17 (1996), pp. 401–425.
[62] D. C. Sorensen, *Implicit application of polynomial filters in a k-step Arnoldi method*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 357–385.
[63] C. M. Soukoulis and E. N. Economou, *Off-diagonal disorder in one-dimensional systems*, Phys. Rev. B, 24 (1981), pp. 5698–5702.
[64] C. M. Soukoulis and E. N. Economou, *Fractal character of eigenstates in disordered systems*, Phys. Rev. Lett., 52 (1984), pp. 565–568.
[65] A. Stathopoulos, *Nearly Optimal Preconditioned Methods for Hermitian Eigenproblems under Limited Memory. Part* I: *Seeking One Eigenvalue*, Technical Report WM-CS-2005-03, College of William & Mary, Williamsburg, VA, 2005.

SIAM J. SCI. COMPUT.
Vol. 28, No. 3, pp. 984–1000
© 2006 Society for Industrial and Applied Mathematics

COMBINED COMPLEX RIDGELET SHRINKAGE AND TOTAL VARIATION MINIMIZATION*

JIANWEI MA† AND MARKUS FENN‡

Abstract. A new algorithm for the characterization of engineering surface topographies with line singularities is proposed. It is based on thresholding complex ridgelet coefficients combined with total variation (TV) minimization. The discrete ridgelet transform is designed by first using a discrete Radon transform based on the nonequispaced fast Fourier transform (NFFT) and then applying a dual-tree complex wavelet transform (DT CWT). The NFFT-based approach of the Radon transform completely avoids linear interpolations of the Cartesian-to-polar grid and requires only $\mathcal{O}(n^2 \log n)$ arithmetic operations for n by n arrays, while its inverse preserves the good reconstruction quality of the filtered backprojection. The DT CWT in the second step of the ridgelet transform provides approximate shift invariance on the projections of the Radon transform. After hard thresholding the ridgelet coefficients, they are restored using TV minimization to eliminate the pseudo-Gibbs artifacts near the discontinuities. Numerical experiments demonstrate the remarkable ability of the methodology to extract line scratches.

Key words. nonequispaced fast Fourier transform, ridgelets, complex wavelets, shift invariance, total variation minimization, detection of line singularities, surface characterization

AMS subject classifications. 65T60, 65K10, 68U10, 94A08

DOI. 10.1137/05062737X

1. Introduction.

Surface topography is one of the most important factors affecting the functional performance of components. The functional topographical features such as peaks/pits and ridges/valleys impact directly on mechanical and physical properties of the whole system such as wear, friction, lubrication, corrosion, fatigue, coating, paintability, etc. For instance, during the functional operation of interacting surfaces, peaks and ridges will act as sites of high contact stresses and abrasion, and consequently, wear particles and debris will be generated by such surface topographical features, whereas the pits and valleys will affect the lubrication and fluid retention properties. Therefore, a highly accurate separation and extraction of different topographical features from surfaces for corresponding functional analysis is an important issue in the initial stages of engineering systems.

Over the past few years, wavelet methods have been applied for the analysis of functional surfaces including nanosurfaces. Unfortunately, the usual discrete wavelet transform (DWT) suffers from shift sensitivity, poor directionality, and pseudo-Gibbs artifacts. Recently, a dual-tree complex wavelet transform (DT CWT) combined with total variation (TV) minimization has been applied by Ma [23] and Ma, Jiang, and Blunt [24] to solve the above problems. However, wavelet-based techniques perform poorly when representing line singularities. In this paper, we pay attention to extract line scratches from engineering surfaces by applying the discrete ridgelet transform.

Ridgelets have been designed by Candès [3] and Candès and Donoho [5] to deal

*Received by the editors March 23, 2005; accepted for publication (in revised form) January 24, 2006; published electronically June 23, 2006.
http://www.siam.org/journals/sisc/28-3/62737.html
†Oxford Centre for Industrial and Applied Mathematics, Mathematical Institute, University of Oxford, 24-29 St. Giles', Oxford, OX1 3LB, UK. Current address: Laboratoire LMC-IMAG, Université Joseph Fourier, Tour IRMA, B.P. 53, 38042 Grenoble CEDEX 9, France (Jianwei.Ma@imag.fr).
‡Department of Mathematics and Computer Science, University of Mannheim, D-68131 Mannheim, Germany (Markus.Fenn@uni-mannheim.de).

with line singularities effectively by mapping them into point singularities using the Radon transform. It should be noted that several other geometric multiresolution structures such as curvelets by Candès and Donoho [6], bandelets by LePennec and Mallat [22], or contourlets by Do and Vetterli [9] have been proposed to restore local image features in a different way. Moreover, these methods have to compete with anisotropic diffusion filtering [37].

When implementing a discrete ridgelet transform one has to cope with certain technical difficulties. The basic strategy of the ridgelet transform is an application of the wavelet transform on the projections of the Radon transform. The Radon transform seems natural and simple on the continuum, but it is a challenging problem for discrete data.

Do and Vetterli [10] proposed an orthonormal version of the ridgelet transform based on a discrete Radon transform defined on the finite grid $\mathbb{Z}_p^2$, where p is a prime number. Unfortunately, the $\mathbb{Z}_p^2$ Radon transform integrates over "lines" which are defined algebraically—due to the arithmetic modulo p—rather than geometrically. This causes a wrap-around effect, i.e., texture-like artifacts in reconstructions.

Carré and Andres [7] presented a so-called discrete analytical ridgelet transform (DART) with a flexible redundancy factor based on discrete analytical lines, which cause only a limited wrap-around effect. By using an arithmetical thickness they choose the best discrete approximation of the Euclidean line for each line direction. The innovative step of this transform is the construction of discrete analytical lines in the Fourier domain, which allows a fast perfect backprojection without interpolation or iteration.

Other discrete ridgelet transforms have been also explored by Donoho et al. In [33], they offer a discrete transform with exact reconstruction, stability against perturbations, and low computational complexity. It uses a pseudopolar sampling grid based on concentric squares by means of a simple nearest-neighbor interpolation scheme. In [11], an effective discrete ridgelet transform based on so-called true ridge functions was proposed using the fast slant stack (FSS) in [1], a pseudopolar FFT-based discrete Radon transform, followed by fast one-dimensional (1-D) wavelet transforms. The essential part of the FSS-based "true" ridgelet transform is an interpolation performed using a Dirichlet kernel, which leads to a transform that is geometrically faithful and has no wrap-around effect. But an iterative approximation process is required for the inverse transform.

In this paper, we develop a discrete complex ridgelet transform based on the nonequispaced fast Fourier transform (NFFT). This ridgelet transform uses the NFFT for the computation of the discrete Radon transform. As for the FSS, this approach completely avoids linear interpolations and requires only $\mathcal{O}(n^2 \log n)$ arithmetic operations. Then 1-D DT CWTs are applied to the projections of the Radon transform. The DT CWT was introduced by Kingsbury [19, 20] to cope with the lack of translation and rotation invariance of the (decimated) wavelet transform in an efficient way. For a mathematical treatment of the DT CWT, see also [32, 28]. Replacing the DWT by the approximate shift invariant DT CWT improves the quality of ridgelet denoising remarkably.

For denoising or feature extraction, a shrinkage function is applied to the ridgelet coefficients. However, this usual shrinkage is far from optimal and leads to blurred edges and pseudo-Gibbs artifacts. Therefore, we combine thresholding with TV minimization in order to reduce these artifacts.

TV minimization was first introduced by Rudin, Osher, and Fatemi [31] for denoising, and then was widely studied in image processing and computer vision

(e.g., restoration, inpainting, blind deconvolution). Recently, the TV model has been combined with computational harmonic analysis (e.g., wavelets, wavelet packets, curvelets, etc.) to reduce both the pseudo-Gibbs and staircasing artifacts [26, 8, 12, 23, 35]. The idea of coupled TV minimization used in this paper is similar to [12, 23] but applied for complex ridgelet coefficients and in two dimensions. The ridgelet transform combined with TV minimization attempts to give a better restoration of ridgelet coefficients. In particular, the oscillations (pseudo-Gibbs artifacts) near discontinuities are eliminated while the strong edges and shapes of the features of surfaces are preserved.

This paper is organized as follows. First, we explain the continuous setting of ridgelets in section 2. In section 3, we introduce our NFFT-based discrete complex ridgelet transform and its inverse. Then we explain hard thresholding and combine it with TV minimization in section 4. Numerical experiments show the good performance of our new method for feature extraction and denoising in section 5.

2. Continuous ridgelet transform. Let $\psi \in L^2(\mathbb{R})$ be a wavelet; i.e., it fulfills the admissibility condition

$$\int_{\mathbb{R}} \frac{|\hat{\psi}(\omega)|^2}{|\omega|}\,\mathrm{d}\omega < \infty.$$

A ridgelet $\psi_{\theta,a,b}$ with orientation parameter $\theta \in [0, 2\pi)$, scale parameter $a > 0$, and location parameter $b \in \mathbb{R}$ is defined by

$$\psi_{\theta,a,b}(x) := a^{-1/2}\psi\big((x_1\cos\theta + x_2\sin\theta - b)/a\big).$$

This function is constant along lines $x_1\cos\theta + x_2\sin\theta = \text{const}$, and transverse it is a wavelet. The continuous ridgelet transform for $f \in L^2(\mathbb{R}^2)$ is defined as

$$\mathfrak{R}_f(\theta,a,b) := \int_{\mathbb{R}^2} f(x)\psi_{\theta,a,b}(x)\,\mathrm{d}x. \tag{2.1}$$

For $f \in \mathcal{S}(\mathbb{R}^2)$, where $\mathcal{S}(\mathbb{R}^2)$ denotes the Schwartz space, the reconstruction formula is given as

$$f(x) = \int_0^{2\pi}\int_{-\infty}^{\infty}\int_0^{\infty} \mathfrak{R}_f(\theta,a,b)\psi_{\theta,a,b}(x)\,\frac{\mathrm{d}a}{a^3}\,\mathrm{d}b\,\frac{\mathrm{d}\theta}{4\pi}.$$

We introduce the vectors $\Theta := (\cos\theta, \sin\theta)^t$ and $\Theta^\perp := (\sin\theta, -\cos\theta)^t$. Now we can rewrite (2.1) by a change of the integration variable $x = s\Theta + t\Theta^\perp$ as

$$\begin{aligned}\mathfrak{R}_f(\theta,a,b) &= \int_{\mathbb{R}}\int_{\mathbb{R}} f(s\Theta + t\Theta^\perp)a^{-1/2}\psi\big((s-b)/a\big)\,\mathrm{d}s\,\mathrm{d}t\\ &= a^{-1/2}\int_{\mathbb{R}} R_f(\theta,s)\psi\big((s-b)/a\big)\,\mathrm{d}s,\end{aligned} \tag{2.2}$$

where the Radon transform $R_f(\theta, s)$ is defined as

$$\begin{aligned}R_f(\theta,s) &:= \int_{\mathbb{R}} f(s\Theta + t\Theta^\perp)\,\mathrm{d}t\\ &= \int_{\mathbb{R}^2} f(x)\delta(s - x_1\cos\theta - x_2\sin\theta)\,\mathrm{d}x\end{aligned} \tag{2.3}$$

with the Dirac delta distribution δ. Thus, ridgelet analysis can be seen as a form of wavelet analysis in the Radon domain.

For $f \in \mathcal{S}(\mathbb{R}^2)$, Radon transform and Fourier transform are related by the so-called projection theorem or Fourier slice theorem

$$\int_{\mathbb{R}} R_f(\theta, s) \mathrm{e}^{-2\pi \mathrm{i} s\sigma} \, \mathrm{d}s = \int_{\mathbb{R}^2} f(x) \, \mathrm{e}^{-2\pi \mathrm{i}\sigma(x_1 \cos\theta + x_2 \sin\theta)} \, \mathrm{d}x; \tag{2.4}$$

see, e.g., [27].

3. NFFT-based discrete complex ridgelet transform. Following (2.2), the basic strategy for the computation of the discrete ridgelet transform is to calculate the Radon transform R_f and then to compute the 1-D wavelet transform of $R_f(\theta, \cdot)$ for every direction θ.

3.1. Discrete radon transform. In this subsection, we are interested in an efficient discrete Radon transform based on the NFFT, e.g., applied in [29]. First, we sketch the NFFT, and then we explain how to apply it to compute the Radon transform.

Let $n \in \mathbb{N}$ be even, $I_n := \{-\frac{n}{2}, \ldots, \frac{n}{2} - 1\}$, and $\mathbb{T}^2 := [-\frac{1}{2}, \frac{1}{2})^2$. The bivariate NFFT computes approximations $\tilde{F}$ of the trigonometric polynomial

$$F(x) = \sum_{(\mu,\nu) \in I_n^2} f_{\mu,\nu} \, \mathrm{e}^{-2\pi \mathrm{i}(\mu x_1 + \nu x_2)} \tag{3.1}$$

at arbitrary knots $x^{(\ell)} := \big(x_1^{(\ell)}, x_2^{(\ell)}\big)^t \in \mathbb{T}^2$, $\ell = 1, \ldots, M$. Straightforward computation of (3.1) requires $\mathcal{O}(n^2 M)$ arithmetical operations, too much for practical purposes. For equispaced knots $x^{(\mu,\nu)} = (\mu/n, \nu/n)^t$, $(\mu, \nu) \in I_n^2$, the arithmetic complexity can be reduced to $\mathcal{O}(n^2 \log n)$ operations by applying the well-known FFT.

The main idea of the NFFT consists in the approximation of F by linear combinations of translates of a window function. Original approaches used Gaussian bells [13], B-splines [2], or Kaiser–Bessel functions [16, 17] as window functions. A general approach to NFFTs can be found in [34, 30, 18]. Free NFFT software packages are available [21, 15].

The fast computation of (3.1) by the NFFT requires

$$\mathcal{O}(\alpha^2 n^2 \log n + m^2 M)$$

arithmetic operations. Both parameters, the oversampling factor α and the cut-off parameter m, have to be chosen in accordance with the desired accuracy of the fast algorithm. In general the approximation error introduced by the NFFT decays exponentially in m, where the basis of the exponent depends on α. In our numerical experiments, we will use the Gaussian window function. Then the error can be estimated by [14] as

$$\frac{|F(x^{(\ell)}) - \tilde{F}(x^{(\ell)})|}{\sum\limits_{(\mu,\nu) \in I_n^2} |f_{\mu,\nu}|} \leq 16 \mathrm{e}^{-m\pi(1 - 1/(2\alpha - 1))}. \tag{3.2}$$

Next, we want to couple the NFFT with the computation of the Radon transform. Let $f_{\mu,\nu}$, $(\mu, \nu) \in I_n^2$, be given discrete data. For a line $x_1 \cos\theta + x_2 \sin\theta = s$ we discretize the Radon transform (2.3) as

$$R_f(\theta, s) := \sum_{(\mu,\nu) \in I_n^2} f_{\mu,\nu} K(s - \mu\cos\theta - \nu \sin\theta), \tag{3.3}$$

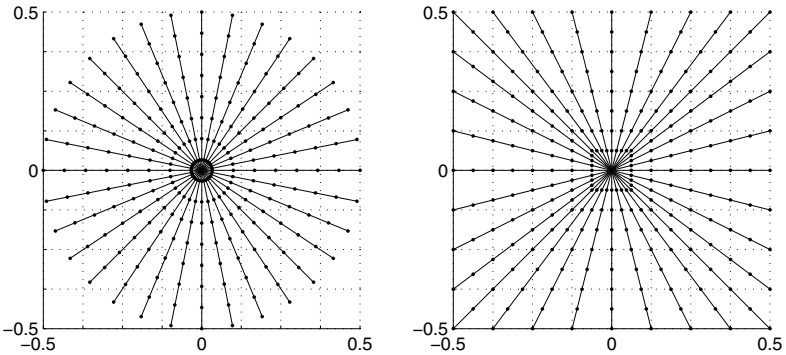


FIG. 3.1. *Polar grid (left) and pseudopolar grid (right).*

where the kernel function

$$K(x) := \sum_{r=-n}^{n-1} \alpha_r \, \mathrm{e}^{2\pi \mathrm{i} r x/(2n)} \tag{3.4}$$

is used as an approximation of the Dirac delta distribution.

We now discretize the parameter domain as $\theta_t = \frac{\pi}{2n} t$ $(t = 0, \ldots, 2n-1)$ and $s = -n, \ldots, n-1$. Together with (3.4), equation (3.3) yields

$$R_f(\theta_t, s) = \sum_{r=-n}^{n-1} \alpha_r \sum_{(\mu,\nu)\in I_n^2} f_{\mu,\nu} \mathrm{e}^{-2\pi \mathrm{i}(\mu \frac{r\cos\theta_t}{2n} + \nu \frac{r\sin\theta_t}{2n})} \mathrm{e}^{2\pi \mathrm{i} r s/(2n)}. \tag{3.5}$$

Note that (3.5) can be seen as a discrete analogue of the Fourier slice theorem (2.4).

Obviously, a fast algorithm for the computation of the discrete Radon transform is given by computing the inner sum in (3.5) by a two-dimensional (2-D) NFFT at the $M = 4n^2$ knots $\left(\frac{r\cos\theta_t}{2n}, \frac{r\sin\theta_t}{2n}\right)_{t,r}$ of the polar grid (see Figure 3.1(left)) followed by multiplication with the kernel coefficients α_r and ends with the computation of the outer sum by 1-D inverse FFTs (iFFTs) for every direction θ_t.

By choosing $\alpha_n = 0$ and $\alpha_{-r} = \alpha_r$ $(r = -n+1, \ldots, n-1)$ in (3.5), we can ensure real values for the discrete Radon transform. In our numerical examples, we will use the (modified) [illegible] $K(x) = \sum_{r=-n+1}^{n-1} \left(1 - \frac{|r|}{n}\right) \mathrm{e}^{2\pi \mathrm{i} r x/(2n)}$.

[illegible] Using in our definition of the discrete Radon transform (3.5) a (modified) Dirichlet kernel $K(x) = \sum_{r=-n}^{n-1} \mathrm{e}^{2\pi \mathrm{i} r x/(2n)}$ together with the pseudopolar grid (see Figure 3.1(right)) leads essentially to the same notion of discrete Radon transform as proposed by Averbuch et al. [1] and Donoho and Flesia [11]. They call their transform [illegible] (FSS). For the computation of the inner sum in (3.5), these authors use the so-called pseudopolar FFT based on the chirp-z transform as an alternative for the NFFT.

3.2. Dual-tree complex wavelet transform. To obtain the ridgelet transform, we have to apply fast 1-D orthogonal wavelet transforms in the Radon domain. Let the projection $R_f(\theta, \cdot)$ in direction θ be represented by a wavelet decomposition of the form

$$R_f(\theta, \cdot) = \sum_{k\in\mathbb{Z}} c_{\theta,j_0,k} \phi_{j_0,k} + \sum_{j\le j_0} \sum_{k\in\mathbb{Z}} d_{\theta,j,k} \psi_{j,k}, \tag{3.6}$$

where $\phi_{j_0,k}$ denotes the scaling function at the coarsest level j_0 associated with the orthogonal wavelet basis $\{\psi_{j,k} := \psi\big((x-b)/a\big) : a = 2^{-j},\ b = k2^{-j},\ k,j \in \mathbb{Z}\}$. Then the coefficients $c_{\theta,j_0,k} = \langle R_f(\theta,\cdot), \phi_{j_0,k}\rangle$ and $d_{\theta,j,k} = \langle R_f(\theta,\cdot), \psi_{j,k}\rangle$ refer to the smooth components at level j_0 and the detail components at level j in direction θ, respectively.

The discrete wavelet transform (DWT) suffers from shift sensitivity, which is caused by aliasing due to the transform having maximal decimation at each level. Many wavelet techniques, including undecimated wavelet transform and complex wavelet transform, have been explored to solve this problem, but they either require highly redundant computational cost or they do not have perfect reconstruction or good filter characteristics. Based on the z-transform theory of linear time invariant systems, Kingsbury [19, 20] proposed the dual-tree complex wavelet transform (DT CWT), which adds perfect reconstruction to approximate shift invariance of complex wavelets. In contrast to the $\mathcal{O}(n \log n)$ undecimated DWT, which is $\log n$ times redundant in one dimension and $3 \log n$ times redundant in two dimensions, the DT CWT is only slightly redundant by a factor of 2 in one dimension and 4 in two dimensions. Thus, the computational complexity of the DT CWT remains $\mathcal{O}(n)$, the same as for the decimated DWT.

Furthermore, the complex ridgelet transform can provide better phase information, a useful property for many applications. In [25], one of us showed that the ridgelet transform can achieve approximate shift invariance by the use of the DT CWT. Unfortunately, textural artifacts are still unavoidable since the finite $\mathbb{Z}_p^2$ Radon transform is used as a building block in [25]. However, the DT CWT used in this paper is well motivated by this prior attempt.

The DT CWT is a special discrete complex wavelet transform, which is essentially a combination of two different real wavelet transforms by a sophisticated dual tree. More precisely, we have two wavelet decompositions of the form (3.6) with scaling functions $\phi^r_{j_0,k}$, $\phi^i_{j_0,k}$ and wavelets $\psi^r_{j,k}$, $\psi^i_{j,k}$, respectively. We set $\phi_{j_0,k} := \frac{1}{\sqrt{2}}(\phi^r_{j_0,k} - \mathrm{i}\phi^i_{j_0,k})$ and $\psi_{j,k} := \frac{1}{\sqrt{2}}(\psi^r_{j,k} - \mathrm{i}\psi^i_{j,k})$. Then the DT CWT can be seen as the real part of a decomposition of the form (3.6) with the outputs of the dual tree interpreted as the real and imaginary parts of the complex coefficients

$$c_{\theta,j_0,k}(f) := \tfrac{1}{\sqrt{2}}\langle R_f(\theta,\cdot), \phi^r_{j_0,k}\rangle + \mathrm{i}\tfrac{1}{\sqrt{2}}\langle R_f(\theta,\cdot), \phi^i_{j_0,k}\rangle,$$
$$d_{\theta,j,k}(f) := \tfrac{1}{\sqrt{2}}\langle R_f(\theta,\cdot), \psi^r_{j,k}\rangle + \mathrm{i}\tfrac{1}{\sqrt{2}}\langle R_f(\theta,\cdot), \psi^i_{j,k}\rangle.$$

Therefore, we will call the proposed ridgelets [illegible]. Note that the filter/wavelet design for the DT CWT is rather sophisticated. In our experiments, we will use the nearly orthogonal (13,19)-tap filters at level 1 and the 18-tap Q-shift filters constructed in [20] for the higher levels.

In summary, we obtain the following algorithm for the fast computation of the discrete complex ridgelet transform.

ALGORITHM 3.1 (discrete complex ridgelet transform).

[illegible] $f := (f_{\mu,\nu})_{(\mu,\nu)\in I_n^2}$

1. [illegible] $\theta_t = \frac{\pi}{2n}t$ $(t = 0,\dots,2n-1)$ [illegible] $r = -n,\dots,n-1$ [illegible]

$$\hat{f}_{\theta_t,r} := \sum_{(\mu,\nu)\in I_n^2} f_{\mu,\nu}\, \mathrm{e}^{-2\pi\mathrm{i}\left(\mu\frac{r\cos\theta_t}{2n} + \nu\frac{r\sin\theta_t}{2n}\right)}$$

[illegible] 2 [illegible]

2. [illegible] $s = -n, \dots, n-1$ [illegible]

$$R_f(\theta_t, s) = \sum_{r=-n}^{n-1} \hat{f}_{\theta_t,r}\, \mathrm{e}^{2\pi \mathrm{i} rs/(2n)}$$

[illegible] 1 [illegible] $\hat{f}_{\theta_t,r}$ [illegible] θ_t

3. [illegible]

$$c_{\theta_t,j_0,k}(f) \quad [\text{illegible}] \quad d_{\theta_t,j,k}(f)$$

[illegible] 1 [illegible] $R_f(\theta_t, \cdot)$ [illegible] θ_t
[illegible] $c_{\theta_t,j_0,k}(f)$ $d_{\theta_t,j,k}(f)$

3.3. The inverse complex ridgelet transform. The inverse discrete complex ridgelet transform can be achieved by inverting the steps of Algorithm 3.1 one after the other in reverse order.

Inverting the 2-D NFFT (3.1) in the first step needs more attention. We consider the following reconstruction or recovery problem. Given the values $F^{(\ell)} \in \mathbb{C}$ of a trigonometric polynomial at nonequispaced knots $x^{(\ell)} \in \mathbb{T}^2$ $(\ell = 1, \dots, M)$, the aim is to reconstruct its Fourier coefficients $f_{\mu,\nu}$, $(\mu, \nu) \in I_n^2$, i.e., to solve the linear system of equations

$$\sum_{(\mu,\nu)\in I_n^2} f_{\mu,\nu} \mathrm{e}^{-2\pi \mathrm{i}(\mu x_1^{(\ell)} + \nu x_2^{(\ell)})} = F^{(\ell)} \qquad (\ell = 1, \dots, M). \tag{3.7}$$

In our case, the number of knots is larger than the dimension of the space of trigonometric polynomials, i.e., $M > n^2$, so that the linear system (3.7) is overdetermined and can thus only be solved approximately. A standard method is to use the least squares approximation. For its numerical solution, the free NFFT software package [21] provides fast iterative CG-type algorithms (iNFFTs) which involve the NFFT for the fast matrix-vector multiplications occurring in the CG steps. The approximation error remains the same as for the NFFT estimated by inequality (3.2).

Since the second step of Algorithm 3.1 can be easily inverted by 1-D FFTs and the first step can be inverted (approximately to arbitrary accuracy) by a 2-D iNFFT, we also have a fast algorithm for the inverse discrete Radon transform. The algorithm possesses low arithmetic costs and leads to a good reconstruction quality similar to that of filtered backprojection.

ALGORITHM 3.2 (inverse discrete complex ridgelet transform).

[illegible] $c_{\theta_t,j_0,k}(f)$ $d_{\theta_t,j,k}(f)$

1. [illegible] $s = -n, \dots, n-1$ [illegible] $\theta_t = \frac{\pi}{2n} t$ $(t = 0, \dots, 2n-1)$ [illegible]

$$R_f(\theta_t, s)$$

2. [illegible] $r = -n, \dots, n-1$ [illegible]

$$\hat{f}_{\theta_t,r} = \sum_{s=-n}^{n-1} R_f(\theta_t, s)\, \mathrm{e}^{-2\pi \mathrm{i} rs/(2n)}$$

[illegible] 1 [illegible] θ_t

3. [illegible] $(\mu,\nu) \in I_n^2$ [illegible] $f_{\mu,\nu}$ [illegible]

$$\hat{f}_{\theta_t,r} = \sum_{(\mu,\nu)\in I_n^2} f_{\mu,\nu}\, \mathrm{e}^{-2\pi \mathrm{i}\left(\mu \frac{r\cos\theta_t}{2n} + \nu \frac{r\sin\theta_t}{2n}\right)}$$

[illegible] 2 [illegible] $f := (f_{\mu,\nu})_{(\mu,\nu)\in I_n^2}$

4. Combination of hard thresholding with TV minimization. The combination of ordinary wavelet shrinkage with TV minimization was successfully applied in various papers [8, 12, 23, 26]. In this paper we apply the technique with respect to our ridgelet coefficients. We mainly follow the approach in [12].

Many problems such as denoising and feature extraction can be written as the model

$$f = f_{\text{true}} + \varepsilon,$$

where f is the observed function, f_{true} is the function (or component) to estimate, and ε is white Gaussian noise or, more generally, an unwanted component. Our aim is to restore f_{true} without artifacts, i.e., recover the smooth parts of f_{true} while preserving the discontinuities.

The [illegible] τ is defined for $x \in \mathbb{C}$ as

$$\tau(x) := \begin{cases} x, & |x| \geq \sigma, \\ 0, & |x| < \sigma. \end{cases}$$

We restrict our attention to hard thresholding, although other thresholding functions such as soft thresholding or garotte thresholding can also be used. Applying hard thresholding with some fixed threshold σ to the complex ridgelets coefficients $d_{\theta_t,j,k}(f)$ of our discrete function f and applying the inverse discrete complex ridgelet transform of Algorithm 3.2 results in a function $f^{(0)}$. The indices of the ridgelet coefficients retained after hard thresholding are recorded in Λ, i.e.,

$$\Lambda := \big\{(\theta_t, j, k) : |d_{\theta_t,j,k}(f)| \geq \sigma\big\}.$$

Unfortunately, wavelet thresholding produces artificial oscillations near discontinuities, a phenomenon known as "pseudo-Gibbs phenomenon" or "side-band effect." To lower the pseudo-Gibbs artifacts we will use TV minimization. For a function $f : \mathbb{R}^2 \supseteq \Omega \to \mathbb{R}$ with $|\nabla f(x)| \in L^1(\mathbb{R}^2)$ the TV functional is defined by

$$TV(f) = \int_\Omega \big|\nabla f(x)\big|\, \mathrm{d}x. \tag{4.1}$$

TV functionals as regularizing terms became very popular in image processing. Meanwhile, there exists a broad literature on this topic. In particular, relations between TV regularization and wavelet shrinkage were examined in [35, 38].

To circumvent computational difficulties arising from the nondifferentiability of the modulus at zero, the TV functional is often replaced by

$$TV(f) = \int_\Omega \sqrt{\big|\nabla f(x)\big|^2 + \beta^2}\, \mathrm{d}x \tag{4.2}$$

with a small parameter $\beta > 0$; see [36]. In the following, we restrict our attention to (4.2) and use the following discrete version for $f := \big(f_{\mu,\nu}\big)_{(\mu,\nu)\in I_n^2}$:

$$TV(f) = \sum_{\mu,\nu} \sqrt{\big|(\delta_1 f)_{\mu,\nu}\big|^2 + \big|(\delta_2 f)_{\mu,\nu}\big|^2 + \beta^2}, \tag{4.3}$$

where $(\delta_1 f)_{\mu,\nu} = f_{\mu+1,\nu} - f_{\mu,\nu}$, $(\delta_2 f)_{\mu,\nu} = f_{\mu,\nu+1} - f_{\mu,\nu}$. More precisely, for our given f let

$$U := \Big\{ u := \big(u_{\mu,\nu}\big)_{(\mu,\nu)\in I_n^2} : c_{\theta_t,j_0,k}(u) = c_{\theta_t,j_0,k}(f) \quad \forall(\theta_t,k), \\ d_{\theta_t,j,k}(u) = d_{\theta_t,j,k}(f) \quad \forall(\theta_t,j,k)\in\Lambda \Big\}.$$

Then we are looking for the solution of the constrained minimization problem

$$\min_{u\in U} TV(u). \tag{4.4}$$

Let the linear subspace V of functions on I_n^2 be given by

$$V := \Big\{ v := \big(v_{\mu,\nu}\big)_{(\mu,\nu)\in I_n^2} : c_{\theta_t,j_0,k}(v) = 0 \quad \forall(\theta_t,k), \\ d_{\theta_t,j,k}(v) = 0 \quad \forall(\theta_t,j,k)\in\Lambda \Big\}.$$

Then it holds that $U = f^{(0)} + V$; i.e., U is an affine and hence convex space. Since the TV functional is convex, too, problem (4.4) has a solution. Given a positive sequence $(t_\ell)_{\ell=0}^\infty$ with $\lim_{\ell\to\infty} t_\ell = 0$ and $\sum_{\ell=0}^\infty t_\ell = \infty$, it was shown in [12] that a solution of (4.4) can be computed by the following [illegible] (using subgradients for the nonmodified TV functional (4.1)):

$$f^{(\ell+1)} = f^{(\ell)} - t_\ell P_V\big(\nabla_f TV(f^{(\ell)})\big)$$

with our hard-thresholded function $f^{(0)}$ as initial guess. Here $P_V(g)$ denotes the projection of g onto V, i.e., the inverse discrete complex ridgelet transform of

$$c_{\theta_t,j_0,k}(v) := 0, \qquad d_{\theta_t,j,k}(v) := \begin{cases} 0 & \text{for } (\theta_t,j,k)\in\Lambda, \\ d_{\theta_t,j,k}(g) & \text{else.} \end{cases}$$

It is applied to the gradient ∇_f of the discrete TV functional (4.3) given by

$$\begin{aligned} \nabla_f TV(f) := {} & (2f_{\mu,\nu} - f_{\mu,\nu+1} - f_{\mu+1,\nu})\big[(f_{\mu,\nu+1} - f_{\mu,\nu})^2 + (f_{\mu+1,\nu} - f_{\mu,\nu})^2 + \beta^2\big]^{-1/2} \\ & + (f_{\mu,\nu} - f_{\mu-1,\nu})\big[(f_{\mu-1,\nu+1} - f_{\mu-1,\nu})^2 + (f_{\mu-1,\nu} - f_{\mu,\nu})^2 + \beta^2\big]^{-1/2} \\ & + (f_{\mu,\nu} - f_{\mu,\nu-1})\big[(f_{\mu+1,\nu-1} - f_{\mu,\nu-1})^2 + (f_{\mu,\nu} - f_{\mu,\nu-1})^2 + \beta^2\big]^{-1/2} \end{aligned} \tag{4.5}$$

for the inner points $(\mu,\nu)\in I_n^2$ and corresponding modifications at the boundary ∂I_n^2.

Algorithm 4.1 (TV minimization of complex ridgelet coefficients).

[illegible] f [illegible] σ [illegible] t_ℓ

1. [illegible] $c_{\theta_t,j_0,k}(f)$ $d_{\theta_t,j,k}(v)$ [illegible] 3.1

2. [illegible] Λ
3. [illegible] $f^{(0)}$ [illegible] 3.2 [illegible]
4. [illegible] $f^{(\ell)}$ [illegible] $\ell = 0, 1, \ldots$
   (a) [illegible] $g^{(\ell)}$ [illegible] $f^{(\ell)}$ [illegible] (4.5)
   (b) [illegible] $c_{\theta_t,j_0,k}(g^{(\ell)})$ $d_{\theta_t,j,k}(g^{(\ell)})$ [illegible] 3.1
   (c) [illegible] $P_V(g^{(\ell)})$ [illegible] 3.2 [illegible]

$$c_{\theta_t,j_0,k}(v) := 0, \qquad d_{\theta_t,j,k}(v) := \begin{cases} 0 & \text{[illegible]} \ (\theta_t, j, k) \in \Lambda, \\ d_{\theta_t,j,k}(g^{(\ell)}) & \text{[illegible]} \ . \end{cases}$$

   (d) [illegible]

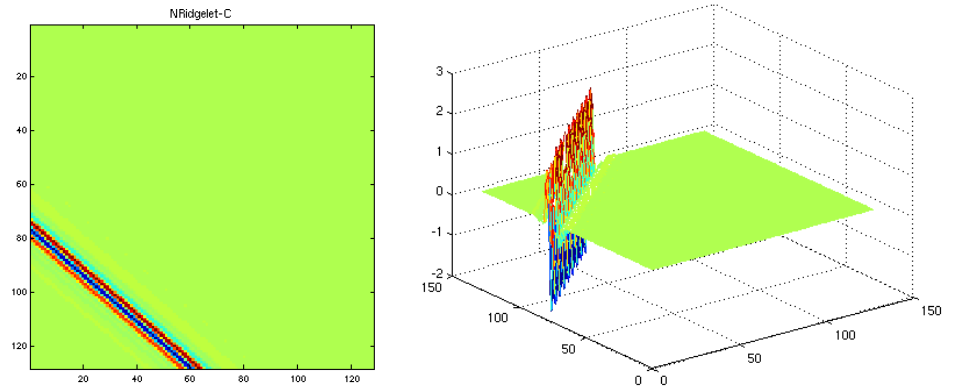
$$f^{(\ell+1)} := f^{(\ell)} + t_\ell P_V(g^{(\ell)}).$$

[illegible] $f^{(\ell)}$

5. Experiments. Some abbreviations used in this section should be declared. Nridgelet-c denotes the discrete complex ridgelet transform of Algorithm 3.1 and Nridgelet-c-tv is the proposed Algorithm 4.1. We always use the Gaussian window function with cut-off parameter $m = 4$ and oversampling factor $\alpha = 2$ for the NFFT [21]. The DT CWT is due to Kingsbury [20] and applied with the near-symmetry (13,19)-tap biorthogonal filters at level 1 and 18-tap Q-shift filters in a cascaded way at levels beyond level 1. Other abbreviations are Nridgelet for the NFFT-based real ridgelet transform, in which we use DWT (Sym4 wavelets) in the second step instead of the DT CWT and FRIT for Do and Vetterli's finite $\mathbb{Z}_p^2$ ridgelet transform.

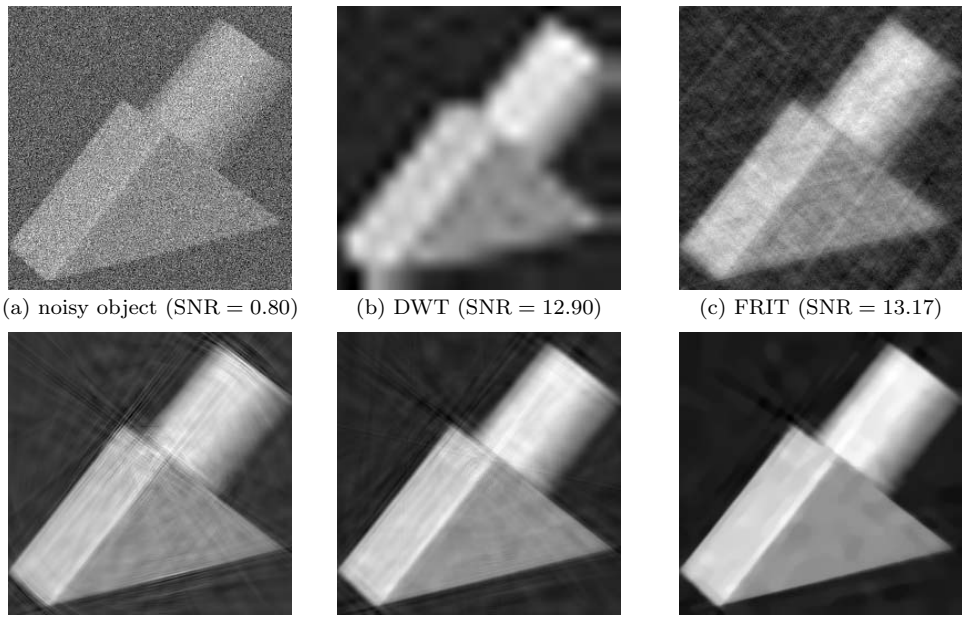
In the first test, we show the inverse Nridgelet-c of a point (all ridgelet coefficients are zero but one). In Figures 5.1(a) and (b) the obtained 2-D (resp., three-dimensional (3-D)) surface is depicted. Unlike FRIT, it can be seen that the Nridgelet-c is free from wrap-around artifacts. The similar sample was given using the recent discrete analytical ridgelet transform (DART) in [7]. Our method is also better than the DART in terms of the influence of the wrap-around artifacts in comparison to the displayed illustrations. However, we note that it is nonsmooth along the ridge shown in Figure 5.1(b). The proposed Nridgelet-c in this paper is not based on true ridge function, because the polar grid is adopted.

Our next test shows the good performance of the proposed method for denoising an image with line singularities. An image (256×256) of an object used in [10] is considered in order to compare with other methods conveniently. Figure 5.2(a) shows the object contaminated with additive zero-mean Gaussian white noise of variance σ^2. A universal hard thresholding is applied to denoising. Figure 5.2(b) shows the result obtained using the (9,7)-tap DWT. As we know, it causes nonsmoothness along the edges. Figure 5.2(c) shows the result obtained using FRIT. It suffers from strongly textural artifacts. Figure 5.2(d) shows the result obtained using the Nridgelet. Figure 5.2(e) shows the result obtained using the Nridgelet-c. It can be seen from Figures 5.2(d) and (e) that the proposed framework is effective in recovering straight edges, even with such heavy background noise. The Nridgelet-c shows a better result with fewer line-like artifacts, as well as in terms of the signal to noise ratio (SNR). Generally speaking, a more oscillating wavelet basis used in the second step produces fewer



(a) (b)

FIG. 5.1. *The inverse discrete complex ridgelet transform of only one nonzero coefficient shown by the* 2*-D* (a) *and* 3*-D surface* (b)*, respectively.*



(a) noisy object (SNR = 0.80) (b) DWT (SNR = 12.90) (c) FRIT (SNR = 13.17)

(d) Nridgelet (SNR = 16.65) (e) Nridgelet-c (SNR = 19.16) (f) Nridgelet-c-tv (SNR = 20.40)

FIG. 5.2. *Denoising using different methods.*

artifacts, as mentioned in [10], but causes more computational burden because of its larger support. A similar denoising sample using the FSS-based ridgelet transform and DART with a very oscillating wavelet basis (Daubechies20) is depicted in [7, Fig. 3]. There seem to be fewer disturbing artifacts than in the Nridgelet result, but the edges are not very clear. Figure 5.2(f) in this paper shows the result obtained using the Nridgelet-c-tv with $t_\ell = 0.1$ ($\ell = 1, \ldots, 120$). It is much more effective in reducing undesirable artifacts, including the pseudo-Gibbs artifacts, while preserving the edges.

Figure 5.3 shows a comparison of the center profile taken across the original object and Figures 5.2(e) and (f), respectively. The effectivity of Nridgelet-c-tv can be seen

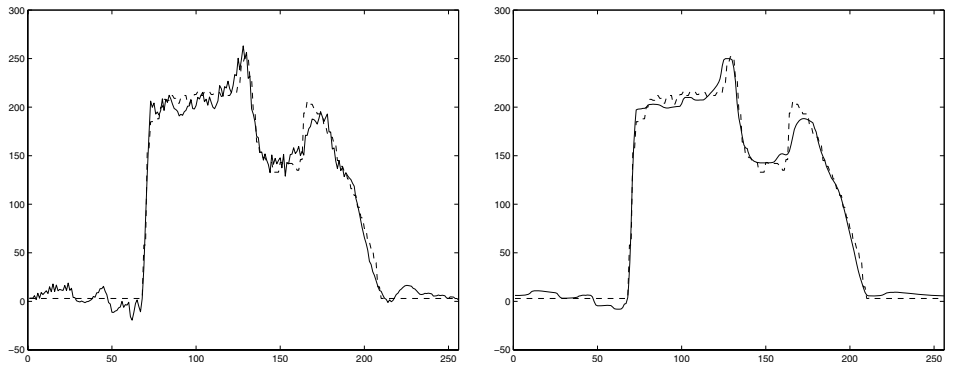


FIG. 5.3. *The center profile taken across the original object (shown as the dashed line) and the reconstruction by* (a) *the Nridgelet-c (shown in Figure* 5.2(e)*) and* (b) *the Nridgelet-c-tv (shown in Figure* 5.2(f)*), respectively.*

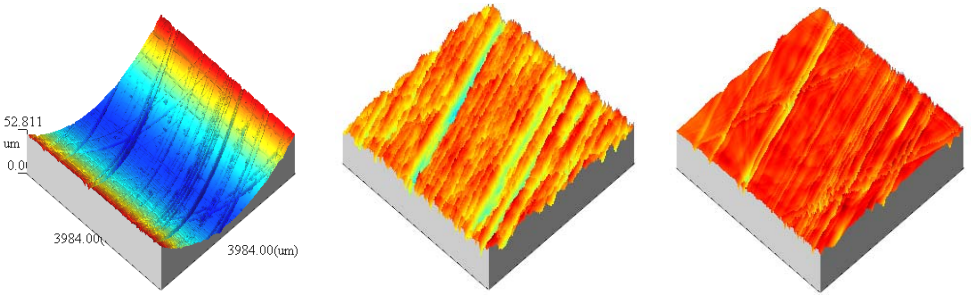
more clearly from these profiles.

The simple thresholding scheme is effective for the ridgelet methods in denoising the piecewise smooth image with line singularities, because the line singularities are represented by a few large coefficients while noise unlikely generates significant coefficients. From the view of the hybrid approach, the Nridgelet-c combines just the multiresolution analysis of the DT CWT with the anisotropy of the Radon transform. Nridgelet-c-tv offers an optimal choice of reconstructed coefficients in the procedure of thresholding.

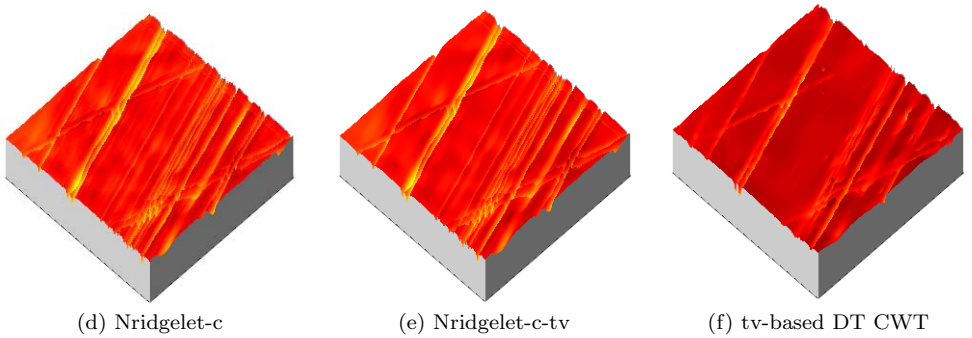
In our third test, we apply the proposed method to extract straight scratches in the bands of roughness and waviness in order to study the functional performance of the 3-D surface topography of systems according to different applications. The extracted information could be fed back to monitor and manufacturing processes or to study actual contact stress, loaded area, asperity volume, and lubrication regimes occurring during the initial stages of wear of surfaces in service.

Figure 5.4 shows a typical engineering surface: a honed surface from an automotive engine cylinder. This kind of surface includes the form, waviness, and roughness components that almost submerge the main features of the deep valleys/scratches. Usually, the most important features that affect the performance of the cylinder are scratches whose distribution and amplitude will considerably influence the flow of gas or air in a pressure balance of an engine.

Figure 5.4(a) is the raw measured surface including the form, waviness, and roughness components. Figure 5.4(b) is the extracted surface using FRIT followed by removing form error. It is far way from satisfaction because of the strongly texture-like artifacts. Figure 5.4(c) is the extracted surface using the Nridgelet. As we supposed, the DWT suffers from shift aliasing, which leads to the line-like artifacts in the extracted surface by Nridgelet. Figure 5.4(d) is the extracted surface using Nridgelet-c. The edge-preserving scratches are nicely restored. But the pseudo-Gibbs artifacts resulting from the aberrant ridgelet coefficients are still visible. Figure 5.4(e) is the extracted surface using the Nridgelet-c-tv with $t_\ell = 0.001$ ($\ell = 1, \ldots, 100$). The nonsmooth artifacts are reduced remarkably. This is due to the fact that Nridgelet-c-tv allows the reconstruction of some small ridgelet coefficients, which are canceled by thresholding. Furthermore, the elimination of pseudo-Gibbs artifacts can be seen more clearly from their center profiles shown in Figure 5.5, in which Figures 5.5(a)

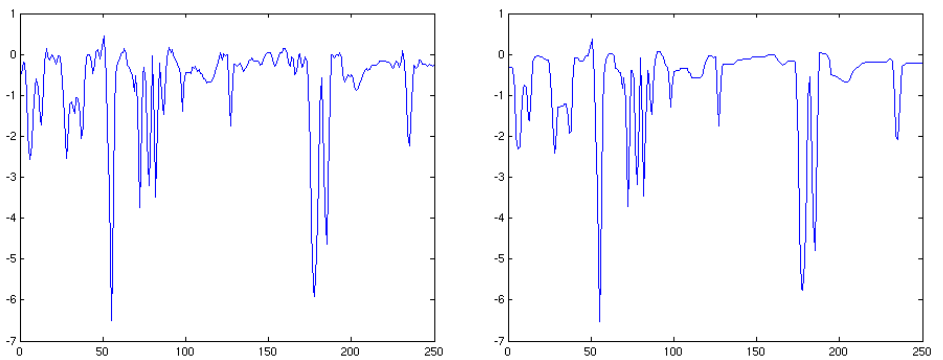


(a) raw measured surface (b) FRIT (c) Nridgelet



(d) Nridgelet-c (e) Nridgelet-c-tv (f) tv-based DT CWT

FIG. 5.4. *Extraction of scratches from a honed surface of an automotive engine cylinder.*



(a) (b)

FIG. 5.5. *The center profiles taken across* (a) *the Nridgelet-c extracted surface (shown in Figure* 5.4(d)*) and* (b) *the Nridgelet-c-tv extracted surface (shown in Figure* 5.4(e)*), respectively.*

and (b) are taken across the extracted surfaces using Nridgelet-c and Nridgelet-c-tv, as shown in Figures 5.4(d) and (e), respectively. The experiments show that the method is more efficient in removing the pseudo-Gibbs oscillations when the DT CWT and TV are joined. As to the method when one uses TV minimization only, i.e., combining TV with Nridgelet instead of Nridgelet-c, one needs many more iterations than with Nridgelet-c-tv in order to eliminate the strong line-like artifacts caused by shift variance of the DWT. In such a case, the intrinsic scratches will be smoothed over to some extent.

Figure 5.4(f) shows the extracted surface using the TV-based DT CWT method with 500 iterations proposed in [23]. The method has a good performance, but in

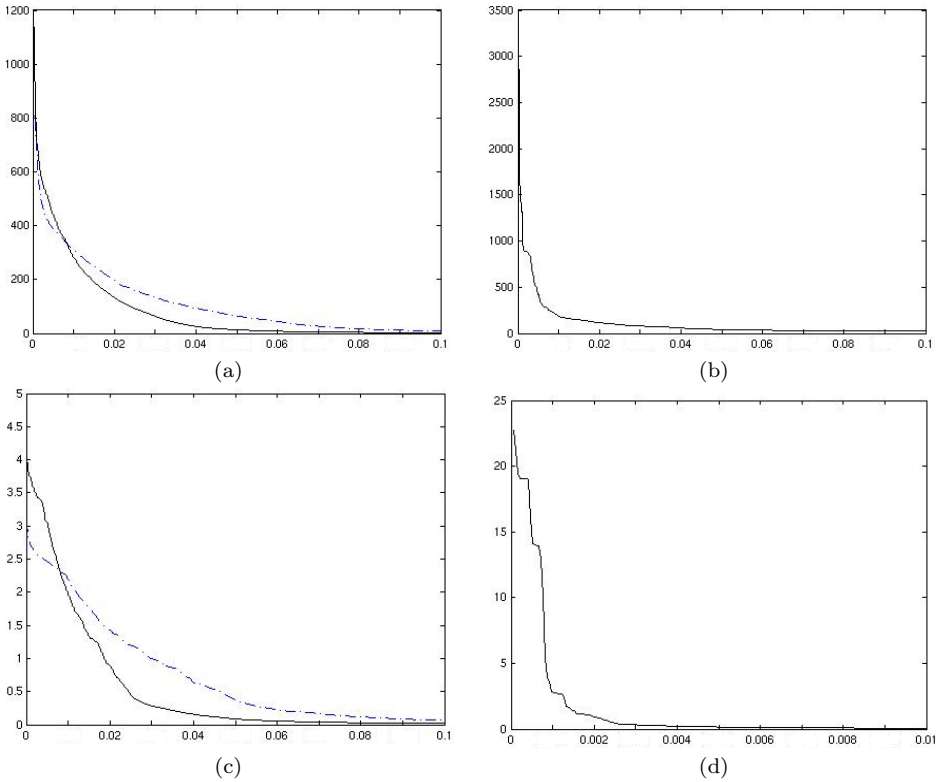


FIG. 5.6. *Decreasing rearrangement of ridgelet coefficients of object (upper) and HalfDome (lower).* (a) *and* (c): *the real line denotes Nridgelet coefficients and the dotted-dashed line denotes Nridgelet-c coefficients;* (b) *and* (d): *FRIT coefficients. The vertical coordinate denotes the absolute value of coefficients; the horizontal coordinate denotes the ratio of rearranged coefficients to all coefficients. Note the difference of the horizontal coordinate in* (d).

terms of the extraction of scratches, setting a high threshold to kick out the point-like features causes the intrinsic features to be destroyed and some shallow scratches to be missed. This is because wavelets lack line sensitivity.

In our last test, we examine the decay of the ridgelet coefficients. The proposed method in this paper is an overcomplete system. Although it shows good performance for denoising and feature extraction, the main disappointment is the relatively slow decay of the coefficients in the ridgelet domain. Figure 5.6 gives two examples to show the decreasing rearrangement of the ridgelet transform coefficients, in comparison to the orthogonal FRIT. The upper line shows the rearrangement of the object used in the second test. The object cannot be represented as a summation of a few global linear singularities, and thus it is not in the optimal class of the ridgelet transform [10]. The lower line shows those of a typically optimal class: HalfDome, i.e., the mutilated Gaussian $g(x_1, x_2) = 1_{\{x_2>0\}} e^{-x_1^2-x_2^2}$. The decay of coefficients of the proposed ridgelet transform is much slower than that of FRIT, especially for the optimal class.

However, leaving out the captious criticism, the proposed ridgelet transform is very effective to sparsely represent line singularities. Figure 5.7 shows the histogram of the coarsest subband $d_{\theta,j_0,k}$ of HalfDome. The distribution is characterized by a very sharp peak (at zero) with an extended tail. The highly non-Gaussian distribution

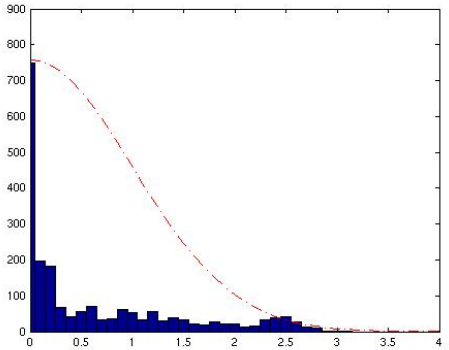


FIG. 5.7. *Histogram of the coarsest subband* $d_{\theta,j_0,k}$ *of HalfDome. The dotted-dashed line denotes the sketch of normal distribution.*

implies that the transform is very spare. Other experiments of real surfaces with line scratches display similar distributions. It is rare to meet natural images with global linear singularities, but for engineering surfaces, many kinds of these images belong to the optimal class of ridgelet transform. So, application of ridgelets to this field is of practical interest.

We remark that visual similar result can be obtained with the pseudopolar grid. Using the Dirichlet kernel instead of the Fejér kernel in (3.5) yields slightly worse results.

6. Conclusions. Based on the requirements of highly accurate functional analysis of engineering surfaces, a new method for line-feature extraction has been proposed by designing a NFFT-based discrete complex ridgelet transform with TV-minimization synthesis. The transform is achieved by taking 1-D DT CWTs (to achieve approximate shift invariance) on the projections of the NFFT-based discrete Radon transform, which identifies the polar grid as a nonequispaced data grid in Cartesian coordinates and thus avoids linear interpolations. This method can be easily modified to incorporate the pseudopolar grid. We remark that the pseudopolar grid together with unequispaced FFT was recently also applied by Candès et al. for the efficient computation of the curvelet transform [4]. TV minimization does not set the nonsignificant ridgelet coefficients simply to zero but typically inputs optimal small values to cancel the pseudo-Gibbs oscillation in the vicinity of discontinuities and eliminates the ripples while perfectly preserving the strong edges and shapes of features. Although the NFFT-based Radon transform, the DT CWT, and TV minimization are not new, they are combined for the first time in this paper. Numerical experiments show that this method performs an almost artifact-free denoising and feature extraction for surfaces with line singularities. It is very significant for ongoing applications on precision engineering involving the steel, pump, and biomedical industries.

Acknowledgments. The authors would like to thank the referees for their constructive remarks. J. Ma especially thanks G. Steidl of the University of Mannheim for her hospitality and many stimulating discussions, I. Moroz of the University of Oxford, L. Blunt, X. Jiang, and P. Scott of the University of Huddersfield for their help on surface characterization, and others.

REFERENCES

[1] A. Averbuch, R. Coifman, D. Donoho, M. Israeli, and J. Waldén, *Fast Slant Stack: A Notion of Radon Transform for Data in a Cartesian Grid Which Is Rapidly Computable, Algebraically Exact, Geometrically Faithful and Invertible*, http://www-stat.stanford.edu/∼donoho/Reports/2001/FastSlantStack.pdf (2001).

[2] G. Beylkin, *On the fast Fourier transform of functions with singularities*, Appl. Comput. Harmon. Anal., 2 (1995), pp. 363–381.

[3] E. Candès, *Harmonic analysis of neural networks*, Appl. Comput. Harmon. Anal., 6 (1999), pp. 197–218.

[4] E. Candès, L. Demanet, D. Donoho, and L. Ying, *Fast discrete curvelet transforms*, Multiscale Model. Simul., to appear.

[5] E. Candès and D. Donoho, *Ridgelets: A key to higher-dimensional intermittency?*, R. Soc. Lond. Philos. Trans. Ser. A Math. Phys. Eng. Sci., 357 (1999), pp. 2495–2509.

[6] E. Candès and D. Donoho, *Curvelets and curvilinear integrals*, J. Approx. Theory, 113 (2001), pp. 59–90.

[7] P. Carré and E. Andres, *Discrete analytical ridgelet transform*, Signal Process., 84 (2004), pp. 2165–2173.

[8] T. Chan and H. Zhou, *Optimal Constructions of Wavelet Coefficients Using Total Variation Regularization in Image Compression*, CAM Tech. Report 00-27, UCLA, Los Angeles, CA, 2000.

[9] M. Do and M. Vetterli, *Contourlets*, in Beyond Wavelets, J. Stoeckler and G. Welland, eds., Academic Press, New York, 2003, pp. 1–27.

[10] M. Do and M. Vetterli, *The finite ridgelet transform for image representation*, IEEE Trans. Image Process., 12 (2003), pp. 16–28.

[11] D. Donoho and A. Flesia, *Digital ridgelet transform based on true ridge functions*, in Beyond Wavelets, J. Stoeckler and G. Welland, eds., Academic Press, New York, 2003, pp. 1–33.

[12] S. Durand and J. Froment, *Reconstruction of wavelet coefficients using total variation minimization*, SIAM J. Sci. Comput., 24 (2003), pp. 1754–1767.

[13] A. Dutt and V. Rokhlin, *Fast Fourier transforms for nonequispaced data*, SIAM J. Sci. Comput., 14 (1993), pp. 1368–1393.

[14] B. Elbel and G. Steidl, *Fast Fourier transforms for nonequispaced data*, in Approximation Theory IX, Vol. 2: Computational Aspects, C. Chui and L. Schumaker, eds., Vanderbilt University Press, Nashville, TN, 1998, pp. 39–46.

[15] J. A. Fessler and B. P. Sutton, *NUFFT—Nonuniform FFT Toolbox for Matlab*, http://www.eecs.umich.edu/∼bpsutton/MR/Code/NUFFT/ (2002).

[16] J. A. Fessler and B. P. Sutton, *Nonuniform fast Fourier transforms using min-max interpolation*, IEEE Trans. Signal Process., 51 (2003), pp. 560–574.

[17] K. Fourmont, *Non-equispaced fast Fourier transforms with applications to tomography*, J. Fourier Anal. Appl., 9 (2003), pp. 431–450.

[18] L. Greengard and J.-Y. Lee, *Accelerating the nonuniform fast Fourier transform*, SIAM Rev., 46 (2004), pp. 443–454.

[19] N. Kingsbury, *Image processing with complex wavelets*, R. Soc. Lond. Philos. Trans. Ser. A Math. Phys. Eng. Sci., 357 (1999), pp. 2543–2560.

[20] N. Kingsbury, *Complex wavelets for shift invariant analysis and filtering of signals*, Appl. Comput. Harmon. Anal., 10 (2001), pp. 234–253.

[21] S. Kunis and D. Potts, *NFFT, Software Package, C Subroutine Library*, http://www.math.uni-luebeck.de/potts/nfft/ (2002).

[22] E. LePennec and S. Mallat, *Sparse geometric image representations with bandelets*, IEEE Trans. Image Process., 14 (2005), pp. 423–438.

[23] J. Ma, *Towards artifact-free characterization of surface topography using complex wavelets and total variation minimization*, Appl. Math. Comput., 170 (2005), pp. 1014–1030.

[24] J. Ma, X. Jiang, and L. Blunt, *Complex wavelet transform for extraction of morphological features on surface topography*, in Euspen's 4th International Conference, Glasgow, UK, 2004, pp. 350–351.

[25] J. Ma, X. Jiang, and P. Scott, *Complex ridgelets for shift invariant characterization of surface topography with line singularities*, Phys. Lett. A, 344 (2005), pp. 423–431.

[26] F. Malgouyres, *A framework for image deblurring using wavelet packet bases*, Appl. Comput. Harmon. Anal., 12 (2002), pp. 309–331.

[27] F. Natterer, *The Mathematics of Computerized Tomography*, B. G. Teubner, Stuttgart, 1986.

[28] J. Neumann and G. Steidl, *Dual-tree complex wavelet transform in the frequency domain and an application to signal classification*, Int. J. Wavelets Multiresolut. Inf. Process., 3

(2005), pp. 43–65.
[29] D. Potts and G. Steidl, *Fourier reconstruction of functions from their nonstandard sampled Radon transform*, J. Fourier Anal. Appl., 8 (2002), pp. 513–533.
[30] D. Potts, G. Steidl, and M. Tasche, *Fast Fourier transforms for nonequispaced data: A tutorial*, in Modern Sampling Theory, Appl. Numer. Harmon. Anal., Birkhäuser Boston, Boston, MA, 2001, pp. 247–270.
[31] L. Rudin, S. Osher, and E. Fatemi, *Nonlinear total variation based noise removal algorithm*, Phys. D, 60 (1992), pp. 259–268.
[32] I. W. Selesnick, *The design of approximate Hilbert transform pairs of wavelet bases*, IEEE Trans. Signal Process., 50 (2002), pp. 1144–1152.
[33] J. Starck, E. Candès, and D. Donoho, *The curvelet transform for image denoising*, IEEE Trans. Image Process., 11 (2002), pp. 670–684.
[34] G. Steidl, *A note on fast Fourier transforms for nonequispaced grids*, Adv. Comput. Math., 9 (1998), pp. 337–352.
[35] G. Steidl, J. Weickert, T. Brox, P. Mrázek, and M. Welk, *On the equivalence of soft wavelet shrinkage, total variation diffusion, total variation regularization, and SIDEs*, SIAM J. Numer. Anal., 42 (2004), pp. 686–713.
[36] C. Vogel and M. Oman, *Fast, robust total variation-based reconstruction of noisy, blurred images*, IEEE Trans. Image Process., 7 (1998), pp. 813–824.
[37] J. Weickert, *Anisotropic Diffusion in Image Processing*, Eur. Consort. Math. Ind. (Berl.), B. G. Teubner, Stuttgart, 1998.
[38] A. Yip and F. Park, *Solution Dynamics, Causality, and Critical Behaviour of the Regularization Parameter in Total Variation Denoising Problems*, CAM Tech. Report 03-59, UCLA, Los Angeles, CA, 2003.

SIAM J. SCI. COMPUT.
Vol. 28, No. 3, pp. 1001–1028
© 2006 Society for Industrial and Applied Mathematics

NEUMANN–NEUMANN DOMAIN DECOMPOSITION PRECONDITIONERS FOR LINEAR-QUADRATIC ELLIPTIC OPTIMAL CONTROL PROBLEMS*

MATTHIAS HEINKENSCHLOSS† AND HOANG NGUYEN‡

Abstract. We present a class of domain decomposition (DD) preconditioners for the solution of elliptic linear-quadratic optimal control problems. Our DD preconditioners are extensions of Neumann–Neumann DD preconditioners, which have been successfully applied to the solution of single PDEs. The DD preconditioners are based on a decomposition of the optimality conditions for the elliptic linear-quadratic optimal control problem into smaller subdomain optimality conditions with Dirichlet boundary conditions for the states and the adjoints on the subdomain interfaces. These subdomain optimality conditions are coupled through Neumann interface conditions for the states and the adjoints. This decomposition leads to a Schur complement system in which the unknowns are the state and adjoint variables on the subdomain interfaces. The Schur complement operator is the sum of subdomain Schur complement operators, the application of which is shown to correspond to the solution of subdomain elliptic linear-quadratic optimal control problems, which are essentially smaller copies of the original optimal control problem. We show that, under suitable conditions, the application of the inverse of the subdomain Schur complement operators requires the solution of a subdomain elliptic linear-quadratic optimal control problem with Neumann interface conditions for the state. The subdomain Schur complement operators are analyzed in the variational setting of the problem as well as the algebraic setting obtained after a finite element discretization of the problem. Definiteness properties of the algebraic form of the (subdomain) Schur complement operator(s) are studied. Numerical tests show that the dependence of these preconditioners on mesh size and subdomain size is comparable to its counterpart applied to elliptic equations only. These tests also show that the preconditioners are insensitive to the size of the control regularization parameter.

Key words. optimal control, preconditioners, domain decomposition, Neumann–Neumann methods

AMS subject classifications. 49M05, 65N55, 65F10, 90C06, 90C20

DOI. 10.1137/040612774

1. Introduction.

This paper is concerned with a class of nonoverlapping domain decomposition (DD) preconditioners for linear-quadratic elliptic optimal control problems. The solution of linear-quadratic elliptic optimal control problems arises in many applications, either directly or as subproblems in Newton or sequential quadratic programming methods for the solution of nonlinear elliptic optimal control problems. After a finite element discretization, convex linear-quadratic elliptic optimal control problems lead to large-scale symmetric indefinite linear systems. The solution of these large systems is a very time consuming step and must be done iteratively, typically with a preconditioned Krylov subspace method. Developing good preconditioners for these linear systems is an important part of improving the overall performance of the solution method.

*Received by the editors August 3, 2004; accepted for publication (in revised form) February 13, 2006; published electronically June 23, 2006. This work was supported in part by NSF grant ACI-0121360 and by the Los Alamos National Laboratory Computer Science Institute (LACSI) through LANL contract 03891-99-23 as part of the prime contract (W-7405-ENG-36) between the Department of Energy and the Regents of the University of California.

http://www.siam.org/journals/sisc/28-3/61277.html

†Department of Computational and Applied Mathematics, MS-134, Rice University, 6100 Main Street, Houston, TX 77005-1892 (heinken@rice.edu).

‡Department of Biostatistics and Applied Mathematics, University of Texas, MD Anderson Cancer Center, 1515 Holcombe Blvd., Unit 447, Houston, TX 77030-4009 (honguyen@mdanderson.org).

To illustrate our ideas, we consider the example problem

$$\text{(1.1a)} \qquad \text{minimize} \quad \frac{1}{2}\int_{\Omega_o}(y(x)-\hat{y}(x))^2dx+\frac{\alpha}{2}\int_{\partial\Omega}u^2(x)dx,$$

$$\text{(1.1b)} \qquad \text{subject to} \quad -\Delta y(x)+\sigma y(x)=\widehat{f}(x) \qquad \text{in } \Omega,$$

$$\text{(1.1c)} \qquad \frac{\partial}{\partial n}y(x)=u(x) \qquad \text{on } \partial\Omega,$$

where $\hat{y}, \widehat{f}$ are given functions and $\alpha > 0$ and $\sigma \geq 0$ are given parameters. The problem (1.1) has to be solved for y and u. Detailed model problem assumptions will be introduced in section 3. While (1.1) is used to illustrate the DD preconditioners introduced in this paper, their formulation and many of their properties carry over to more general linear-quadratic elliptic optimal control problems.

Our DD preconditioners for linear-quadratic elliptic optimal control problems such as (1.1) are based on a decomposition of the system of necessary and sufficient optimality conditions. We decompose Ω into nonoverlapping subdomains. We then introduce state and adjoint variables on the subdomain interfaces and formulate subdomain optimality conditions with Dirichlet boundary conditions for states and adjoints on the subdomain interfaces. These subdomain optimality conditions are coupled by requiring continuity of normal derivatives of subdomain state and adjoints, defined as solutions of the local optimality conditions, on the subdomain interfaces. This leads to subdomain Schur complement operators that map Dirichlet interface data into Neumann interface data. The original optimality conditions are equivalent to an optimality Schur complement system in which the operator is given by the sum of subdomain Schur complements. The properties of the subdomain Schur complements allow the construction of preconditioners for the optimality condition Schur complement system that are extensions of the Neumann–Neumann methods, which have been successfully applied to single elliptic PDEs [13, 14, 24, 26, 33, 34, 36, 38].

We will show that the application of subdomain Schur complements corresponds to the solution of subdomain optimal control problems that are essentially smaller copies of the original optimal control problem but impose Dirichlet boundary conditions for the state on the subdomain interface and include an objective function term that implies Dirichlet boundary conditions for the adjoint on the subdomain interface. Under suitable conditions, these subdomain Schur complements are invertible. The application of the inverses of local Schur complement operators corresponds to the solution of subdomain optimal control problems that are also essentially smaller copies of the original optimal control problem but impose Neumann boundary conditions on the subdomain interfaces.

The paper is organized as follows. In the next section, we give a brief overview of other DD approaches to solve linear-quadratic optimal control problems and describe how our approach differs from existing ones.

In section 3 we study the model problem (1.1) with emphasis on the case $\sigma = 0$. In this case, the state equation (1.1b), (1.1c) has a solution if and only if the control satisfies a certain compatibility condition; the solution of (1.1b), (1.1c), if it exists, is not unique. In section 3 we will show that the optimal control problem (1.1) has a unique solution. However, the issues outlined above regarding the existence and uniqueness of the solution of the state equation (1.1b), (1.1c) later may lead to singularities in the subproblems arising in our DD methods. This is one reason for choosing the model problem (1.1).

Our DD approach is introduced in section 4. This section introduces the Schur complement formulation of the optimality system and investigates properties of the subdomain Schur complement. The presentation in section 4 is based on the variational form of the optimal control problem (1.1).

The algebraic view of our DD approach is presented in section 5. This view emphasizes the formal similarity between the Neumann–Neumann methods for single elliptic PDEs and our approach. In section 5 we will also investigate the inertia of the Schur complements arising in our DD preconditioners. We will see that the Schur complements arising in the optimal control context are highly indefinite; the number of positive and negative eigenvalues is essentially the same and is equal to the number of discrete state variables on the subdomain interfaces.

Numerical results for our (balancing) Neumann–Neumann preconditioners applied to the model problem (1.1) are given in section 6. The results indicate that the numerical performance of our (balancing) Neumann–Neumann DD methods applied to linear-quadratic elliptic optimal control problems is similar to that of their counterparts applied to single elliptic PDEs. We also observe that the performance of our preconditioners applied to the model problem (1.1) is largely independent of the size of the regularization parameter.

This work significantly expands the material in [20] in which an additional numerical example may be found.

2. Comparison with other domain decomposition approaches. DD-based iterative methods for the solution of linear-quadratic elliptic optimal control problems such as (1.1) can be split into three categories, depending on how the DD is integrated with the optimization. In all cases DD introduces parallelism into the optimization but affects the (outer) iteration differently. We give an outline and refer to [32, sect. 1.2] for more details.

1. The methods in the first category apply preconditioned Krylov subspace methods to the system of discretized optimality conditions (also known as the KKT system) for (1.1). This system has a 3×3 block structure, and the preconditioners for the system matrix (also known as the KKT matrix) are obtained from approximate block factorizations. DD methods are used for the approximate solution of systems involving the stiffness matrix and its transpose. DD is used at the PDE level. These methods require a good preconditioner for the so-called reduced Hessian, which can be difficult to obtain. The methods in [9, 10] belong to this category.

2. The methods in the second category use DD to reformulate the optimization problem (1.1) or its discretization [11, 28]. For example, a reformulation may eliminate state variables in the interior of subdomains by viewing them as functions of the controls in that subdomain as well as state variables on the subdomain interfaces. The optimization variables visible to the outer iteration then consist of the control variables and the state variables restricted to the subdomain interfaces. The optimality conditions for the reformulated problem can be viewed as a Schur complement of the KKT matrix corresponding to the original problem (1.1). This Schur complement involves the state and adjoint variables on the subdomain interfaces as well as all control variables. In contrast, the Schur complement formulation introduced in this paper involves only the state and adjoint variables on the subdomain interfaces. The application of the Schur complement arising in the DD methods in category 2 to a vector requires the solution of subdomain PDEs. Typically, the Schur complements arising in the methods in category 2 are better conditioned than the original KKT matrix, but additional preconditioners are required, especially when α or the mesh

size h is small (see [32, sect. 1.2] for an example). The approach in [12] for a nonlinear parameter identification problem also falls into this category.

3. In [5, 6, 7] DD methods for linear-quadratic elliptic optimal control problems are introduced that are based on a decomposition of the optimality conditions. The resulting iterative method requires the solution of KKT-type systems restricted to smaller domains, and the variables in the iteration are related to states and adjoints on the subdomain interfaces. In [5, 6, 7] Robin-Robin-type transmission conditions are used to connect the subdomain problems. Convergence of the method is proved in [6, 7], but no convergence rates are given. No results regarding the dependence of the convergence on mesh size, subdomain size, and regularization parameter α are given. The KKT-type systems restricted to smaller domains are related to subdomain optimal control problems, but these optimal control problems have a structure different from that of the original ones. In particular, the state space for the subdomain optimal control problems in [6, p. 2411] is twice as large as that of the original optimal control problems restricted to the same subdomain.

DD methods for a class of nonlinear parameter identification problems are investigated in [25, 37]. The augmented Lagrangian method is used for the solution of the nonlinear optimization problem, and DD approaches for unconstrained minimization are used to solve the subproblems within the augmented Lagrangian method.

The methods proposed in this paper, like those in [5, 6, 7], are based on a decomposition of the optimality conditions but use different transmission conditions. One consequence is that the DD methods proposed in this paper require the solution of KKT systems restricted to smaller subdomains, which are related to smaller copies of the original linear-quadratic elliptic optimal control problem. Hence existing code can be reused for subproblem solves. The DD methods proposed in this paper are extensions of the Neumann–Neumann methods, which have been successfully applied to single elliptic PDEs [13, 14, 24, 26, 33, 34, 36, 38]. The numerical performance of our (balancing) Neumann–Neumann DD methods applied to the model problem (1.1) indicates that their performance is similar to that of their counterparts applied to single elliptic PDEs. We also observe that their performance on the model problem (1.1) is largely independent of the size of the regularization parameter. This is important since the condition numbers of the KKT matrix corresponding to (1.1) and of the reduced Hessian grow linearly with α^{-1} (see [32, Chap. 1] for an example). The subdomain problems that need to be solved in our (balancing) Neumann–Neumann DD methods are essentially smaller subdomain copies of (1.1), and state and adjoint variables restricted to subdomain interfaces have to be communicated between adjacent subdomains. Hence our methods, like those in [5, 6, 7], involve more computations per communication than those in categories 1 and 2 sketched above.

The DD preconditioners proposed in this paper complement existing preconditioners for KKT systems, such as those in [1, 3, 4, 8, 9, 10, 21, 15, 16, 18, 19, 23, 29, 31], which can be used as subproblem solvers within our DD preconditioners. Such a combination introduces parallelism. For some applications, such a combination may make existing preconditioners that rely on sparse matrix factorizations feasible, since they now are applied to smaller subproblems.

3. The example problem. In this section, we define the setting for the model problem (1.1), establish existence and uniqueness of its solution, and derive necessary and sufficient optimality conditions. If $\sigma > 0$, the state equation (1.1b) has a unique solution y for given control u. In this case existence and uniqueness of solutions as well as derivation of necessary and sufficient optimality conditions are standard (see,

e.g., [27]). The case $\sigma = 0$ is more interesting and is the focus of this section. In the case $\sigma = 0$, the state equation (1.1b) is solvable if and only if the control satisfies a certain compatibility condition; if (1.1b) is solvable, the solution is not unique.

Let $\Omega \subset \mathbb{R}^d$, $d = 2, 3$, be an open, bounded set with Lipschitz boundary, let $\Omega_o \subset \Omega$ be measurable with $\operatorname{meas}(\Omega_o) > 0$, let $\hat{y} \in L^2(\Omega_0)$, $\widehat{f} \in L^2(\Omega)$ be given functions, and let $\alpha > 0$, $\sigma \geq 0$ be given parameters. We define the (bi)linear forms

$$a : H^1(\Omega) \times H^1(\Omega) \to \mathbb{R}, \qquad a(y, \psi) = \int_\Omega \nabla y(x) \nabla \psi(x) + \sigma y(x) \psi(x) dx,$$

$$b : L^2(\partial\Omega) \times H^1(\Omega) \to \mathbb{R}, \qquad b(u, \psi) = -\int_{\partial\Omega} u(x) \psi(x) dx,$$

$$m : H^1(\Omega) \times H^1(\Omega) \to \mathbb{R}, \qquad m(y, \psi) = \int_{\Omega_o} y(x) \psi(x) dx,$$

$$q : L^2(\partial\Omega) \times L^2(\partial\Omega) \to \mathbb{R}, \qquad q(u, \mu) = \alpha \int_{\partial\Omega} u(x) \mu(x) dx,$$

$$f : H^1(\Omega) \to \mathbb{R}, \qquad f(\psi) = \int_\Omega \widehat{f}(x) \psi(x) dx.$$

We are interested in the solution $y \in H^1(\Omega)$, $u \in L^2(\partial\Omega)$ of the optimal control problem

$$\text{minimize} \quad \tfrac{1}{2} m(y - \hat{y}, y - \hat{y}) + \tfrac{1}{2} q(u, u), \tag{3.1a}$$

$$\text{subject to} \quad a(y, \phi) + b(u, \phi) = f(\phi) \quad \forall \phi \in H^1(\Omega). \tag{3.1b}$$

THEOREM 3.1. [illegible] $\sigma \geq 0$ [illegible] (3.1) [illegible] $(u_*, y_*) \in L^2(\partial\Omega) \times H^1(\Omega)$ [illegible] $p_* \in H^1(\Omega)$ [illegible]

$$a(\psi, p) + m(y, \psi) = m(\hat{y}, \psi) \qquad \forall \psi \in H^1(\Omega), \tag{3.2a}$$

$$q(u, \mu) + b(\mu, p) = 0 \qquad \forall \mu \in L^2(\partial\Omega), \tag{3.2b}$$

$$a(y, \phi) + b(u, \phi) = f(\phi) \qquad \forall \phi \in H^1(\Omega). \tag{3.2c}$$

[illegible] (3.2) [illegible] $y_* \in H^1(\Omega)$ $u_* \in L^2(\partial\Omega)$ $p_* \in H^1(\Omega)$ [illegible]. For $\sigma > 0$ the state equation (3.1b) has a unique solution y for any $u_* \in L^2(\partial\Omega)$ and $f \in (H^1(\Omega))'$. In this case, the assertion of the theorem is well known and a proof can be found, e.g., in [27] or by a modification of the arguments below.

Let $\sigma = 0$. We recall that for given $l \in (H^1(\Omega))'$ the equation

$$a(y, \phi) = l(\phi)$$

has a solution if and only if $l(1) = 0$. If $l(1) = 0$ and if y_0 is a solution, then the set of solutions is given by $y_0 + \gamma$, $\gamma \in \mathbb{R}$.

(i) [illegible] We define

$$Y = \left\{ y \in H^1(\Omega) \ : \ \int_\Omega y(x)dx = 0 \right\}.$$

By the second Poincaré inequality (see, e.g., [35, Cor. 6.100], [39, Thm. 7.7]),

$$\|y\|_Y = \|\nabla y\|_{L^2(\Omega)}$$

defines a norm on Y which on Y is equivalent to $\|\cdot\|_{H^1(\Omega)}$. Moreover, Y equipped with $\|\cdot\|_Y$ is a Hilbert space. Standard arguments can be used to show that if $u \in L^2(\partial\Omega)$ satisfies $f(1) = b(u,1)$, then (3.1b) has a unique solution $y \in Y$ and there exists $\kappa > 0$, independent of u and f such that

$$\|y\|_Y \le \kappa(\|u\|_{L^2(\partial\Omega)} + \|f\|_{(H^1(\Omega))'}).$$

The problem (3.1), which is an optimization problem in $u \in L^2(\partial\Omega)$, $y \in H^1(\Omega)$, can be written as an optimization problem in $u \in L^2(\partial\Omega)$, $y_0 \in Y$, $\gamma \in \mathbb{R}$ given by

$$\text{(3.3a)} \qquad \text{minimize} \quad \tfrac{1}{2}m(y_0 + \gamma - \hat{y}, y_0 + \gamma - \hat{y}) + \tfrac{1}{2}q(u,u),$$

$$\text{(3.3b)} \qquad \text{subject to} \quad a(y_0,\psi) + b(u,\psi) = f(\psi) \quad \forall \psi \in H^1(\Omega).$$

Using the affine linear map $u \mapsto y_0(u)$ that maps $u \in L^2(\partial\Omega)$ with $b(u,1) = f(1)$ to the solution $y_0(u) \in Y$ of (3.3b), we can write (3.3) as

$$\text{(3.4a)} \qquad \text{minimize} \quad \tfrac{1}{2}m(y_0(u) + \gamma - \hat{y}, y_0(u) + \gamma - \hat{y}) + \tfrac{1}{2}q(u,u),$$

$$\text{(3.4b)} \qquad \text{subject to} \quad b(u,1) = f(1).$$

For fixed u satisfying (3.4b), the unique solution of

$$\min_\gamma \tfrac{1}{2}m(y_0(u) + \gamma - \hat{y}, y_0(u) + \gamma - \hat{y})$$

is given by

$$\gamma(u) = -\int_{\Omega_o} y_0(u;x) - \hat{y}(x)dx \Big/ \int_{\Omega_o} dx. \tag{3.5}$$

Note that

$$\int_{\Omega_o} y_0(u;x) + \gamma(u) - \hat{y}(x)dx = 0. \tag{3.6}$$

Using the definition (3.5) of $\gamma(u)$, (3.4) can be written as the following optimization problem in u only:

$$\text{(3.7a)} \qquad \text{Minimize} \quad \tfrac{1}{2}m(y_0(u) + \gamma(u) - \hat{y}, y_0(u) + \gamma(u) - \hat{y}) + \tfrac{1}{2}q(u,u),$$

$$\text{(3.7b)} \qquad \text{subject to} \quad b(u,1) = f(1).$$

Since $u \mapsto y_0(u)$ is an affine linear map, $u \mapsto y_0(u) + \gamma(u)$ is affine linear and (3.7a) is strictly convex. The existence and uniqueness of the optimal control u_* follows by standard arguments (see, e.g., [22]). The unique corresponding state is given by $y_* = y_0(u_*) + \gamma(u_*)$.

(ii) [illegible]. The optimality conditions follow from an application of the Lagrange multiplier theorem (see, e.g., [22]). First, we show that the linear constraints are surjective. Let $f \in (H^1(\Omega))'$ be arbitrary. If we define $u = -\|f\|_{(H^1(\Omega))'}/\int_{\partial\Omega} dx$, then the state equation (3.2c) has a solution $y \in H^1(\Omega)$. This proves the surjectivity of the constraints.

Since the constraints are surjective, there exists $p \in H^1(\Omega)$ such that the partial Fréchet derivatives of

$$L(y,u,p) = \tfrac{1}{2}m(y-\hat{y},y-\hat{y}) + \tfrac{1}{2}q(u,u) + a(y,p) + b(y,p) - f(p)$$

with respect to y, u, and p are equal to zero. This gives (3.2). Since the optimal control problem (3.1) is convex, the necessary optimality conditions are also sufficient.

(iii) [illegible]. In part (i) we have already shown the uniqueness of the optimal state and control. The uniqueness of the associated adjoint p_* remains to be shown. We write the optimal state, control, and adjoint as

$$y_* = y_0 + \gamma, \quad u_* = u_0 + u_f, \quad p_* = p_0 + \eta,$$

where $y_0, p_0 \in Y$, $u_0 \in \{u \in L^2(\partial\Omega) \,:\, b(u,1) = 0\}$, $u_f = -\|f\|_{(H^1(\Omega))'}/\int_{\partial\Omega} dx$, and $\gamma, \eta \in \mathbb{R}$. The function y_0 is the unique solution (in Y) of the state equation (3.2c), and γ is given by (3.5). Since the optimal state $y_* = y_0 + \gamma$ satisfies (3.6), the adjoint equation (3.2a) has a solution $p_* = p_0 + \eta$, where $p_0 \in Y$ is unique. Using the representation of u_* and p_* as well as the decomposition $L^2(\partial\Omega) = \{u \in L^2(\partial\Omega) \,:\, b(u,1) = 0\} \oplus \operatorname{span}\{1\}$ (by definition of b, $\{u \in L^2(\partial\Omega) \,:\, b(u,1) = 0\}$ is the orthogonal complement of $\operatorname{span}\{1\}$ in $L^2(\partial\Omega)$), we can write condition (3.2b) equivalently as

$$\begin{aligned} &q(u_0 + u_f, 1) + b(1, p_0 + \eta) = 0, \\ &q(u_0 + u_f, \mu) + b(\mu, p_0 + \eta) = 0, \quad \mu \in \{u \in L^2(\partial\Omega) \,:\, b(u,1) = 0\}. \end{aligned}$$

The first condition implies

$$\eta = -(q(u_0 + u_f, 1) + b(1, p_0))/b(1,1),$$

and the second condition characterizes $u_0 \in \{u \in L^2(\partial\Omega) \,:\, b(u,1) = 0\}$. ∎

Note that in the case $\sigma = 0$, adjoint equation (3.2a) alone does not have a unique solution, but because the optimal state $y_* = y_0(u_*) + \gamma(u_*)$ satisfies (3.6), the adjoint equation (3.2a) is solvable. The entire optimality system (3.2) specifies optimal state, control, and adjoint uniquely.

4. Domain decomposition Schur complement formulation of the example problem.

4.1. Domain decomposition of the example problem. We discretize (3.1) using conforming linear finite elements. Let $\{T_l\}$ be a triangulation of Ω. We divide Ω into nonoverlapping subdomains Ω_i, $i = 1, \ldots, s$, such that each T_l belongs to exactly one $\overline{\Omega}_i$.

We define

$$\Gamma_i = \partial\Omega_i \setminus \partial\Omega$$

and

$$\Gamma = \bigcup_{i=1}^{s} \Gamma_i.$$

The unit outward normal of Ω_i is denoted by n_i. By Tr_{Γ_i} we denote the trace on Γ_i.

We split the subdomains into no-control subdomains

$$\mathcal{N} = \{i \ : \ \mathrm{int}(\partial\Omega_i \cap \partial\Omega) = \emptyset\}$$

and control subdomains

$$\mathcal{C} = \{i \ : \ \mathrm{int}(\partial\Omega_i \cap \partial\Omega) \neq \emptyset\}.$$

Here int denotes the relative interior.

The state y is approximated using piecewise linear functions. We define the finite dimensional spaces

$$\begin{aligned} V^h &= \left\{v \in H^1(\Omega) \ : \ v|_{T_l} \in P^1(T_l) \ \forall l\right\}, \\ V_i^h &= \left\{v \in H^1(\Omega_i) \ : \ v|_{T_l} \in P^1(T_l) \ \forall T_l \subset \overline{\Omega}_i\right\}, \qquad i = 1, \ldots, s, \\ V_{i,0}^h &= \left\{v \in V_i^h \ : \ v = 0 \text{ on } \Gamma_i\right\}, \qquad i = 1, \ldots, s. \end{aligned} \tag{4.1}$$

We can identify $v_i \in V_{i,0}^h$ with a function in V^h if we extend v_i by zero onto Ω. Furthermore, we introduce the "trace" space V_{i,Γ_i}^h of functions in V_i^h that are zero at vertices in $\overline{\Omega}_i \setminus \Gamma_i$ and the "trace" space V_Γ^h of functions in V^h that are zero at vertices in $\overline{\Omega} \setminus \Gamma$. Hence, each function $v^h \in V^h$ can be written as

$$V^h = \bigoplus_{i=1}^{s} V_{i,0}^h \oplus V_\Gamma^h. \tag{4.2}$$

For our discretization of the control, we use piecewise linear functions on $\partial\Omega$. However, our discretization of the control is somewhat nonstandard. Typically, one would choose the space of discretized controls to be

$$U^h = \left\{u \in C^0(\partial\Omega) \ : \ u \text{ is linear on } \partial\Omega \cap \partial T_l \ \forall T_l \subset \overline{\Omega}\right\}. \tag{4.3}$$

A DD formulation based on such a discretization would introduce "interface controls" (dotted hat function in the left-hand plot in Figure 4.1) defined on a "band" of width $O(h)$ around $\partial\Omega \cap \partial\Omega_i \cap \partial\Omega_j$, $i \neq j$. See the left-hand plot in Figure 4.1. Since the evaluation of $u \in L^2(\partial\Omega)$ on $\partial\Omega \cap \partial\Omega_i \cap \partial\Omega_j$ does not make sense, we avoid interface controls.

We discretize the control u by a function which is continuous on each $\partial\Omega_i$, $i = 1, \ldots, s$, and linear on each $\partial\Omega \cap \partial\Omega_i \cap T_l$. The discretized control is not assumed to be continuous at $\partial\Omega \cap \partial\Omega_i \cap \partial\Omega_j$, $i \neq j$. In particular, for each point $x_k \in \partial\Omega \cap \partial\Omega_i \cap \partial\Omega_j$, $i \neq j$, there are two discrete controls u_{k_i}, u_{k_j} belonging to subdomains Ω_i and Ω_j, respectively (see the right-hand plot in Figure 4.1). Hence, our control discretization depends on the partition $\{\Omega_i\}_{i=1}^d$ of the domain Ω. We define the discrete control spaces

$$U_i^h = \left\{u \in C^0(\partial\Omega \cap \partial\Omega_i) \ : \ u \text{ is linear on } \partial\Omega \cap \partial\Omega_i \cap \partial T_l \ \forall T_l \subset \overline{\Omega}_i\right\}. \tag{4.4}$$

We identify U_i^h with a subspace of $L^2(\partial\Omega)$ by extending functions $u_i \in U_i^h$ by zero onto $\partial\Omega$. We define

$$U^h = \bigcup_{i=1}^{s} U_i^h \subset L^2(\partial\Omega).$$

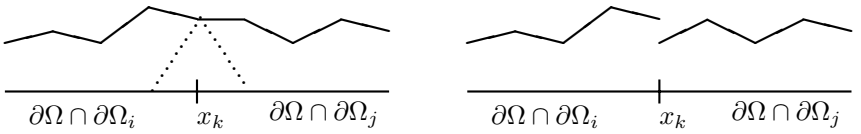


FIG. 4.1. *Sketch of the control discretization for the case* $\Omega \subset \mathbb{R}^2$.

In particular,

$$U^h = \bigoplus_{i=1}^{s} U_i^h. \tag{4.5}$$

Our discretization of (3.2) is given by

$$a(\psi, p) + m(y, \psi) = m(\hat{y}, \psi) \qquad \forall \psi \in V^h, \tag{4.6a}$$

$$q(u, \mu) + b(\mu, p) = 0 \qquad \forall \mu \in U^h, \tag{4.6b}$$

$$a(y, \phi) + b(u, \phi) = f(\phi) \qquad \forall \phi \in V^h. \tag{4.6c}$$

The system (4.6) is also the system of necessary and sufficient optimality conditions for the discretization of (3.1) using piecewise linear finite elements for the state and controls, as described above.

We decompose the optimality system (4.6) using (4.2), (4.5). For this purpose, we introduce the following (bi)linear forms. Let

$$a_i : V_i^h \times V_i^h \to \mathbb{R}, \qquad a_i(y, \psi) = \int_{\Omega_i} \nabla y(x) \nabla \psi(x) + \sigma y(x) \psi(x) dx,$$

$$m_i : V_i^h \times V_i^h \to \mathbb{R}, \qquad m_i(y, \psi) = \int_{\Omega_i \cap \Omega_o} y(x) \psi(x) dx,$$

$$f_i : V_i^h \to \mathbb{R}, \qquad f_i(\psi) = \int_{\Omega_i} \widehat{f}(x) \psi(x) dx,$$

and, if Ω_i is a control subdomain,

$$b_i : U_i^h \times V_i^h \to \mathbb{R}, \qquad b_i(u, \psi) = -\int_{\partial\Omega_i \cap \partial\Omega} u(x) \psi(x) dx,$$

$$q_i : U_i^h \times U_i^h \to \mathbb{R}, \qquad q_i(u, \mu) = \alpha \int_{\partial\Omega_i \cap \partial\Omega} u(x) \mu(x) dx.$$

The parameters $\alpha > 0$ and $\sigma \geq 0$ as well as $\widehat{f} \in L^2(\Omega)$ are given as in section 3.

Now we write $y = y_\Gamma + \sum_{i=1}^s y_i$, $p = p_\Gamma + \sum_{i=1}^s p_i$, and $u = \sum_{i=1}^s u_i$, where $y_\Gamma, p_\Gamma \in V_\Gamma^h$, $y_i, p_i \in V_{i,0}^h$, $i = 1, \ldots, s$, and $u_i \in U_i^h$, $i = 1, \ldots, s$ (cf. (4.2) and (4.5)), and insert these representations into (4.6). Using an analogous decomposition of the test functions, this leads to the systems

$$a_i(\psi, p_i) + m_i(y_i, \psi) = m_i(\hat{y}, \psi) \qquad \forall \psi \in V_{i,0}^h, \tag{4.7a}$$

$$b_i(\mu, p_i) + q_i(u_i, \mu) = 0 \qquad \forall \mu \in U_i^h, \tag{4.7b}$$

$$a_i(y_i, \psi) + b_i(u_i, \psi) = f_i(\psi) \qquad \forall \psi \in V_{i,0}^h, \tag{4.7c}$$

$$y_i = y_\Gamma, \quad p_i = p_\Gamma \qquad \text{on } \Gamma_i \tag{4.7d}$$

for $i \in \mathcal{C}$,

$$a_i(\psi, p_i) + m_i(y_i, \psi) = m_i(\hat{y}, \psi) \qquad \forall \psi \in V_{i,0}^h, \tag{4.8a}$$

$$a_i(y_i, \psi) = f_i(\psi) \qquad \forall \psi \in V_{i,0}^h, \tag{4.8b}$$

$$y_i = y_\Gamma, \quad p_i = p_\Gamma \qquad \text{on } \Gamma_i \tag{4.8c}$$

for $i \in \mathcal{N}$, and

$$\sum_{i=1}^{s} a_i(y_i, q_\Gamma) + b_i(u_i, q_\Gamma) + a_i(v_\Gamma, p_i) + m_i(y_i, v_\Gamma) = \sum_{i=1}^{s} f_i(q_\Gamma) + m_i(\hat{y}_i, v_\Gamma) \quad \forall\, v_\Gamma, q_\Gamma \in V_\Gamma^h. \tag{4.9}$$

The next theorem states that solving the optimality conditions (4.6) is equivalent to finding $(y_\Gamma, p_\Gamma) \in V_\Gamma^h \times V_\Gamma^h$ such that the solutions $(y_i, u_i, p_i) \in V_i^h \times U_i^h \times V_i^h$, $i \in \mathcal{C}$, and $(y_i, p_i) \in V_i^h \times V_i^h$, $i \in \mathcal{N}$, of (4.7) and (4.8), respectively, satisfy the interface conditions (4.9).

THEOREM 4.1. [illegible] $(y, u, p) \in V^h \times U^h \times V^h$ [illegible] (4.6) [illegible] $y_i = y|_{\Omega_i}$ $p_i = p|_{\Omega_i}$ $i = 1, \ldots, s$ $u_i = u|_{\Omega_i}$ $i \in \mathcal{C}$ [illegible] (4.7) (4.9) [illegible] $(y_\Gamma, p_\Gamma) \in V_\Gamma^h \times V_\Gamma^h$ [illegible] $(y_i, u_i, p_i) \in V_i^h \times U_i^h \times V_i^h$ $i \in \mathcal{C}$ [illegible] $(y_i, p_i) \in V_i^h \times V_i^h$ $i \in \mathcal{N}$ [illegible] (4.7) [illegible] (4.8) [illegible] (4.9) [illegible] $(y, u, p) \in V^h \times U^h \times V^h$ [illegible] $y_i = y|_{\Omega_i}$ $p_i = p|_{\Omega_i}$ $i = 1, \ldots, s$ $u_i = u|_{\Omega_i}$ $i \in \mathcal{C}$ [illegible] (4.6)

The proof of this theorem is analogous to the proof of [34, Lem. 1.2.1] and is omitted.

Our next goal is to view the solution of (4.7), (4.8) as an affine linear function of (y_Γ, p_Γ) and then consider (4.9) as a linear equation in (y_Γ, p_Γ). This Schur complement system will be solved using a preconditioned Krylov subspace method. To derive this Schur complement system, however, we first have to investigate the solvability of (4.7) and (4.8).

4.2. The subdomain optimality systems. We investigate the solvability of the systems (4.7) and (4.8). These are the weak forms of

$$-\Delta y_i(x) + \sigma y_i(x) = f(x) \qquad \text{in } \Omega_i, \tag{4.10a}$$

$$\frac{\partial}{\partial n} y_i(x) = u_i(x) \qquad \text{on } \partial\Omega \cap \partial\Omega_i, \tag{4.10b}$$

$$y_i(x) = y_\Gamma(x) \qquad \text{on } \Gamma_i, \tag{4.10c}$$

$$-\Delta p_i(x) + \sigma p_i(x) = -(y_i(x) - \hat{y}(x))|_{\Omega_o} \qquad \text{in } \Omega_i, \tag{4.10d}$$

$$\frac{\partial}{\partial n} p_i(x) = 0 \qquad \text{on } \partial\Omega \cap \partial\Omega_i, \tag{4.10e}$$

$$p_i(x) = p_\Gamma(x) \qquad \text{on } \Gamma_i, \tag{4.10f}$$

$$\alpha u_i(x) - p_i(x) = 0 \qquad \text{on } \partial\Omega \cap \partial\Omega_i \tag{4.10g}$$

and

$$-\Delta y_i(x) + \sigma y_i(x) = f(x) \qquad \text{in } \Omega_i, \tag{4.11a}$$

$$y_i(x) = y_\Gamma(x) \qquad \text{on } \Gamma_i, \tag{4.11b}$$

$$-\Delta p_i(x) + \sigma p_i(x) = -(y_i(x) - \hat{y}(x))|_{\Omega_o} \qquad \text{in } \Omega_i, \tag{4.11c}$$

$$p_i(x) = p_\Gamma(x) \qquad \text{on } \Gamma_i, \tag{4.11d}$$

respectively.

We will show that (4.7), (4.8) have unique solutions, even if $\sigma = 0$ (see Theorems 4.3 and 4.4 below). This is due to the presence of Dirichlet boundary conditions, which ensure that the state equations (4.7c) and (4.8b) have a unique solution for each $u_i \in U_i^h$ and that the adjoint equations (4.7a) and (4.8a) have a unique solution for each $y_i \in V_i^h$.

LEMMA 4.2. [illegible] $\sigma \geq 0$ [illegible] $l_i \in (V_{i,0}^h)^*$ [illegible]

$$a_i(y_i, \psi) = l_i(\psi) \quad \forall \psi \in V_{i,0}^h.$$

[illegible] $\kappa_i > 0$ [illegible] l_i [illegible] $\|y_i\|_{V_i^h} \leq \kappa_i \|l_i\|_{(V_i^h)^*}$ [illegible]. By the Poincaré inequality there exists $c_i > 0$ such that $c_i \|y_i\|_{V_i^h}^2 \leq a_i(y_i, y_i)$ for all $y_i \in V_{i,0}^h$. The lemma now follows from a standard application of the Lax–Milgram theorem. □

The system (4.8) is a system of elliptic differential equations in triangular form that can be easily solved by first determining $y_i \in V_i^h$ from the second equation in (4.8) and then $p_i \in V_i^h$ from the first equation in (4.8). The unique solvability of the two elliptic PDEs follows from the definition of V_{i,Γ_i}^h and from Lemma 4.2.

THEOREM 4.3. [illegible] (4.8) [illegible] $(y_i, u_i, p_i) \in V_i^h \times U_i^h \times V_i^h$

Next we will show that (4.7) are the optimality conditions for a convex linear-quadratic optimal control problem restricted to the subdomain Ω_i. Let Ω_i be a control subdomain. Formally the system (4.7) (or its corresponding strong form (4.10)) may be viewed as the necessary and sufficient optimality conditions for

(4.12a)
$$\text{minimize } \frac{1}{2}\int_{\Omega_o \cap \Omega_i} (y_i(x) - \hat{y}(x))^2 dx + \frac{\alpha}{2}\int_{\partial\Omega_i} u_i^2(x) dx + \int_{\Gamma_i} \frac{\partial}{\partial n_i} y_i(x) p_\Gamma(x) dx,$$

(4.12b)
$$\text{subject to } -\Delta y_i(x) + \sigma y_i(x) = \widehat{f}(x) \text{ in } \Omega_i,$$

(4.12c)
$$\frac{\partial}{\partial n} y_i(x) = u_i(x) \text{ on } \partial\Omega_i \cap \partial\Omega,$$

(4.12d)
$$y_i(x) = y_\Gamma(x) \text{ on } \Gamma_i.$$

The local optimal control problem (4.12) is an optimization problem in the variables u_i, y_i; all other quantities in (4.12) are given.

To rigorously establish a relationship between (4.7) and (4.12) we have to make precise the meaning of the state equation (4.12b)–(4.12d) and of the objective function (4.12a). The state equation (4.12b)–(4.12d) is understood in the weak sense (4.7c), (4.7d). Concerning the objective function (4.12a) we note that equations (4.12b), (4.12c) imply

$$\begin{aligned}
&\int_{\Gamma_i} \frac{\partial}{\partial n} y_i(x) \psi(x) dx \\
&= \int_{\Omega_i} \nabla y_i(x) \nabla \psi(x) + \sigma y_i(x) \psi(x) dx - \int_{\partial\Omega \cap \partial\Omega_i} u(x) \psi(x) dx - \int_{\Omega_i} \widehat{f}(x) \psi(x) dx, \\
&= a_i(y_i, \psi) + b_i(u_i, \psi) - f_i(\psi).
\end{aligned}$$

Thus, in (4.12a) we set

$$\int_{\Gamma_i} \frac{\partial}{\partial n} y_i(x) p_\Gamma(x) dx = a_i(y_i, \mathcal{R}_i p_\Gamma) + b_i(u_i, \mathcal{R}_i p_\Gamma) - f_i(\mathcal{R}_i p_\Gamma),$$

where $\mathcal{R}_i : V_{i,\Gamma_i}^h \to V_i^h$, is a continuous linear extension operator satisfying $\mathrm{Tr}_{\Gamma_i}(\mathcal{R}_i v_\Gamma) = v_\Gamma$ for all $v_\Gamma \in V_{i,\Gamma_i}^h$. To summarize, the problem (4.12) is understood as follows:

$$\begin{aligned} &\text{Minimize} && \frac{1}{2} m_i(y_i - \hat{y}, y_i - \hat{y}) + \frac{\alpha}{2} q_i(u_i, u_i) \\ &&& + a_i(y_i, \mathcal{R}_i p_\Gamma) + b_i(u_i, \mathcal{R}_i p_\Gamma) - f_i(\mathcal{R}_i p_\Gamma), && \text{(4.13a)} \\ &\text{subject to} && a_i(y_i, \psi) + b_i(u_i, \psi) = f_i(\psi) \quad \forall \psi \in V_{i,0}^h, && \text{(4.13b)} \\ &&& y_i = y_\Gamma \quad \text{on } \Gamma_i. && \text{(4.13c)} \end{aligned}$$

THEOREM 4.4. [illegible] (4.13) [illegible] (4.7) [illegible] (4.13)

[illegible]. (i) By definition of V_{i,Γ_i}^h and Lemma 4.2, the state equations (4.13b), (4.13c) have a unique solution $y_i(u_i) \in V_i^h$ for every $u_i \in U_i^h$. Moreover, $u_i \mapsto y_i(u_i)$ is a continuous affine linear map. Thus, the presence of $\frac{\alpha}{2} q_i(u_i, u_i)$ in the objective ensures that (4.13) is strictly convex and has a unique solution.

(ii) Since (4.13) is a convex linear-quadratic optimal control problem, $y_i \in V_i^h, u_i \in U_i^h$ is a solution of (4.13) if and only if (4.13b), (4.13c) are satisfied and the Fréchet derivative of (4.13a) applied to all vectors in the null-space of the constraints (4.13b), (4.13c) is zero, i.e., if

$$m_i(y_i - \hat{y}, z_i) + q_i(u_i, v_i) + a_i(z_i, \mathcal{R}_i p_\Gamma) + b_i(v_i, \mathcal{R}_i p_\Gamma) = 0 \tag{4.14a}$$

for all $v_i \in U_i^h$ and all $z_i \in V_{i,0}^h$ satisfying the homogeneous linear state equation

$$a_i(z_i, \psi) + b_i(V_i^h, \psi) = 0 \quad \forall \psi \in V_{i,0}^h. \tag{4.14b}$$

Let $y_i \in V_i^h, u_i \in U_i^h$ be a solution of (4.13). Let p_i solve (4.7a) and $p_i = p_\Gamma$ on Γ_i, and define $p_i^0 = p_i - \mathcal{R}_i p_\Gamma \in V_{i,0}^h$. Let $v_i \in U_i^h$ be arbitrary and let $z_i \in V_{i,0}^h$ be the corresponding solution of (4.14b). Using (4.14b) with $\psi = p_i^0$, (4.7a) with $\phi = z_i$, and (4.14a) implies that

$$\begin{aligned} 0 &= m_i(y_i - \hat{y}, z_i) + q_i(u_i, v_i) + a_i(z_i, \mathcal{R}_i p_\Gamma) + b_i(v_i, \mathcal{R}_i p_\Gamma) \\ &\quad + a_i(z_i, p_i^0) + b_i(v_i, p_i^0) \\ &= m_i(y_i - \hat{y}, z_i) + q_i(u_i, v_i) + a_i(z_i, p_i) + b_i(v_i, p_i) \\ &= q_i(u_i, v_i) + b_i(v_i, p_i) \end{aligned}$$

for all $V_i^h \in U_i^h$. Hence, if $y_i \in V_i^h, u_i \in U_i^h$ is a solution of (4.13), then (4.7) is satisfied.

On the other hand, let $y_i, p_i \in V_i^h, U_i^h \in U_i^h$ solve (4.7). Define $p_i^0 = p_i - \mathcal{R}_i p_\Gamma \in V_{i,0}^h$. Let $v_i \in U_i^h, z_i \in V_{i,0}^h$ satisfy the homogeneous linear state equation (4.14b). If we use (4.7a) with $\phi = z_i$, (4.7b) with $\mu = v_i$, and (4.14b) with $\psi = p_i^0$, we obtain

$$\begin{aligned} 0 &= m_i(y_i - \hat{y}, z_i) + q_i(u_i, v_i) + a_i(z_i, p_i) + b_i(v_i, p_i) - a_i(z_i, p_i^0) + b_i(v_i, p_i^0) \\ &= m_i(y_i - \hat{y}, z_i) + q_i(u_i, v_i) + a_i(z_i, \mathcal{R}_i p_\Gamma) + b_i(v_i, \mathcal{R}_i p_\Gamma). \end{aligned}$$

Thus, (4.14a) is satisfied and, therefore, $y_i \in V_i^h, u_i \in U_i^h$ solve (4.13). ∎

4.3. Schur complement formulation. Now we formulate the Schur complement system; i.e, we view the solution of (4.7), (4.8) as an affine linear function of (y_Γ, p_Γ) and then consider (4.9) as a linear equation in (y_Γ, p_Γ).

For $i \in \mathcal{C}$ we define the operator

$$\mathcal{H}_i^h \in \mathcal{L}(V_\Gamma^h \times V_\Gamma^h, V_i^h \times U_i^h \times V_i^h) \tag{4.15a}$$

with

$$\mathcal{H}_i^h(y_\Gamma, p_\Gamma) = \begin{pmatrix} (\mathcal{H}_i^h)^y(y_\Gamma, p_\Gamma) \\ (\mathcal{H}_i^h)^u(y_\Gamma, p_\Gamma) \\ (\mathcal{H}_i^h)^p(y_\Gamma, p_\Gamma) \end{pmatrix} = \begin{pmatrix} y_i^0 \\ u_i^0 \\ p_i^0 \end{pmatrix}, \tag{4.15b}$$

where (y_i^0, u_i^0, p_i^0) is the solution of (4.7) with $f = 0$ and $\hat{y} = 0$. For $i \in \mathcal{N}$ we define the operator

$$\mathcal{H}_i^h \in \mathcal{L}(V_\Gamma^h \times V_\Gamma^h, V_i^h \times V_i^h) \tag{4.16a}$$

with

$$\mathcal{H}_i^h(y_\Gamma, p_\Gamma) = \begin{pmatrix} (\mathcal{H}_i^h)^y(y_\Gamma, p_\Gamma) \\ (\mathcal{H}_i^h)^p(y_\Gamma, p_\Gamma) \end{pmatrix} = \begin{pmatrix} y_i^0 \\ p_i^0 \end{pmatrix}, \tag{4.16b}$$

where (y_i^0, p_i^0) is the solution of (4.8) with $f = 0$ and $\hat{y} = 0$. The linear operators (4.15), (4.16) may be viewed as generalizations of the harmonic extension used in DD methods for linear elliptic PDEs to elliptic linear-quadratic optimal control problems.

THEOREM 4.5. [illegible] $0 < c < C$ [illegible] h [illegible]

$$c\|(y_\Gamma, p_\Gamma)\|_{(V_{i,\Gamma_i}^h)^2} \le \|\mathcal{H}_i^h(y_\Gamma, p_\Gamma)\|_{V_i^h \times U_i^h \times V_i^h} \le C\|(y_\Gamma, p_\Gamma)\|_{(V_{i,\Gamma_i}^h)^2} \tag{4.17}$$

[illegible] $y_\Gamma, p_\Gamma \in (V_{i,\Gamma_i}^h)^2$

[illegible]. We consider the case that Ω_i is a control subdomain, i.e., that (y_i^0, u_i^0, p_i^0) in (4.15b) is the solution of (4.7) with $f = 0$ and $\hat{y} = 0$. The other case can be proved similarly.

(i) Let $y_\Gamma, p_\Gamma \in (V_{i,\Gamma_i}^h)^2$ be arbitrary and let (y_i^0, u_i^0, p_i^0) be the corresponding solution of (4.7) with $f = 0$ and $\hat{y} = 0$. By the trace theorem, there exist constants $c_1, c_2 > 0$, independent of $y_\Gamma, p_\Gamma \in (V_{i,\Gamma_i}^h)^2$, such that

$$\|y_\Gamma\|_{V_{i,\Gamma_i}^h} \le c_1\|y_i^0\|_{V_i^h}, \quad \|p_\Gamma\|_{V_{i,\Gamma_i}^h} \le c_2\|p_i^0\|_{V_i^h},$$

where (y_i^0, u_i^0, p_i^0) is the solution of (4.7) with $f = 0$ and $\hat{y} = 0$. This implies the left-hand inequality in (4.17).

(ii) Let $\mathcal{R}_i : V_{i,\Gamma_i}^h \to V_i^h$ be a continuous linear extension operator satisfying $\mathrm{Tr}_{\Gamma_i}(\mathcal{R}_i v_\Gamma) = v_\Gamma$ for all $v_\Gamma \in V_{i,\Gamma_i}^h$.

The y, p component of the solution (y_i^0, u_i^0, p_i^0) of (4.7) with $f = 0$ and $\hat{y} = 0$ can be written as $y_i^0 = \mathcal{R}_i y_\Gamma + \tilde{y}_i$, $p_i^0 = \mathcal{R}_i p_\Gamma + \tilde{p}_i$, where $\tilde{y}_i, \tilde{p}_i \in V_{i,0}^h$, and $(\tilde{y}_i, u_i^0, \tilde{p}_i) \in V_{i,0}^h \times U_i^h \times V_{i,0}^h$ solve

$$a_i(\psi, \tilde{p}_i) + m_i(\tilde{y}_i, \psi) = -a_i(\psi, \mathcal{R}_i p_\Gamma) - m_i(\mathcal{R}_i y_\Gamma, \psi) \qquad \forall \psi \in V_{i,0}^h, \tag{4.18a}$$

$$b_i(\mu, \tilde{p}_i) + q_i(u_i^0, \mu) = -b_i(\mu, \mathcal{R}_i p_\Gamma) \qquad \forall \mu \in U_i^h, \tag{4.18b}$$

$$a_i(\tilde{y}_i, \phi) + b_i(u_i^0, \phi) = -a_i(\mathcal{R}_i y_\Gamma, \phi) \qquad \forall \phi \in V_{i,0}^h. \tag{4.18c}$$

The properties of the bilinear forms a_i, b_i, equation (4.18c), and the continuity of $\mathcal{R}_i$ imply the existence of $c_3, c_4 > 0$ independent of $\tilde{y}_i, u_i^0, y_\Gamma$ such that

$$\begin{aligned} \|\tilde{y}_i\|_{V_i^h} &\le c_3\Big(\|u_i^0\|_{U_i^h} + \|\mathcal{R}_i y_\Gamma\|_{V_i^h}\Big) \\ &\le c_4\Big(\|u_i^0\|_{U_i^h} + \|y_\Gamma\|_{V_{i,\Gamma_i}^h}\Big). \end{aligned} \tag{4.19}$$

Similarly, properties of the bilinear forms a_i, m_i, equation (4.18a), the continuity of $\mathcal{R}_i$, and (4.19) imply the existence of $c_5, c_6 > 0$ independent of $\tilde{y}_i, u_i^0, y_\Gamma, \tilde{p}_i, p_\Gamma$ such that

$$\begin{aligned} \|\tilde{p}_i\|_{V_i^h} &\le c_5\Big(\|\tilde{y}_i\|_{V_i^h} + \|\mathcal{R}_i y_\Gamma\|_{V_i^h} + \|\mathcal{R}_i p_\Gamma\|_{V_i^h}\Big) \\ &\le c_6\Big(\|u_i^0\|_{U_i^h} + \|y_\Gamma\|_{V_{i,\Gamma_i}^h} + \|p_\Gamma\|_{V_{i,\Gamma_i}^h}\Big). \end{aligned} \tag{4.20}$$

If we set $\psi = \tilde{y}_i$, $\mu = u_i^0$, $\phi = -\tilde{p}_i$ in (4.18) and add the equations, we obtain

$$m_i(\tilde{y}_i, \tilde{y}_i) + q_i(u_i^0, u_i^0) = -a_i(\tilde{y}_i, \mathcal{R}_i p_\Gamma) - m_i(\mathcal{R}_i y_\Gamma, \tilde{y}_i) - b_i(u^0, \mathcal{R}_i p_\Gamma) + a_i(\mathcal{R}_i y_\Gamma, \tilde{p}_i).$$

The properties of the bilinear forms a_i, b_i, q_i, m_i, the continuity of $\mathcal{R}_i$, and the inequalities (4.19), (4.20) guarantee the existence of $c_7, c_8 > 0$ independent of $\tilde{y}_i, u_i^0, y_\Gamma, p_\Gamma$ such that

$$\begin{aligned} \alpha\|u_i^0\|_{U_i^h}^2 &\le m_i(\tilde{y}_i, \tilde{y}_i) + q_i(u_i^0, u_i^0) \\ &\le c_7\Big(\|\tilde{y}_i\|_{V_i^h}\|p_\Gamma\|_{V_{i,\Gamma_i}^h} + \|\tilde{y}_i\|_{V_i^h}\|y_\Gamma\|_{V_{i,\Gamma_i}^h} + \|u_i^0\|_{U_i^h}\|p_\Gamma\|_{V_{i,\Gamma_i}^h} \\ &\qquad + \|\tilde{p}_i\|_{V_i^h}\|y_\Gamma\|_{V_{i,\Gamma_i}^h}\Big) \\ &\le c_8\Big(\|y_\Gamma\|_{V_{i,\Gamma_i}^h} + \|p_\Gamma\|_{V_{i,\Gamma_i}^h}\Big)\|u_i^0\|_{U_i^h} + c_8\Big(\|y_\Gamma\|_{V_{i,\Gamma_i}^h} + \|p_\Gamma\|_{V_{i,\Gamma_i}^h}\Big)^2 \\ &\le \frac{\alpha}{2}\|u_i^0\|_{U_i^h}^2 + \frac{c_8^2}{2\alpha}\Big(\|y_\Gamma\|_{V_{i,\Gamma_i}^h} + \|p_\Gamma\|_{V_{i,\Gamma_i}^h}\Big)^2 + c_8\Big(\|y_\Gamma\|_{V_{i,\Gamma_i}^h} + \|p_\Gamma\|_{V_{i,\Gamma_i}^h}\Big)^2. \end{aligned}$$

Hence,

$$\|u_i^0\|_{U_i^h} \le c_9\Big(\|y_\Gamma\|_{V_{i,\Gamma_i}^h} + \|p_\Gamma\|_{V_{i,\Gamma_i}^h}\Big), \tag{4.21}$$

where $c_9 = \sqrt{2c_8/\alpha + c_8^2/\alpha^2}$. The right-hand inequality in (4.17) now follows from (4.19), (4.20), (4.21). ∎

Let $\langle\!\langle \cdot, \cdot \rangle\!\rangle$ denote the duality pairing between $(V_\Gamma^h)^2$ and $((V_\Gamma^h)^*)^2$. We define the subdomain Schur complement operator

$$\mathcal{S}_i^h \in \mathcal{L}((V_\Gamma^h)^2, ((V_\Gamma^h)^*)^2), \tag{4.22a}$$

$i = 1, \ldots, s$, with

$$\begin{aligned} &\langle\!\langle \mathcal{S}_i^h(y_\Gamma, p_\Gamma), (v_\Gamma, q_\Gamma) \rangle\!\rangle \\ &= a_i((\mathcal{H}_i^h)^y(y_\Gamma, p_\Gamma), (\mathcal{H}_i^h)^p(v_\Gamma, q_\Gamma)) + b_i((\mathcal{H}_i^h)^u(y_\Gamma, p_\Gamma), (\mathcal{H}_i^h)^p(v_\Gamma, q_\Gamma)) \\ &\quad + a_i((\mathcal{H}_i^h)^y(v_\Gamma, q_\Gamma), (\mathcal{H}_i^h)^p(y_\Gamma, p_\Gamma)) + m_i((\mathcal{H}_i^h)^y(y_\Gamma, p_\Gamma), (\mathcal{H}_i^h)^y(v_\Gamma, q_\Gamma)) \end{aligned} \tag{4.22b}$$

for $i \in \mathcal{C}$, and

$$\begin{aligned}
&\langle\!\langle \mathcal{S}_i^h(y_\Gamma, p_\Gamma), (v_\Gamma, q_\Gamma) \rangle\!\rangle \\
&= a_i((\mathcal{H}_i^h)^y(y_\Gamma, p_\Gamma), (\mathcal{H}_i^h)^p(v_\Gamma, q_\Gamma)) \\
&\quad + a_i((\mathcal{H}_i^h)^y(v_\Gamma, q_\Gamma), (\mathcal{H}_i^h)^p(y_\Gamma, p_\Gamma)) + m_i((\mathcal{H}_i^h)^y(y_\Gamma, p_\Gamma), (\mathcal{H}_i^h)^y(v_\Gamma, q_\Gamma))
\end{aligned} \tag{4.22c}$$

for $i \in \mathcal{N}$. The definition of $\mathcal{H}_i^h$ implies that for $i \in \mathcal{C}$,

$$\begin{aligned}
&\langle\!\langle \mathcal{S}_i^h(y_\Gamma, p_\Gamma), (v_\Gamma, q_\Gamma) \rangle\!\rangle \\
&= a_i((\mathcal{H}_i^h)^y(y_\Gamma, p_\Gamma), (\mathcal{H}_i^h)^p(v_\Gamma, q_\Gamma)) + b_i((\mathcal{H}_i^h)^u(y_\Gamma, p_\Gamma), (\mathcal{H}_i^h)^p(v_\Gamma, q_\Gamma)) \\
&\quad + b_i((\mathcal{H}_i^h)^u(v_\Gamma, q_\Gamma), (\mathcal{H}_i^h)^p(y_\Gamma, p_\Gamma)) + q_i((\mathcal{H}_i^h)^u(y_\Gamma, p_\Gamma), (\mathcal{H}_i^h)^u(v_\Gamma, q_\Gamma)) \\
&\quad + a_i((\mathcal{H}_i^h)^y(v_\Gamma, q_\Gamma), (\mathcal{H}_i^h)^p(y_\Gamma, p_\Gamma)) + m_i((\mathcal{H}_i^h)^y(y_\Gamma, p_\Gamma), (\mathcal{H}_i^h)^y(v_\Gamma, q_\Gamma)).
\end{aligned} \tag{4.22d}$$

We define $r_i \in ((V_\Gamma^h)^*)^2$, $i = 1, \ldots, s$, as follows. If $i \in \mathcal{C}$, then

$$\begin{aligned}
\langle\!\langle r_i, (v_\Gamma, q_\Gamma) \rangle\!\rangle &= f_i((\mathcal{H}_i^h)^p(v_\Gamma, q_\Gamma)) + m_i(\hat{y}_i, (\mathcal{H}_i^h)^y(v_\Gamma, q_\Gamma)) \\
&\quad - a_i(y_i, (\mathcal{H}_i^h)^p(v_\Gamma, q_\Gamma)) - b_i(u_i, (\mathcal{H}_i^h)^p(v_\Gamma, q_\Gamma)) \\
&\quad - a_i((\mathcal{H}_i^h)^y(v_\Gamma, q_\Gamma), p_i) - m_i(y_i, (\mathcal{H}_i^h)^y(v_\Gamma, q_\Gamma)),
\end{aligned} \tag{4.23}$$

where (y_i, u_i, p_i) is the solution of (4.7) with $y_\Gamma = 0$ and $p_\Gamma = 0$. If $i \in \mathcal{N}$, then

$$\begin{aligned}
\langle\!\langle r_i, (v_\Gamma, q_\Gamma) \rangle\!\rangle &= f_i((\mathcal{H}_i^h)^p(v_\Gamma, q_\Gamma)) + m_i(\hat{y}_i, (\mathcal{H}_i^h)^y(v_\Gamma, q_\Gamma)) - a_i(y_i, (\mathcal{H}_i^h)^p(v_\Gamma, q_\Gamma)) \\
&\quad - a_i((\mathcal{H}_i^h)^y(v_\Gamma, q_\Gamma), p_i) - m_i(y_i, (\mathcal{H}_i^h)^y(v_\Gamma, q_\Gamma)),
\end{aligned} \tag{4.24}$$

where (y_i, p_i) is the solution of (4.8) with $y_\Gamma = 0$ and $p_\Gamma = 0$.

Theorem 4.1 implies that the system (4.6) of optimality conditions is equivalent to the Schur complement system

$$\sum_{i=1}^{s} \mathcal{S}_i^h(y_\Gamma, p_\Gamma) = \sum_{i=1}^{s} r_i \quad \text{in } ((V_\Gamma^h)^*)^2. \tag{4.25}$$

From (4.22c) and (4.22d) it is easy to see that the subdomain Schur complement operator is symmetric; i.e.,

$$\langle\!\langle \mathcal{S}_i^h(y_\Gamma, p_\Gamma), (v_\Gamma, q_\Gamma) \rangle\!\rangle = \langle\!\langle \mathcal{S}_i^h(v_\Gamma, q_\Gamma), (y_\Gamma, p_\Gamma) \rangle\!\rangle \quad \forall y_\Gamma, v_\Gamma, p_\Gamma, q_\Gamma \in V_{i,\Gamma_i}^h.$$

However, unlike subdomain Schur complement operators arising in DD methods for elliptic PDEs, the subdomain Schur complement operator $\mathcal{S}_i^h$ is not positive definite. The inertia will be investigated in Theorem 5.5 and we will show that the Schur complement operator essentially has the same number of positive and negative eigenvalues.

Remark 4.6. If Ω_i is a control subdomain, Theorem 4.4 states that the evaluation of $\mathcal{S}_i(y_\Gamma, p_\Gamma)$ requires the solution of a linear-quadratic elliptic optimal control problem (4.13). If Ω_i is a no-control subdomain, the evaluation of $\mathcal{S}_i(y_\Gamma, p_\Gamma)$ requires the solution of a triangular system of elliptic PDEs (4.8).

We will solve (4.25) using a preconditioned Krylov subspace method. Our preconditioner is constructed using the inverses of the subdomain Schur complement operators $\mathcal{S}_i^h(y_\Gamma, p_\Gamma)$, $i = 1, \ldots, s$. Their existence and computation will be studied next.

4.4. The inverse subdomain Schur complement. The next result establishes the invertibility of the subdomain Schur complement operator $\mathcal{S}_i^h$ in the case that $\sigma > 0$.

THEOREM 4.7. [illegible] $\sigma > 0$ [illegible] $r_i = (r_i^y, r_i^p) \in ((V_{i,\Gamma_i}^h)^*)^2$

(i) [illegible] Ω_i [illegible] $(y_\Gamma, p_\Gamma) \in (V_{i,\Gamma_i}^h)^2$ [illegible]

$$\mathcal{S}_i^h(y_\Gamma, p_\Gamma) = r_i \quad \text{[illegible]} \quad ((V_{i,\Gamma_i}^h)^*)^2 \tag{4.26}$$

[illegible]

$$(y_\Gamma, p_\Gamma) = (\ \cdot_{\Gamma_i} y_i, \ \cdot_{\Gamma_i} p_i),$$

[illegible] $(y_i, u_i, p_i) \in V_i^h \times U_i^h \times V_i^h$ [illegible]

$$a_i(\psi, p_i) + m_i(y_i, \psi) = \langle r_i^y, \ \cdot_{\Gamma_i}\psi\rangle \qquad \forall \psi \in V_i^h, \tag{4.27a}$$

$$b_i(\mu, p_i) + q_i(u_i, \mu) = 0 \qquad \forall \mu \in U_i^h, \tag{4.27b}$$

$$a_i(y_i, \psi) + b_i(u_i, \psi) = \langle r_i^p, \ \cdot_{\Gamma_i}\psi\rangle \qquad \forall \psi \in V_i^h, \tag{4.27c}$$

[illegible] $\langle\cdot,\cdot\rangle$ [illegible] V_{i,Γ_i}^h [illegible] $(V_{i,\Gamma_i}^h)^*$

(ii) [illegible] Ω_i [illegible] $(y_\Gamma, p_\Gamma) \in (V_{i,\Gamma_i}^h)^2$ [illegible] (4.26) [illegible]

$$(y_\Gamma, p_\Gamma) = (\ \cdot_{\Gamma_i} y_i, \ \cdot_{\Gamma_i} p_i),$$

[illegible] $(y_i, u_i, p_i) \in V_i^h \times U_i^h \times V_i^h$ [illegible]

$$a_i(\psi, p_i) + m_i(y_i, \psi) = \langle r_i^y, \ \cdot_{\Gamma_i}\psi\rangle \qquad \forall \psi \in V_i^h, \tag{4.28a}$$

$$a_i(y_i, \psi) = \langle r_i^p, \ \cdot_{\Gamma_i}\psi\rangle \qquad \forall \psi \in V_i^h. \tag{4.28b}$$

[illegible] (i) By definition (4.22) of $\mathcal{S}_i^h$, the equality (4.26) can be written as

$$\begin{aligned} &a_i((\mathcal{H}_i^h)^y(y_\Gamma, p_\Gamma), (\mathcal{H}_i^h)^p(v_\Gamma, q_\Gamma)) + b_i((\mathcal{H}_i^h)^u(y_\Gamma, p_\Gamma), (\mathcal{H}_i^h)^p(v_\Gamma, q_\Gamma)) \\ &\quad + a_i((\mathcal{H}_i^h)^y(v_\Gamma, q_\Gamma), (\mathcal{H}_i^h)^p(y_\Gamma, p_\Gamma)) + m_i((\mathcal{H}_i^h)^y(y_\Gamma, p_\Gamma), (\mathcal{H}_i^h)^y(v_\Gamma, q_\Gamma)) \\ &= \langle r_i^y, v_\Gamma\rangle + \langle r_i^p, q_\Gamma\rangle \end{aligned} \tag{4.29}$$

for all $v_\Gamma, q_\Gamma \in V_{i,\Gamma_i}^h$. Using the definition (4.15) of $\mathcal{H}_i^h(y_\Gamma, p_\Gamma)$ together with (4.29), we see that (4.26) is equivalent to

$$\begin{aligned} &a_i(y_i, (\mathcal{H}_i^h)^p(v_\Gamma, q_\Gamma)) \\ &+ b_i(u_i, (\mathcal{H}_i^h)^p(v_\Gamma, q_\Gamma)) \\ &+ a_i((\mathcal{H}_i^h)^y(v_\Gamma, q_\Gamma), p_i) \\ &+ m_i(y_i, (\mathcal{H}_i^h)^y(v_\Gamma, q_\Gamma)) = \langle r_i^y, V_\Gamma^h\rangle + \langle r_i^p, q_\Gamma\rangle \qquad \forall v_\Gamma, q_\Gamma \in V_{i,\Gamma_i}^h, \end{aligned} \tag{4.30a}$$

$$a_i(\psi^0, p_i) + m_i(y_i, \psi^0) = 0 \qquad \forall \psi^0 \in V_{i,0}^h, \tag{4.30b}$$

$$b_i(\mu, p_i) + q_i(u_i, \mu) = 0 \qquad \forall \mu \in U_i^h, \tag{4.30c}$$

$$a_i(y_i, \phi^0) + b_i(u_i, \phi^0) = 0 \qquad \forall \phi^0 \in V_{i,0}^h, \tag{4.30d}$$

$$y_i = y_\Gamma, \quad p_i = p_\Gamma \qquad \text{on } \Gamma. \tag{4.30e}$$

If we set $\psi = \psi^0 + (\mathcal{H}_i^h)^p(v_\Gamma, q_\Gamma) \in V_i^h$ and $\phi = \phi^0 + (\mathcal{H}_i^h)^y(v_\Gamma, q_\Gamma) \in V_i^h$, then (4.30) is equivalent to

$$a_i(\psi, p_i) + m_i(y_i, \psi) = \langle r_i^y, \mathrm{Tr}_{\Gamma_i}\psi\rangle \qquad \forall \psi \in V_i^h, \tag{4.31a}$$

$$b_i(\mu, p_i) + q_i(u_i, \mu) = 0 \qquad \forall \mu \in U_i^h, \tag{4.31b}$$

$$a_i(y_i, \phi) + b_i(u_i, \phi) = \langle r_i^p, \mathrm{Tr}_{\Gamma_i}\phi\rangle \qquad \forall \phi \in V_i^h, \tag{4.31c}$$

$$y_i = y_\Gamma, \quad p_i = p_\Gamma \qquad \text{on } \Gamma. \tag{4.31d}$$

The assertion follows if we prove that (4.27) has a unique solution $(y_i, u_i, p_i) \in V_i^h \times U_i^h \times V_i^h$. Let $(y_i^1, u_i^1, p_i^1), (y_i^2, u_i^2, p_i^2) \in V_i^h \times U_i^h \times V_i^h$ be solutions of (4.27). Then $(e_i^y, e_i^u, e_i^p) = (y_i^1 - y_i^2, u_i^1 - u_i^2, p_i^1 - p_i^2) \in V_i^h \times U_i^h \times V_i^h$ satisfies

$$a_i(\psi, e_i^p) + m_i(e_i^y, \psi) = 0 \qquad \forall \psi \in V_i^h, \tag{4.32a}$$

$$b_i(\mu, e_i^p) + q_i(e_i^u, \mu) = 0 \qquad \forall \mu \in U_i^h, \tag{4.32b}$$

$$a_i(e_i^y, \phi) + b_i(e_i^u, \phi) = 0 \qquad \forall \phi \in V_i^h. \tag{4.32c}$$

If we set $\psi = e_i^y$, $\mu = e_i^u$, and $\phi = -e_i^p$ in (4.32) and add the resulting equations, we obtain

$$0 = m_i(e_i^y, e_i^y) + q_i(e_i^u, e_i^u) \geq \alpha \|e_i^u\|^2_{U_i^h}.$$

Hence $e_i^u = 0$. Equation (4.32c) implies $e_i^y = 0$ and, finally, (4.32a) implies $e_i^p = 0$.

(ii) The second statement can be proved analogously. ∎

REMARK 4.8. Equations (4.27) can be interpreted as the weak form of

$$-\Delta y_i(x) + \sigma y_i(x) = 0 \qquad \text{in } \Omega_i, \tag{4.33a}$$

$$\frac{\partial}{\partial n} y_i(x) = U_i^h(x) \qquad \text{on } \partial\Omega \cap \partial\Omega_i, \tag{4.33b}$$

$$\frac{\partial}{\partial n} y_i(x) = r_i^y(x) \qquad \text{on } \Gamma, \tag{4.33c}$$

$$-\Delta p_i(x) + \sigma p_i(x) = -y_i(x) \qquad \text{in } \Omega_i, \tag{4.33d}$$

$$\frac{\partial}{\partial n} p_i(x) = 0 \qquad \text{on } \partial\Omega \cap \partial\Omega_i, \tag{4.33e}$$

$$\frac{\partial}{\partial n} p_i(x) = r_i^p(x) \qquad \text{on } \Gamma, \tag{4.33f}$$

$$\alpha u_i(x) - p_i(x) = 0 \qquad \text{on } \partial\Omega \cap \partial\Omega_i. \tag{4.33g}$$

REMARK 4.9. If $\sigma = 0$, then the subdomain Schur complement operator $\mathcal{S}_i$ may no longer be invertible. In this case we define

$$\widehat{a}_i : V_i^h \times V_i^h \to \mathbb{R}, \qquad \widehat{a}_i(y, \psi) = \int_{\Omega_i} \nabla y(x) \nabla \psi(x) + \varepsilon y(x)\psi(x) dx,$$

where $\varepsilon > 0$ and we introduce the perturbed subdomain Schur complement operator $\widehat{\mathcal{S}}_i$, which is defined as in (4.22) but with a_i replaced by $\widehat{a}_i$. Theorem 4.7 implies that $\widehat{\mathcal{S}}_i$ is continuously invertible. A situation somewhat comparable to the case $\sigma = 0$ arises in so-called Neumann–Neumann DD methods for the Laplace equation. Following, e.g., [36, p. 175], we use $\widehat{a}_i$ with $\varepsilon = 1/H_i^2$, where H_i is the diameter of Ω_i.

5. Algebraic formulation. In this section we present the matrix view of the DD Schur complement formulation introduced in the previous section and introduce the Neumann–Neumann preconditioners for the Schur complement. Furthermore, we study some properties of the Schur complement matrix arising in the optimal control context and discuss the numerical solution of the preconditioned Schur complement system.

5.1. Matrix representation of the discretized optimal control problem. A finite element discretization of the optimal control problem (3.1) leads to a large-scale linear-quadratic problem of the form

$$\text{minimize } \frac{1}{2}\mathbf{y}^T\mathbf{M}\mathbf{y} + \mathbf{c}^T\mathbf{y} + \mathbf{y}^T\mathbf{N}\mathbf{u} + \frac{\alpha}{2}\mathbf{u}^T\mathbf{Q}\mathbf{u} + \mathbf{d}^T\mathbf{u}, \tag{5.1a}$$

$$\text{subject to } \mathbf{A}\mathbf{y} + \mathbf{B}\mathbf{u} + \mathbf{b} = \mathbf{0}, \tag{5.1b}$$

with $\mathbf{A} \in \mathbb{R}^{m\times m}$, $\mathbf{B} \in \mathbb{R}^{m\times n}$, $\mathbf{N} \in \mathbb{R}^{m\times n}$, $\mathbf{M} \in \mathbb{R}^{m\times m}$, $\mathbf{Q} \in \mathbb{R}^{n\times n}$, $\mathbf{b} \in \mathbb{R}^m$, $\mathbf{c} \in \mathbb{R}^m$, $\mathbf{d} \in \mathbb{R}^n$, $\mathbf{y} \in \mathbb{R}^m$, $\mathbf{u} \in \mathbb{R}^n$, and $\alpha > 0$.

For our model problem the matrices and vectors $\mathbf{A}, \dots$ are given as follows. Let $\{x_j\}_{j=1}^{n_v}$ be the set of vertices of $\{T_l\}$ and let $\{\phi_j\}_{j=1}^{n_v}$ be the piecewise linear nodal basis for V^h defined in (4.1). Furthermore, let $\{\mu_j^i\}_{j=1}^{n_{bv}^i}$ be the piecewise linear nodal basis for U_i^h defined in (4.4), where n_{bv}^i is the number of vertices on $\partial\Omega \cap \partial\Omega_i$. We identify μ_j^i with a function in $L^2(\partial\Omega)$ by extending μ_j^i by zero outside $\partial\Omega_i$. We set

$$\mu_1 = \mu_1^1, \dots, \mu_{n_{bv}^i} = \mu_{n_{bv}^1}^1,\ \mu_{n_{bv}^1+1} = \mu_1^2, \dots, \mu_{n_{bv}^1+n_{bv}^2} = \mu_{n_{bv}^2}^2, \dots.$$

Then $m = n_v$, $n = \sum_{i=1}^s n_{bv}^i$, and

$$\mathbf{A}_{jk} = a(\phi_k, \phi_j), \quad \mathbf{M}_{jk} = m(\phi_k, \phi_j), \quad \mathbf{c}_j = -m(\hat{y}, \phi_j), \quad \mathbf{b}_j = f(\phi_j)$$

for $j, k = 1, \dots, m$, and

$$\mathbf{B}_{jk} = b(\mu_k, \phi_j), \quad \mathbf{Q}_{jk} = q(\mu_k, \mu_j)$$

for $j, k = 1, \dots, n$, and $\mathbf{N} = \mathbf{0}$, $\mathbf{d} = \mathbf{0}$.

LEMMA 5.1. [illegible] $\sigma \geq 0$ [illegible]

$$\text{range}(\mathbf{A} \mid \mathbf{B}) = \mathbb{R}^m \tag{5.2}$$

[illegible] $\mathbf{M}, \mathbf{Q}$ [illegible] (5.1a) [illegible] [illegible] (5.1b).

$$\begin{pmatrix} \mathbf{z} \\ \mathbf{v} \end{pmatrix}^T \begin{pmatrix} \mathbf{M} & \mathbf{N} \\ \mathbf{N}^T & \alpha\mathbf{Q} \end{pmatrix} \begin{pmatrix} \mathbf{z} \\ \mathbf{v} \end{pmatrix} > 0 \quad \forall\, \mathbf{z} \in \mathbb{R}^m, \mathbf{v} \in \mathbb{R}^n \text{ [illegible] } \mathbf{A}\mathbf{z} + \mathbf{B}\mathbf{v} = 0. \tag{5.3}$$

[illegible]. If $\sigma > 0$, then $\mathbf{A}$ is symmetric positive definite and (5.2) follows immediately. It holds that

$$\text{null}(\mathbf{A} \mid \mathbf{B}) = \left\{ \begin{pmatrix} -\mathbf{A}^{-1}\mathbf{B}\mathbf{v} \\ \mathbf{v} \end{pmatrix} \,:\, \mathbf{v} \in \mathbb{R}^n \right\}$$

and (5.3) is equivalent to

$$\mathbf{v}^T(\alpha\mathbf{Q} + \mathbf{B}^T\mathbf{A}^{-T}\mathbf{M}\mathbf{A}^{-1}\mathbf{B})\mathbf{v} > 0 \quad \forall\, \mathbf{v} \in \mathbb{R}^n, \tag{5.4}$$

where we have used the fact that in the model problem $\mathbf{N} = \mathbf{0}$. Condition (5.4) follows, since in our model problem $\mathbf{M}$ is symmetric positive semidefinite and $\mathbf{Q}$ is symmetric positive definite.

If $\sigma = 0$, then $\mathbf{A}$ is symmetric positive semidefinite and $\text{null}(\mathbf{A}) = \text{span}\{\mathbf{e}\}$, where $\mathbf{e} \in \mathbb{R}^m$ is the vector of all ones. If $(\mathbf{z}^T, \mathbf{v}^T)^T \in \text{null}(\mathbf{A} \mid \mathbf{B})$, then $0 = \mathbf{e}^T(\mathbf{A}\mathbf{z} + \mathbf{B}\mathbf{v}) = \mathbf{e}^T\mathbf{B}\mathbf{v}$. Hence, $\mathbf{v} \in \text{span}\{\mathbf{B}^T\mathbf{e}\}^\perp$. Moreover, for each $\mathbf{v} \in \text{span}\{\mathbf{B}^T\mathbf{e}\}^\perp$, the equation $\mathbf{A}\mathbf{z} = -\mathbf{B}\mathbf{v}$ has a solution and the minimum norm solution of this equation is characterized by $\mathbf{e}^T\mathbf{z} = 0$. This implies that

$$\text{null}(\mathbf{A} \mid \mathbf{B}) = \text{span}\left\{\begin{pmatrix}\mathbf{e}\\ \mathbf{0}\end{pmatrix}\right\} + \left\{\begin{pmatrix}-\mathbf{A}^\dagger\mathbf{B}\mathbf{v}\\ \mathbf{v}\end{pmatrix} \;:\; \mathbf{v} \in \text{span}\{\mathbf{B}^T\mathbf{e}\}^\perp\right\},$$

where $\mathbf{A}^\dagger$ denotes the pseudoinverse of $\mathbf{A}$, and $\dim(\text{null}(\mathbf{A} \mid \mathbf{B})) = n$. Consequently, $\dim(\text{range}(\mathbf{A} \mid \mathbf{B})) = m$, which implies (5.2). Condition (5.3) follows, since in our model problem $\mathbf{M}$ is symmetric positive semidefinite with $\mathbf{e}^T\mathbf{M}\mathbf{e} = \int_{\Omega_o} 1 dx > 0$, $\mathbf{Q}$ is symmetric positive definite, and $\mathbf{N} = \mathbf{0}$. □

The following result is known, and it is stated here for completeness. A proof can be found in [17].

THEOREM 5.2. (i) [illegible] (5.2) [illegible] (5.3) [illegible] (5.1) [illegible]

$$\begin{pmatrix}\mathbf{M} & \mathbf{N} & \mathbf{A}^T\\ \mathbf{N}^T & \alpha\mathbf{Q} & \mathbf{B}^T\\ \mathbf{A} & \mathbf{B} & \mathbf{0}\end{pmatrix}\begin{pmatrix}\mathbf{y}\\ \mathbf{u}\\ \mathbf{p}\end{pmatrix} = -\begin{pmatrix}\mathbf{c}\\ \mathbf{d}\\ \mathbf{b}\end{pmatrix}. \tag{5.5}$$

(ii) [illegible] (5.2) [illegible] (5.3) [illegible] (5.5) [illegible] $m + n$ [illegible] m [illegible]

The equation (5.5) is the matrix representation of the discretized optimality conditions (4.6).

5.2. Domain decomposition Schur complement formulation. We can use the decomposition of Ω to decompose the matrices $\mathbf{A}$, etc. Let the local (bi)linear forms $a_i, \ldots$ be defined as in section 4.

Let $\tilde{\mathbf{R}}_i^y$, $i = 1, \ldots, s$, be the restriction operator which maps from the vector of coefficient unknowns on the artificial boundary, $\mathbf{y}_\Gamma$, to only those associated with the boundary of Ω_i, and define

$$\tilde{\mathbf{R}}_i = \begin{pmatrix}\tilde{\mathbf{R}}_i^y & \\ & \tilde{\mathbf{R}}_i^p\end{pmatrix}, \quad \tilde{\mathbf{R}}_i^p = \tilde{\mathbf{R}}_i^y. \tag{5.6}$$

For $i = 1, \ldots, s$, we define the submatrices

$$\begin{aligned}(\mathbf{A}_{II}^i)_{jk} &= a_i(\phi_k, \phi_j), \quad x_j, x_k \in \Omega_i,\\ (\mathbf{A}_{I\Gamma}^i)_{jk} &= a_i(\phi_k, \phi_j), \quad x_j \in \Omega_i,\ x_k \in \partial\Omega_i \setminus \partial\Omega,\\ (\mathbf{A}_{\Gamma\Gamma}^i)_{jk} &= a_i(\phi_k, \phi_j), \quad x_j, x_k \in \partial\Omega_i \setminus \partial\Omega,\end{aligned}$$

$\mathbf{A}_{\Gamma I}^i = (\mathbf{A}_{I\Gamma}^i)^T$, and we define $\mathbf{A}_{\Gamma\Gamma} = \sum_{i=1}^s (\tilde{\mathbf{R}}_i^y)^T \mathbf{A}_{\Gamma\Gamma}^i \tilde{\mathbf{R}}_i^y$. After a suitable reordering of rows and columns, the stiffness matrix can be written as

$$\mathbf{A} = \begin{pmatrix}\mathbf{A}_{II}^1 & & & \mathbf{A}_{I\Gamma}^1\tilde{\mathbf{R}}_1^y\\ & \ddots & & \vdots\\ & & \mathbf{A}_{II}^s & \mathbf{A}_{I\Gamma}^s\tilde{\mathbf{R}}_s^y\\ (\tilde{\mathbf{R}}_1^y)^T\mathbf{A}_{\Gamma I}^1 & \cdots & (\tilde{\mathbf{R}}_s^y)^T\mathbf{A}_{\Gamma I}^s & \mathbf{A}_{\Gamma\Gamma}\end{pmatrix}.$$

Similar decompositions can be introduced for $\mathbf{M}$ and $\mathbf{c}$, as well as $\mathbf{y}, \mathbf{p}$.

For $i = 1, \ldots, s$, we define the submatrices

$$(\mathbf{B}_{II}^i)_{jk} = b_i(\mu_k, \phi_j), \quad x_j \in \Omega_i, x_k \in \partial\Omega_i \cap \partial\Omega,$$
$$(\mathbf{B}_{\Gamma I}^i)_{jk} = b_i(\mu_k, \phi_j), \quad x_j \in \partial\Omega_i \setminus \partial\Omega, x_k \in \partial\Omega_i \cap \partial\Omega.$$

After a suitable reordering of rows and columns, the matrix $\mathbf{B}$ can be written as

$$\mathbf{B} = \begin{pmatrix} \mathbf{B}_{II}^1 & & \\ & \ddots & \\ & & \mathbf{B}_{II}^s \\ (\tilde{\mathbf{R}}_1^y)^T \mathbf{B}_{\Gamma I}^1 & \cdots & (\tilde{\mathbf{R}}_s^y)^T \mathbf{B}_{\Gamma I}^s \end{pmatrix}.$$

Note that in our particular control discretization, all basis functions μ_k^i for the discretized control u_h have support in only one subdomain boundary $\partial\Omega_i$ (see the right-hand plot in Figure 4.1). Consequently, there is no $\mathbf{B}_{\Gamma\Gamma}^i$. The matrix $\mathbf{Q}$ and the vector $\mathbf{u}$ can be decomposed analogously. Note that there is no $\mathbf{u}_\Gamma$.

We can now insert the DD structure of the matrices $\mathbf{A}, \mathbf{M}, \mathbf{B}, \mathbf{Q}$ into (5.5). After a symmetric permutation, (5.5) can be written as

$$\begin{pmatrix} \mathbf{K}_{II}^1 & & & (\mathbf{K}_{\Gamma I}^1)^T \tilde{\mathbf{R}}_1 \\ & \ddots & & \vdots \\ & & \mathbf{K}_{II}^s & (\mathbf{K}_{\Gamma I}^s)^T \tilde{\mathbf{R}}_s \\ \tilde{\mathbf{R}}_1^T \mathbf{K}_{\Gamma I}^1 & \cdots & \tilde{\mathbf{R}}_s^T \mathbf{K}_{\Gamma I}^s & \mathbf{K}_{\Gamma\Gamma} \end{pmatrix} \begin{pmatrix} \mathbf{x}_I^1 \\ \vdots \\ \mathbf{x}_I^d \\ \mathbf{x}_\Gamma \end{pmatrix} = \begin{pmatrix} \mathbf{g}_I^1 \\ \vdots \\ \mathbf{g}_I^d \\ \mathbf{g}_\Gamma \end{pmatrix}, \tag{5.7}$$

where

$$\mathbf{K}_{\Gamma\Gamma}^i = \begin{pmatrix} \mathbf{M}_{\Gamma\Gamma}^i & (\mathbf{A}_{\Gamma\Gamma}^i)^T \\ \mathbf{A}_{\Gamma\Gamma}^i & \end{pmatrix}, \ i = 1, \ldots, s, \quad \mathbf{K}_{\Gamma\Gamma} = \sum_{i=1}^s \tilde{\mathbf{R}}_i^T \mathbf{K}_{\Gamma\Gamma}^i \tilde{\mathbf{R}}_i,$$

$$\mathbf{K}_{II}^i = \begin{pmatrix} \mathbf{M}_{II}^i & \mathbf{N}_{II}^i & (\mathbf{A}_{II}^i)^T \\ (\mathbf{N}_{II}^i)^T & \alpha \mathbf{Q}_{II}^i & (\mathbf{B}_{II}^i)^T \\ \mathbf{A}_{II}^i & \mathbf{B}_{II}^i & \end{pmatrix}, \quad \mathbf{K}_{\Gamma I}^i = \begin{pmatrix} \mathbf{M}_{\Gamma I}^i & \mathbf{N}_{\Gamma I}^i & (\mathbf{A}_{I\Gamma}^i)^T \\ \mathbf{A}_{\Gamma I}^i & \mathbf{B}_{\Gamma I}^i & \end{pmatrix}, \quad i \in \mathcal{C},$$

and

$$\mathbf{K}_{II}^i = \begin{pmatrix} \mathbf{M}_{II}^i & (\mathbf{A}_{II}^i)^T \\ \mathbf{A}_{II}^i & \end{pmatrix}, \quad \mathbf{K}_{\Gamma I}^i = \begin{pmatrix} \mathbf{M}_{\Gamma I}^i & (\mathbf{A}_{I\Gamma}^i)^T \\ \mathbf{A}_{\Gamma I}^i & \end{pmatrix}, \quad i \in \mathcal{N}.$$

Furthermore,

$$\mathbf{x}_\Gamma = \begin{pmatrix} \mathbf{y}_\Gamma \\ \mathbf{p}_\Gamma \end{pmatrix}, \quad \mathbf{g}_\Gamma = \begin{pmatrix} \mathbf{c}_\Gamma \\ \mathbf{b}_\Gamma \end{pmatrix}, \quad \mathbf{x}_I^i = \begin{pmatrix} \mathbf{y}_I^i \\ \mathbf{u}_I^i \\ \mathbf{p}_I^i \end{pmatrix}, \quad \mathbf{g}_I^i = \begin{pmatrix} \mathbf{c}_I^i \\ \mathbf{d}_I^i \\ \mathbf{b}_I^i \end{pmatrix}, \quad i \in \mathcal{C},$$

and

$$\mathbf{x}_I^i = \begin{pmatrix} \mathbf{y}_I^i \\ \mathbf{p}_I^i \end{pmatrix}, \quad \mathbf{g}_I^i = \begin{pmatrix} \mathbf{c}_I^i \\ \mathbf{b}_I^i \end{pmatrix}, \quad i \in \mathcal{N}.$$

Frequently, we use the compact notation

$$\begin{pmatrix} \mathbf{K}_{II} & \mathbf{K}_{\Gamma I}^T \\ \mathbf{K}_{\Gamma I} & \mathbf{K}_{\Gamma\Gamma} \end{pmatrix} \begin{pmatrix} \mathbf{x}_I \\ \mathbf{x}_\Gamma \end{pmatrix} = \begin{pmatrix} \mathbf{g}_I \\ \mathbf{g}_\Gamma \end{pmatrix} \tag{5.8}$$

or even $\mathbf{K}\mathbf{x} = \mathbf{g}$ instead of (5.7).

The matrix representation of the operators $\mathcal{H}_i^h$ defined in (4.15) and (4.16) is given by

$$\mathbf{H}_i = \begin{pmatrix} -(\mathbf{K}_{II}^i)^{-1}\mathbf{K}_{I\Gamma}^i \\ \mathbf{I} \end{pmatrix}.$$

The matrix representation of the local Schur complement operators $\mathcal{S}_i^h$ defined in (4.22) is given by

$$\mathbf{S}_i = \mathbf{K}_{\Gamma\Gamma}^i - \mathbf{K}_{\Gamma I}^i(\mathbf{K}_{II}^i)^{-1}(\mathbf{K}_{\Gamma I}^i)^T. \tag{5.9}$$

The matrix

$$\mathbf{K}^i = \begin{pmatrix} \mathbf{K}_{II}^i & (\mathbf{K}_{\Gamma I}^i)^T \\ \mathbf{K}_{\Gamma I}^i & \mathbf{K}_{\Gamma\Gamma}^i \end{pmatrix}$$

plays an important role in the computation of the inverse of $\mathbf{S}_i$ (assuming it exists). In fact, if $\mathbf{K}_{II}^i$ is invertible,

$$\mathbf{K}^i = \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{K}_{\Gamma I}^i(\mathbf{K}_{II}^i)^{-1} & \mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{K}_{II}^i & \mathbf{0} \\ \mathbf{0} & \mathbf{S}_i \end{pmatrix} \begin{pmatrix} \mathbf{I} & (\mathbf{K}_{II}^i)^{-1}(\mathbf{K}_{\Gamma I}^i)^T \\ \mathbf{0} & \mathbf{I} \end{pmatrix}, \tag{5.10}$$

and $\mathbf{S}_i$ is invertible if and only if $\mathbf{K}^i$ is invertible. In this case,

$$\mathbf{S}_i^{-1}\mathbf{v} = (\mathbf{0} \quad \mathbf{I})\,(\mathbf{K}^i)^{-1} \begin{pmatrix} \mathbf{0} \\ \mathbf{I} \end{pmatrix} \mathbf{v}$$

(see, e.g., [36, p. 113]).

The following theorem is concerned with the invertibility of the submatrices $\mathbf{K}_{II}^i$, which is important for the computation of $\mathbf{S}_i$, and with the invertibility of the submatrices $\mathbf{K}^i$, which is important for the computation of $(\mathbf{S}_i)^{-1}$.

THEOREM 5.3.

(i) $\sigma \geq 0$

a. $\mathbf{A}_{II}^i \in \mathbb{R}^{m_i^I \times m_i^I}$ $\mathbf{M}_{II}^i \in \mathbb{R}^{m_i^I \times m_i^I}$ $i = 1, \ldots, s$

b. $i \in \mathcal{C}$

$$\begin{pmatrix} \mathbf{z}_{II}^i \\ \mathbf{v}_{II}^i \end{pmatrix}^T \begin{pmatrix} \mathbf{M}_{II}^i & \mathbf{N}_{II}^i \\ (\mathbf{N}_{II}^i)^T & \alpha\mathbf{Q}_{II}^i \end{pmatrix} \begin{pmatrix} \mathbf{z}_{II}^i \\ \mathbf{v}_{II}^i \end{pmatrix} > 0 \tag{5.11}$$

$\mathbf{z}_{II}^i \in \mathbb{R}^{m_i^I}, \mathbf{v}_{II}^i \in \mathbb{R}^{n_i^I}$ $\mathbf{A}_{II}^i\mathbf{z}_{II}^i + \mathbf{B}_{II}^i\mathbf{v}_{II}^i = 0$

c. $\mathbf{K}_{II}^i$ $i = 1, \ldots, s$

(ii) $\sigma > 0$

a. $\mathbf{A}^i \in \mathbb{R}^{m_i \times m_i}$ $\mathbf{M}^i \in \mathbb{R}^{m_i \times m_i}$ $i = 1, \ldots, s$

b. [illegible] $i \in \mathcal{C}$

$$\begin{pmatrix} \mathbf{z}^i \\ \mathbf{v}^i \end{pmatrix}^T \begin{pmatrix} \mathbf{M}^i & \mathbf{N}^i \\ (\mathbf{N}^i)^T & \alpha \mathbf{Q}^i \end{pmatrix} \begin{pmatrix} \mathbf{z}^i \\ \mathbf{v}^i \end{pmatrix} > 0 \tag{5.12}$$

[illegible] $\mathbf{z}^i \in \mathbb{R}^{m_i}, \mathbf{v}^i \in \mathbb{R}^{n_i}$ [illegible] $\mathbf{A}^i \mathbf{z}^i + \mathbf{B}^i \mathbf{v}^i = 0$

c. [illegible] $\mathbf{K}^i$ $i = 1, \ldots, s$ [illegible]

(iii) [illegible] $\sigma = 0$ [illegible] $\text{null}(\mathbf{A}^i) = \text{span}\{\mathbf{e}\}$ [illegible] $\mathbf{e} \in \mathbb{R}^{m_i}$ [illegible]

$\mathbf{M}^i \in \mathbb{R}^{m_i \times m_i}$ [illegible] $i = 1, \ldots, s$

[illegible] $i \in \mathcal{C}$ [illegible] (5.12) [illegible] $\mathbf{z}^i \in \mathbb{R}^{m_i}, \mathbf{v}^i \in \mathbb{R}^{n_i}$

[illegible] $\mathbf{A}^i \mathbf{z}^i + \mathbf{B}^i \mathbf{v}^i = 0$ [illegible] $\mathbf{e}^T \mathbf{M}^i \mathbf{e} > 0$

[illegible] The matrices $\mathbf{A}^i_{II} \in \mathbb{R}^{m_i \times m_i}$ and $\mathbf{A}^i \in \mathbb{R}^{m_i \times m_i}$ are the stiffness matrices corresponding to the problems

$$-\Delta y + \sigma y = f \text{ in } \Omega_i, \quad y = 0 \text{ on } \partial\Omega_i,$$

and

$$-\Delta y + \sigma y = f \text{ in } \Omega_i, \quad \partial y / \partial n = 0 \text{ on } \partial\Omega_i,$$

respectively. This implies (a) that $\mathbf{A}^i_{II}$ is symmetric positive definite, (b) that $\mathbf{A}^i$ is symmetric positive definite if $\sigma > 0$, and (c) that $\text{null}(\mathbf{A}^i) = \text{span}\{\mathbf{e}\}$ if $\sigma = 0$.

The proof of assertions (i)–(iii) is identical to the proof of Theorem 5.2. ∎

[illegible] 5.4. Consider the model problem with $\sigma = 0$.

For $i \in \mathcal{N}$, the matrix $\mathbf{K}^i$ is singular. After a symmetric permutation, $\mathbf{K}^i$ is given by

$$\widetilde{\mathbf{K}^i} = \begin{pmatrix} \mathbf{M}^i & \mathbf{A}^i \\ \mathbf{A}^i & \mathbf{0} \end{pmatrix},$$

where

$$\mathbf{A}^i = \begin{pmatrix} \mathbf{A}^i_{II} & (\mathbf{A}^i_{\Gamma I})^T \\ \mathbf{A}^i_{\Gamma I} & \mathbf{A}^i_{\Gamma\Gamma} \end{pmatrix}, \quad \mathbf{M}^i = \begin{pmatrix} \mathbf{M}^i_{II} & (\mathbf{M}^i_{\Gamma I})^T \\ \mathbf{M}^i_{\Gamma I} & \mathbf{M}^i_{\Gamma\Gamma} \end{pmatrix}.$$

The vector $(\mathbf{0}^T, \mathbf{e}^T)^T \in \mathbb{R}^{2m_i}$ is in the null-space of $\widetilde{\mathbf{K}^i}$ and, if $\mathbf{M}^i \mathbf{e} = 0$, so is the vector $(\mathbf{e}^T, \mathbf{0}^T) \in \mathbb{R}^{2m_i}$.

If $i \in \mathcal{C}$ and $\mathbf{M}^i \mathbf{e} = 0$, the matrix $\mathbf{K}^i$ is singular. In fact, after a symmetric permutation, $\mathbf{K}^i$ is given by

$$\widetilde{\mathbf{K}^i} = \begin{pmatrix} \mathbf{M}^i & \mathbf{0} & \mathbf{A}^i \\ \mathbf{0} & \mathbf{Q}^i_{II} & (\mathbf{B}^i)^T \\ \mathbf{A}^i & \mathbf{B}^i & \mathbf{0} \end{pmatrix},$$

where $\mathbf{A}^i, \mathbf{M}^i$ are defined as before and

$$\mathbf{B}^i = \begin{pmatrix} \mathbf{B}^i_{II} \\ \mathbf{B}^i_{\Gamma I} \end{pmatrix}.$$

The vector $(\mathbf{e}^T, \mathbf{0}^T, \mathbf{0}^T)^T \in \mathbb{R}^{2m_i + n_i}$ is in the null-space of $\widetilde{\mathbf{K}^i}$.

By definition of $\mathbf{M}^i$,

$$\mathbf{e}^T \mathbf{M}^i \mathbf{e} = \sum_{x_j, x_k \in \Omega_i} \int_{\Omega_o} \phi_j(x) \phi_k(x) dx.$$

Hence $\mathbf{M}^i \mathbf{e} = \mathbf{0}$ if and only if $\Omega_i \cap \Omega_o = \emptyset$.

Theorem 5.3(i) guarantees that $\mathbf{K}_{II}$ is invertible. Hence, we can form the Schur complement system

$$\mathbf{S}\mathbf{x}_\Gamma = \mathbf{r}, \tag{5.13}$$

where

$$\mathbf{S} = \mathbf{K}_{\Gamma\Gamma} - \mathbf{K}_{\Gamma I}\mathbf{K}_{II}^{-1}\mathbf{K}_{\Gamma I}^T \tag{5.14}$$

and

$$\mathbf{r} = \mathbf{g}_\Gamma - \mathbf{K}_{\Gamma I}\mathbf{K}_{II}^{-1}\mathbf{g}_I.$$

The Schur complement matrix $\mathbf{S}$ can be written as a sum of subdomain Schur complement matrices,

$$\mathbf{S} = \sum_{i=1}^{s} \tilde{\mathbf{R}}_i^T \mathbf{S}_i \tilde{\mathbf{R}}_i, \tag{5.15}$$

where $\mathbf{S}_i$, $i = 1, \ldots, s$, is defined in (5.9).

THEOREM 5.5. (i) [illegible] $\sigma \geq 0$ [illegible] $\mathbf{S}$ [illegible] $m - \sum_{i=1}^{s} m_i^I$ [illegible] $m - \sum_{i=1}^{s} m_i^I$ [illegible] m_i^I [illegible] $\overline{\Omega}_i \setminus \Gamma$

(ii) [illegible] $\sigma > 0$ [illegible] $\mathbf{S}_i$ $i = 1, \ldots, s$ [illegible] $m_i - m_i^I$ [illegible] $m_i - m_i^I$ [illegible] m_i [illegible] $\overline{\Omega}_i$ [illegible] m_i^I [illegible] $\overline{\Omega}_i \setminus \Gamma$

(iii) [illegible] $\sigma = 0$ $i \in \mathcal{N}$ $i \in \mathcal{C}$ $\mathbf{e}^T\mathbf{M}^i\mathbf{e} = 0$ [illegible] $\mathbf{S}_i$ [illegible] $\mathbf{S}_i$ [illegible].

(i) Recall (5.14). It is easy to verify that

$$\begin{pmatrix} \mathbf{K}_{II} & \mathbf{K}_{\Gamma I}^T \\ \mathbf{K}_{\Gamma I} & \mathbf{K}_{\Gamma\Gamma} \end{pmatrix} = \begin{pmatrix} \mathbf{K}_{II} & 0 \\ \mathbf{K}_{\Gamma I} & \mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{K}_{II}^{-1} & 0 \\ 0 & \mathbf{S} \end{pmatrix} \begin{pmatrix} \mathbf{K}_{II} & \mathbf{K}_{\Gamma I}^T \\ 0 & \mathbf{I} \end{pmatrix}.$$

The matrix $\mathbf{K}$ is a symmetric permutation of the system matrix in (5.5) and, hence, both matrices have the same eigenvalues. By Theorem 5.2, the system matrix in (5.5), and hence $\mathbf{K}$, both have $m + n$ positive and m negative eigenvalues.

Let m_i^I be the number of finite element nodes in Ω_i and let n_i^I be the number of finite element nodes in $\partial\Omega_i \setminus \partial\Omega$. Applying the same arguments used to analyze the inertia of $\mathbf{K}$ to $\mathbf{K}_{II}$, one can show that the matrix $\mathbf{K}_{II}$ has $\sum_{i=1}^{s} m_i^I + n_i^I$ positive and $\sum_{i=1}^{s} m_i^I$ negative eigenvalues. By Sylvester's law of inertia, the number of positive (negative) eigenvalues of $\mathbf{K}$ is equal to the number of positive (negative) eigenvalues of $\mathbf{K}_{II}^{-1}$ plus the number of positive (negative) eigenvalues of $\mathbf{S}$. Since $n = \sum_{i=1}^{s} n_i^I$, this implies that $\mathbf{S}$ has $m - \sum_{i=1}^{s} m_i^I$ positive and $m - \sum_{i=1}^{s} m_i^I$ negative eigenvalues.

(ii) The second assertion can be proved using arguments analogous to those applied in (i). We omit the details.

(iii) By Lemma 5.4, $\mathbf{K}_{II}^i$ is invertible. In Remark 5.4 we have shown that $\mathbf{K}^i$ is singular if $i \in \mathcal{N}$ or if $i \in \mathcal{C}$ and $\mathbf{e}^T\mathbf{M}^i\mathbf{e} = 0$. In these cases the zero eigenvalue of $\mathbf{K}^i$ has mutiplicity one or two. Hence, the assertion follows from (5.10). ∎

5.3. The Neumann–Neumann preconditioners. It is now relatively easy to generalize the Neumann–Dirichlet and Neumann–Neumann preconditioners used in the context of elliptic PDEs (see, e.g., [36, Chap. 4] for an overview) to the optimal control context. We focus on Neumann–Neumann preconditioners.

Let $\mathbf{D}_i^y$ be the diagonal matrix whose entries are computed as follows. If $x_k \in \Gamma_i$, then $(\mathbf{D}_i^y)_{kk}^{-1}$ is the number of subdomains that share node x_k. Note that $\sum_i \mathbf{D}_i^y = \mathbf{I}$. Furthermore, let $\tilde{\mathbf{D}}_i^p = \tilde{\mathbf{D}}_i^y$ and

$$\mathbf{D}_i = \begin{pmatrix} \mathbf{D}_i^y & \\ & \mathbf{D}_i^p \end{pmatrix}.$$

By Theorem 5.3(i), $\mathbf{S}_i$, $i = 1, \ldots, s$, is well defined. If $\sigma > 0$, then Theorem 5.5(ii) guarantees the existence of $\mathbf{S}_i^{-1}$, $i = 1, \ldots, s$. The one-level Neumann–Neumann preconditioner is given by

$$\mathbf{P} = \sum_i \mathbf{D}_i \tilde{\mathbf{R}}_i^T \widehat{\mathbf{S}}_i^{-1} \tilde{\mathbf{R}}_i \mathbf{D}_i, \tag{5.16}$$

where $\widehat{\mathbf{S}}_i^{-1}$ is equal to $\mathbf{S}_i^{-1}$ if $\mathbf{S}_i$ is invertible and otherwise $\widehat{\mathbf{S}}_i^{-1}$ is the inverse of the Schur complement matrix $\widehat{\mathbf{S}}_i$ obtained by adding a positive multiple of $\int_{\Omega_i} \psi_k, \psi_j$ to the bilinear form a_i (cf. Remark 4.9). (With this modified bilinear form Theorem 5.5(ii) guarantees the invertibility of $\widehat{\mathbf{S}}_i$.)

It is well known that the performance of one-level Neumann–Neumann preconditioners for elliptic PDEs deteriorates quickly as the number of subdomains increases. The same is observed for the Neumann–Neumann preconditioner (5.16) in the optimal control context (see section 6). To avoid this, we include a coarse grid. More precisely, we adapt the balancing Neumann–Neumann preconditioner proposed in [30] to the optimal control context. Following the description in [36, sect. 4.3.3], the balancing Neumann–Neumann preconditioner for the optimal control problem is given by

$$\mathbf{P} = \left(\mathbf{I} - \tilde{\mathbf{R}}_0^T \mathbf{S}_0^{-1} \tilde{\mathbf{R}}_0 \mathbf{S}\right) \left(\sum_{i=1}^{s} \mathbf{D}_i \tilde{\mathbf{R}}_i^T \widehat{\mathbf{S}}_i^{-1} \tilde{\mathbf{R}}_i \mathbf{D}_i\right) \left(\mathbf{I} - \mathbf{S}\tilde{\mathbf{R}}_0^T \mathbf{S}_0^{-1} \tilde{\mathbf{R}}_0\right) + \tilde{\mathbf{R}}_0^T \mathbf{S}_0^{-1} \tilde{\mathbf{R}}_0, \tag{5.17}$$

where $\widehat{\mathbf{S}}_i^{-1}$ is defined as in (5.16), and where $\mathbf{S}_0 = \tilde{\mathbf{R}}_0 \mathbf{S} \tilde{\mathbf{R}}_0^T$ and $\tilde{\mathbf{R}}_0$ is defined as in (5.6) with $\tilde{\mathbf{R}}_0^y$ being the restriction operator which returns for each subdomain the weighted sum of the values of all the nodes on the boundary of that subdomain. The weight corresponding to an interface node is one divided by the number of subdomains in which the node is contained.

6. Numerical results. We consider the model problem (1.1) with $\Omega = \Omega_o = (-1, 1)^2$, $\sigma = 0$, $\widehat{f}(x) = (2\pi^2 + 1)\sin(\pi x_1)\sin(\pi x_2)$, and $\hat{y}(x) = \sin(\pi x_1)\sin(\pi x_2)$.

Additional numerical results for the model problem (1.1) with $\sigma = 1$ can be found in [20], and results using a problem with distributed control are given in [32, Chap. 4].

The domain $\Omega = (-1, 1)^2$ was partitioned into equal-sized square subdomains in a checkerboard pattern. The side length of each subdomain was denoted by H. Regular meshes consisting of triangular elements of various widths (denoted as h) were generated. The preconditioned KKT system was solved using symmetric QMR (sQMR) with the Neumann–Neumann and the balancing Neumann–Neumann preconditioners. In all cases, the stopping criteria was 10^{-8} for the ℓ_2-norm of the preconditioned residual.

TABLE 6.1

Number of iterations for the sQMR method with Neumann–Neumann and balancing Neumann–Neumann preconditioners for varying mesh sizes h, subdomain sizes H, and regularization parameters α.

Neumann–Neumann, $\alpha = 1.0$
sQMR iterations

H \ h	$\frac{1}{4}$	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{32}$	$\frac{1}{64}$	$\frac{1}{128}$
$\frac{1}{2}$	12	15	19	24	26	28
$\frac{1}{4}$		53	69	94	107	119
$\frac{1}{8}$			170	226	287	345
$\frac{1}{16}$				509	679	798
$\frac{1}{32}$					1578	2233

Bal. Neumann–Neumann, $\alpha = 1.0$
sQMR iterations

H \ h	$\frac{1}{4}$	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{32}$	$\frac{1}{64}$	$\frac{1}{128}$
$\frac{1}{2}$	5	6	8	10	11	12
$\frac{1}{4}$		5	9	12	14	15
$\frac{1}{8}$			5	10	13	15
$\frac{1}{16}$				5	9	13
$\frac{1}{32}$					4	9

Neumann–Neumann, $\alpha = 10^{-4}$
sQMR iterations

H \ h	$\frac{1}{4}$	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{32}$	$\frac{1}{64}$	$\frac{1}{128}$
$\frac{1}{2}$	13	14	16	19	21	23
$\frac{1}{4}$		55	54	63	67	71
$\frac{1}{8}$			129	143	165	187
$\frac{1}{16}$				432	439	465
$\frac{1}{32}$					1544	1588

Bal. Neumann–Neumann, $\alpha = 10^{-4}$
sQMR iterations

H \ h	$\frac{1}{4}$	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{32}$	$\frac{1}{64}$	$\frac{1}{128}$
$\frac{1}{2}$	6	9	9	11	11	13
$\frac{1}{4}$		8	11	13	16	19
$\frac{1}{8}$			7	10	13	17
$\frac{1}{16}$				6	10	14
$\frac{1}{32}$					5	11

Neumann–Neumann, $\alpha = 10^{-8}$
sQMR iterations

H \ h	$\frac{1}{4}$	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{32}$	$\frac{1}{64}$	$\frac{1}{128}$
$\frac{1}{2}$	15	16	16	21	21	23
$\frac{1}{4}$		58	61	63	74	76
$\frac{1}{8}$			217	191	202	214
$\frac{1}{16}$				583	536	584
$\frac{1}{32}$					1255	1249

Bal. Neumann–Neumann, $\alpha = 10^{-8}$
sQMR iterations

H \ h	$\frac{1}{4}$	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{32}$	$\frac{1}{64}$	$\frac{1}{128}$
$\frac{1}{2}$	9	10	13	15	17	20
$\frac{1}{4}$		11	17	21	26	30
$\frac{1}{8}$			13	18	24	30
$\frac{1}{16}$				12	19	24
$\frac{1}{32}$					11	16

The subdomain Schur complements are well defined, but for $\sigma = 0$ subdomain Schur complements corresponding to no-control subdomains are not invertible (see Theorem 5.5 and Remark 5.4 and recall that $\Omega = \Omega_o$). In this case we construct $\widehat{\mathbf{S}}_i$, where $\widehat{\mathbf{S}}_i$ is computed as $\mathbf{S}_i$ but with the bilinear form a_i replaced by

$$\widehat{a}_i : V_i \times V_i \to \mathbb{R}, \qquad \widehat{a}_i(y, \psi) = \int_{\Omega_i} \nabla y(x) \nabla \psi(x) + H^{-2} y(x) \psi(x) dx;$$

cf. Remark 4.9. The subproblem solves involving $\mathbf{K}_{II}^i$ and $\mathbf{K}^i$ (the latter constructed with bilinear form $\widehat{a}_i$) were performed using a sparse LU decomposition.

Table 6.1 summarizes the sQMR iteration counts for both preconditioners. These numerical results show that both the Neumann–Neumann and the balancing Neumann–Neumann preconditioners grow worse slowly when the mesh size h is reduced while other parameters are kept constant. This behavior is consistent with the PDE version of these preconditioners. As expected, the Neumann–Neumann preconditioner grows

worse quickly as the number of subdomains increases while the balancing Neumann–Neumann preconditioner remains effective. A notable result is that both preconditioners depend only weakly on the regularization parameter α. As α is reduced from 1 to 10^{-8}, the iteration count for the balancing Neumann–Neumann preconditioner grows by only a factor of two or less in most cases. The one-level Neumman–Neumann preconditioner appeared to be even less sensitive to small α.

7. Conclusion. We have introduced Neumann–Neumann preconditioners for linear-quadratic elliptic optimal control problems, which are obtained by introducing auxiliary states and adjoints on the subdomain interfaces and formulating subdomain optimality conditions with Dirichlet boundary conditions for states and adjoints on the subdomain interfaces. These subdomain optimality conditions are coupled by requiring continuity of normal derivatives of subdomain states and adjoints, defined as solutions of the local optimality conditions, on the subdomain interfaces. This leads to subdomain Schur complement operators that map Dirichlet interface data into Neumann interface data. The original optimality conditions are equivalent to a Schur complement system in which the operator is given by the sum of subdomain Schur complements.

The application of subdomain Schur complements has been shown to correspond to the solution of subdomain optimal control problems with Dirichlet boundary conditions on the subdomain interfaces that are otherwise smaller copies of the original optimal control problem. Conditions are given that establish the invertibility of subdomain Schur complements. The application of the inverses of local Schur complement operators has been shown to correspond to the solution of subdomain optimal control problems with Neumann boundary conditions on the subdomain interfaces that are otherwise smaller copies of the original optimal control problem. For a finite element discretization, the inertia of the local Schur complement matrices as well as that of the sum of the local matrices has been analyzed.

Numerical results for our (balancing) Neumann–Neumann preconditioners applied to a model problem indicate that the numerical performance of our (balancing) Neumann–Neumann DD methods applied to linear-quadratic elliptic optimal control problems is similar to that of their counterparts applied to single elliptic PDEs. We also observe that the performance of our preconditioners applied to the model problem (1.1) is largely independent of the size of the regularization parameter. A convergence theory for Neumann–Neumann methods applied to elliptic optimal control problems comparable to the convergence theory for these methods applied to individual symmetric elliptic PDEs [13, 14] or to the Stokes equation [33] is still missing.

While we have used an elliptic model optimal control problem with Neumann boundary control to illustrate the preconditioners, they can be applied to many other elliptic model optimal control problems. For example, it is straightforward to admit state equations with nonconstant coefficient functions (although the scaling matrices $\mathbf{D}_i$ may have to be adjusted if there are large jumps in coefficients across subdomains [13, 14]). It is also easy to extend the methods to distributed controls (see [32, Chap. 4]). Recently, we have extended the approach in this paper to advection dominated optimal control problems [2]. This requires a careful adjustment of the interface conditions to account for advection terms.

Acknowledgment. The authors thank the anonymous referees for their careful reading and suggestions, which have led to an improvement in the presentation of the paper.

REFERENCES

[1] U. M. Ascher and E. Haber, *A multigrid method for distributed parameter estimation problems*, in Tenth Copper Mountain Conference on Multigrid Methods (Copper Mountain, CO, 2001), Electron. Trans. Numer. Anal., 15 (2003), pp. 1–17.

[2] R. Bartlett, M. Heinkenschloss, D. Ridzal, and B. van Bloemen Waanders, *Domain decomposition methods for advection dominated linear-quadratic elliptic optimal control problems*, Comput. Methods Appl. Mech. Engrg., to appear.

[3] A. Battermann and M. Heinkenschloss, *Preconditioners for Karush–Kuhn–Tucker matrices arising in the optimal control of distributed systems*, in Control and Estimation of Distributed Parameter Systems, W. Desch, F. Kappel, and K. Kunisch, eds., Internat. Ser. Numer. Math. 126, Birkhäuser-Verlag, Basel, 1998, pp. 15–32.

[4] A. Battermann and E. W. Sachs, *Block preconditioners for KKT systems in PDE-governed optimal control problems*, in Fast Solution of Discretized Optimization Problems (Berlin, 2000), K.-H. Hoffmann, R. H. W. Hoppe, and V. Schulz, eds., Internat. Ser. Numer. Math. 138, Birkhäuser-Verlag, Basel, 2001, pp. 1–18.

[5] J.-D. Benamou, *A Domain Decomposition Method with Coupled Transmission Conditions for the Optimal Control of Systems Governed by Elliptic Partial Differential Equations*, Technical report 2246, INRIA, Rocquencourt, France, 1994.

[6] J.-D. Benamou, *A domain decomposition method with coupled transmission conditions for the optimal control of systems governed by elliptic partial differential equations*, SIAM J. Numer. Anal., 33 (1996), pp. 2401–2416.

[7] J.-D. Benamou, *A domain decomposition method for control problems*, in Proceedings of the 9th International Conference on Domain Decomposition, P. Bjørstad, M. Espedal, and D. Keyes, eds., Domain Decomposition Press, Bergen, Norway, 1998, pp. 266–273.

[8] M. Benzi, G. H. Golub, and J. Liesen, *Numerical solution of saddle point problems*, Acta Numer., 14 (2005), pp. 1–137.

[9] G. Biros and O. Ghattas, *Parallel preconditioners for KKT systems arising in optimal control of viscous incompressible flows*, in Proceedings of Parallel CFD '99, Williamsburg, VA, 1999, North–Holland, Amsterdam, London, New York, 1999; available online from http://www.cs.cmu.edu/∼oghattas.

[10] G. Biros and O. Ghattas, *Parallel Lagrange–Newton–Krylov–Schur methods for PDE-constrained optimization. Part* I: *The Krylov–Schur solver*, SIAM J. Sci. Comput., 27 (2005), pp. 687–713.

[11] A. Bounaim, *A Lagrangian approach to a DDM for an optimal control problem*, in Proceedings of the 9th International Conference on Domain Decomposition, P. Bjørstad, M. Espedal, and D. Keyes, eds., Domain Decomposition Press, Bergen, Norway, 1998, pp. 283–289.

[12] J. E. Dennis and R. M. Lewis, *A Comparison of Nonlinear Programming Approaches to an Elliptic Inverse Problem and a New Domain Decomposition Approach*, Technical report TR94–33, Department of Computational and Applied Mathematics, Rice University, Houston, TX, 1994.

[13] M. Dryja, B. F. Smith, and O. B. Widlund, *Schwarz analysis of iterative substructuring algorithms for elliptic problems in three dimensions*, SIAM J. Numer. Anal., 31 (1994), pp. 1662–1694.

[14] M. Dryja and O. B. Widlund, *Schwarz methods of Neumann-Neumann type for three-dimensional elliptic finite element methods*, Comm. Pure Appl. Math., 48 (1995), pp. 121–155.

[15] P. E. Gill, W. Murray, D. B. Ponceleón, and M. A. Saunders, *Preconditioners for indefinite systems arising in optimization*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 292–311.

[16] G. H. Golub and C. Greif, *On solving block-structured indefinite linear systems*, SIAM J. Sci. Comput., 24 (2003), pp. 2076–2092.

[17] N. I. M. Gould, *On practical conditions for the existence and uniqueness of solutions to the general equality quadratic programming problem*, Math. Programming, 32 (1985), pp. 90–99.

[18] N. I. M. Gould, M. E. Hribar, and J. Nocedal, *On the solution of equality constrained quadratic programming problems arising in optimization*, SIAM J. Sci. Comput., 23 (2001), pp. 1376–1395.

[19] E. Haber and U. Ascher, *Preconditioned all-at-once methods for large, sparse parameter estimation problems*, Inverse Problems, 17 (2001), pp. 1847–1864.

[20] M. Heinkenschloss and H. Nguyen, *Balancing Neumann-Neumann methods for elliptic optimal control problems*, in Domain Decomposition Methods in Science and Engineering, R. Kornhuber, R. H. W. Hoppe, J. Periaux, O. Pironneau, O. B. Widlund, and J. Xu, eds., Lect. Notes Comput. Sci. Eng. 40, Springer-Verlag, Heidelberg, 2004, pp. 589–596.

[21] R. H. W. Hoppe, S. I. Petrova, and V. Schulz, *Primal-dual Newton-type interior-point method for topology optimization*, J. Optim. Theory Appl., 114 (2002), pp. 545–571.
[22] J. Jahn, *Introduction to the Theory of Nonlinear Optimization*, 2nd ed., Springer-Verlag, Berlin, Heidelberg, New York, 1996.
[23] C. Keller, N. I. M. Gould, and A. J. Wathen, *Constraint preconditioning for indefinite linear systems*, SIAM J. Matrix Anal. Appl., 21 (2000), pp. 1300–1317.
[24] A. Klawonn and O. B. Widlund, *FETI and Neumann-Neumann iterative substructuring methods: Connections and new results*, Comm. Pure Appl. Math., 54 (2001), pp. 57–90.
[25] K. Kunisch and X.-C. Tai, *Nonoverlapping domain decomposition methods for inverse problems*, in Proceedings of the 9th International Conference on Domain Decomposition Methods, Domain Decomposition Press, Bergen, Norway, 1998, pp. 517–524.
[26] P. Le Tallec, *Domain decomposition methods in computational mechanics*, Comput. Mech. Adv., 1 (1994), pp. 121–220.
[27] J.-L. Lions, *Optimal Control of Systems Governed by Partial Differential Equations*, Springer-Verlag, Berlin, Heidelberg, New York, 1971.
[28] J.-L. Lions and O. Pironneau, *Sur le contrôle parallèle des systèmes distribués*, C.R. Acad. Sci. Paris Sér. I Math., 327 (1998), pp. 993–998.
[29] L. Lukšan and J. Vlček, *Indefinitely preconditioned inexact Newton method for large sparse equality constrained non-linear programming problems*, Numer. Linear Algebra Appl., 5 (1998), pp. 219–247.
[30] J. Mandel, *Balancing domain decomposition*, Comm. Numer. Methods Engrg., 9 (1993), pp. 233–241.
[31] M. F. Murphy, G. H. Golub, and A. J. Wathen, *A note on preconditioning for indefinite linear systems*, SIAM J. Sci. Comput., 21 (2000), pp. 1969–1972.
[32] H. Q. Nguyen, *Domain Decomposition Methods for Linear-Quadratic Elliptic Optimal Control Problems*, Ph.D. thesis, Department of Computational and Applied Mathematics, Rice University, Houston, TX, 2004; available online from http://www.caam.rice.edu/caam/trs/2004/TR04-16.pdf.
[33] L. F. Pavarino and O. B. Widlund, *Balancing Neumann-Neumann methods for incompressible Stokes equations*, Comm. Pure Appl. Math., 55 (2002), pp. 302–335.
[34] A. Quarteroni and A. Valli, *Domain Decomposition Methods for Partial Differential Equations*, Oxford University Press, Oxford, UK, 1999.
[35] M. Renardy and R. C. Rogers, *An Introduction to Partial Differential Equations*, Texts Appl. Math. 13, Springer-Verlag, Berlin, Heidelberg, New York, 1993.
[36] B. Smith, P. Bjørstad, and W. Gropp, *Domain Decomposition. Parallel Multilevel Methods for Elliptic Partial Differential Equations*, Cambridge University Press, Cambridge, London, New York, 1996.
[37] X.-C. Tai, J. Frøyen, M. S. Espedal, and T. F. Chan, *Overlapping domain decomposition and multigrid methods for inverse problems*, in Domain Decomposition Methods 10 (Boulder, CO, 1997), Contemp. Math. 218, AMS, Providence, RI, 1998, pp. 523–529.
[38] A. Toselli and O. Widlund, *Domain decomposition methods—Algorithms and theory*, Springer Ser. Comput. Math. 34, Springer-Verlag, Berlin, 2005.
[39] J. Wloka, *Partial Differential Equations*, Cambridge University Press, Cambridge, UK, 1987.

SIAM J. SCI. COMPUT.
Vol. 28, No. 3, pp. 1029–1053
© 2006 Society for Industrial and Applied Mathematics

SOLVING THE BOLTZMANN EQUATION IN $N \log_2 N$*

FRANCIS FILBET†, CLÉMENT MOUHOT‡, AND LORENZO PARESCHI§

Abstract. In [C. Mouhot and L. Pareschi, "Fast algorithms for computing the Boltzmann collision operator," *Math. Comp.*, to appear; C. Mouhot and L. Pareschi, *C. R. Math. Acad. Sci. Paris*, 339 (2004), pp. 71–76], fast deterministic algorithms based on spectral methods were derived for the Boltzmann collision operator for a class of interactions including the *hard spheres model* in three dimensions. These algorithms are implemented for the solution of the Boltzmann equation in two and three dimensions, first for homogeneous solutions, then for general nonhomogeneous solutions. The results are compared to explicit solutions, when available, and to Monte Carlo methods. In particular, the computational cost and accuracy are compared to those of Monte Carlo methods as well as to those of previous spectral methods. Finally, for inhomogeneous solutions, we take advantage of the great computational efficiency of the method to show an oscillation phenomenon of the entropy functional in the trend to equilibrium, which was suggested in the work [L. Desvillettes and C. Villani, *Invent. Math.*, 159 (2005), pp. 245–316].

Key words. Boltzmann equation, spectral methods, fast algorithms, entropy

AMS subject classifications. 65T50, 68Q25, 74S25, 76P05

DOI. 10.1137/050625175

1. Introduction.

The construction of approximate methods of a solution for the Boltzmann equation has a long history and can be traced back to Hilbert, Chapmann, and Enskog [13] at the beginning of the last century. The mathematical difficulties related to the Boltzmann equation make it extremely difficult, if not impossible, to determine analytic solutions in most physically relevant situations. Only in recent years, starting in the 1970s with the pioneering works by Chorin [14] and Sod [46], has the problem been tackled numerically with particular care given to accuracy and computational cost. Even nowadays the deterministic numerical solution of the Boltzmann equation still represents a challenge for scientific computing.

Most of the difficulties are due to the multidimensional structure of the collisional integral, since the integration runs on a highly dimensional unflat manifold. In addition the numerical integration requires great care since the collision integral is at the basis of the macroscopic properties of the equation. Further difficulties are represented by the presence of stiffness, such as the case of small mean free path [24] or the case of large velocities [19].

For such reasons realistic numerical simulations are based on Monte Carlo techniques. The most famous examples are the direct simulation Monte Carlo (DSMC) methods by Bird [3] and Nanbu [36]. These methods guarantee efficiency and preservation of the main physical properties. However, avoiding statistical fluctuations in the results becomes extremely expensive in the presence of nonstationary flows or when close to continuum regimes.

*Received by the editors February 24, 2005; accepted for publication (in revised form) January 5, 2006; published electronically June 30, 2006. This work was supported by the European network HYKE, funded by the EC as contract HPRN-CT-2002-00282.

http://www.siam.org/journals/sisc/28-3/62517.html

†MIP, Université Paul Sabatier 118, route de Narbonne, 31062 Toulouse Cedex 04, France (filbet@mip.ups-tlse.fr).

‡CEREMADE, Université Paris IX, Place du Mal de Lattre de Tassigny, 75775 Paris Cedex 16, France (cmouhot@ceremade.dauphine.fr).

§Dipartimento di Matematica, Universitá di Ferrara, Via Machiavelli 35, 44100 Ferrara, Italy (lorenzo.pareschi@unife.it).

Among deterministic approximations, perhaps the most popular method is represented by the so-called discrete velocity models (DVM) of the Boltzmann equation. These methods [29, 45, 6, 10, 39] are based on a Cartesian grid in velocity and on a discrete collision mechanism on the points of the grid that preserves the main physical properties. Unfortunately DVM are not competitive with Monte Carlo methods in terms of computational cost, and their accuracy seems to be less than first order [37, 38, 39]. In this work we are interested in high-order deterministic methods and therefore we shall not discuss algorithms based on DVM. We refer the reader to the work in preparation [34].

More recently a new class of numerical methods based on the use of spectral techniques in the velocity space has been developed. The methods were first derived in [40], inspired from spectral methods in fluid mechanics [11] and by previous works on the use of Fourier transform techniques for the Boltzmann equation (see [5], for instance). The numerical methods are based on approximating the distribution function by a periodic function in the phase space and on its representation by Fourier series. The resulting Fourier–Galerkin approximation can be evaluated with a computational cost of $O(n^2)$ (where n is the total number of discretization parameters in velocity), which is lower than that of previous deterministic methods (but still larger than that of Monte Carlo methods).

The new methods were further developed in [41, 43], where evolution equations for the Fourier modes were explicitly derived and spectral accuracy of the methods was proven. Strictly speaking these methods are not conservative, since they preserve mass, whereas momentum and energy are approximated with spectral accuracy. This trade-off between accuracy and conservation seems to be an unavoidable compromise in the development of numerical schemes for the Boltzmann equation (with the noticeable exception of [28]).

We recall here that the spectral method has been applied also to nonhomogeneous situations [20, 22]; to the Landau equation [19, 42], where fast algorithms can be readily derived; and to the case of granular gases [35, 21]. For a recent introduction to numerical methods for the Boltzmann equation and related kinetic equations we refer the reader to [15]. Finally, let us mention that Bobylev and Rjasanow [7, 8] have also constructed fast algorithms based on a Fourier transform approximation of the distribution function.

In [33, 32] a fast spectral method was proposed for a class of particle interactions, including pseudo-Maxwell molecules (in dimension 2) and hard spheres (in dimension 3), on the basis of the previous spectral method together with a suitable semidiscretization of the collision operator. This method permits us to reduce the computational cost from $O(n^2)$ to $O(n \log_2 n)$ without losing the spectral accuracy, and thus making the method competitive with Monte Carlo. The principles and basic features of this new method will be presented in the following sections.

The rest of the paper is organized as follows. Section 2 is devoted to a short introduction on the Boltzmann equation and its physical properties. In section 3 we explain the principles of the different spectral algorithms used to compute the collision operator. Several numerical results and comparisons to exact solutions as well as to Monte Carlo methods are given in section 4. An application to a challenging nonhomogeneous test case is finally given in section 5. Some final considerations close the paper in the last section.

2. The Boltzmann equation. The Boltzmann equation describes the behavior of a dilute gas of particles when the only interactions taken into account are binary

elastic collisions. It reads as $x \in \Omega$, $v \in \mathbb{R}^d$, where $\Omega \subset \mathbb{R}^d$ is the spatial domain ($d \geq 2$)

$$\frac{\partial f}{\partial t} + v \cdot \nabla_x f = Q(f, f),$$

where $f(t, x, v)$ is the time-dependent particles distribution function in the phase space. The Boltzmann collision operator Q is a quadratic operator local in (t, x). The time and position act only as parameters in Q and therefore will be omitted in its description

$$Q(f, f)(v) = \int_{v_* \in \mathbb{R}^d} \int_{\sigma \in \mathbb{S}^{d-1}} B(|v - v_*|, \cos\theta)\ (f'_* f' - f_* f)\ d\sigma\, dv_*. \tag{2.1}$$

In (2.1) we used the shorthand $f = f(v)$, $f_* = f(v_*)$, $f' = f(v')$, $f'_* = f(v'_*)$. The velocities of the colliding pairs (v, v_*) and (v', v'_*) are related by

$$v' = v - \frac{1}{2}\big((v - v_*) - |v - v_*|\,\omega\big), \qquad v'_* = v - \frac{1}{2}\big((v - v_*) + |v - v_*|\,\omega\big).$$

The collision kernel B is a nonnegative function which by physical arguments of invariance depends only on $|v - v_*|$ and $\cos\theta = \hat{g} \cdot \omega$ (where $\hat{g} = (v - v_*)/|v - v_*|$). In this work we are concerned with [illegible] models. More precisely we assume that B is locally integrable. This assumption is satisfied by the [illegible], which in dimension $d = 3$ is written as

$$B(|v - v_*|, \cos\theta) = |v - v_*| \tag{2.2}$$

and is known as [illegible] when it is (artificially) extended to interactions deriving from power-law potentials. As an important benchmark model for the numerical simulation we therefore introduce the so-called [illegible], which is written as

$$B(|v - v_*|, \cos\theta) = C_\gamma\, |v - v_*|^\gamma, \tag{2.3}$$

for some $\gamma \in [0, 1]$ and a constant $C_\gamma > 0$.

For this class of model, one can split the collision operator into

$$Q(f, f) = Q^+(f, f) - L(f)\, f,$$

with

$$Q^+(f, f) = \int_{\mathbb{R}^d} \int_{S^{d-1}} B(|v - v_*|, \cos\theta) f' f'_*\, d\sigma\, dv_*, \tag{2.4}$$

$$L(f) = \int_{\mathbb{R}^d} \int_{S^{d-1}} B(|v - v_*|, \cos\theta) f_*\, d\sigma\, dv_*. \tag{2.5}$$

Boltzmann's collision operator has the fundamental properties of conserving mass, momentum, and energy

$$\int_{v \in \mathbb{R}^d} Q(f, f)\, \phi(v)\, dv = 0, \qquad \phi(v) = 1, v, |v|^2,$$

and satisfies the well-known Boltzmann's H theorem

$$-\frac{d}{dt}\int_{v\in\mathbb{R}^d} f\log f\,dv = -\int_{v\in\mathbb{R}^d} Q(f,f)\log(f)\,dv \geq 0.$$

The functional $-\int f\log f$ is the [illegible] of the solution. Boltzmann's H theorem implies that any equilibrium distribution function, i.e., any function which is a maximum of the entropy, has the form of a locally Maxwellian distribution

$$M(\rho,u,T)(v) = \frac{\rho}{(2\pi T)^{d/2}}\exp\left(-\frac{|u-v|^2}{2T}\right), \tag{2.6}$$

where ρ, u, T are the [illegible], and [illegible] of the gas, defined, respectively, by

$$\rho = \int_{v\in\mathbb{R}^d} f(v)dv, \quad u = \frac{1}{\rho}\int_{v\in\mathbb{R}^d} vf(v)dv, \quad T = \frac{1}{d\rho}\int_{v\in\mathbb{R}^d} |u-v|^2 f(v)dv. \tag{2.7}$$

For further details on the physical background and derivation of the Boltzmann equation we refer to [13] and [49].

3. The spectral methods. In this section we shall explain the principles of the algorithms for computing the collision integral for a fixed value of the spatial variable x. Indeed, it is well known that one can reduce to this case by some splitting strategy (see [41, 20], for example).

3.1. A general framework. We consider the spatially homogeneous Boltzmann equation written in the general form

$$\frac{\partial f}{\partial t} = Q(f,f), \tag{3.1}$$

where Q is given by

$$Q(f,f) = \int_{\{(y,z)\in\mathcal{C}\}} \mathcal{B}(y,z)\big(f'f'_* - f_*f\big)\,dy\,dz, \quad v\in\mathbb{R}^d, \tag{3.2}$$

with

$$v' = v+\Theta'(y,z), \qquad v'_* = v+\Theta'_*(y,z), \qquad v_* = v+\Theta_*(y,z).$$

In the equations above, $\mathcal{C}$ is some given unbounded domain, and Θ, Θ', Θ'_* are suitable functions, to be defined later. This general framework emphasizes the translation invariance property of the collision operator, which is crucial for the spectral methods. We will explain more precisely in the next paragraphs some changes of variables which allow us to reduce the classical operator (2.1) to the form (3.2).

A problem associated with deterministic methods which use a fixed discretization in the velocity domain is that the velocity space is approximated by a finite region. Physically the domain for the velocity is $\mathbb{R}^d$, and the property of having compact support is not preserved by the collision operator. In general the collision process spreads the support by a factor of $\sqrt{2}$ in the elastic case (see [44, 31] and also [30] for similar properties in the inelastic case). As a consequence, for the continuous equation in time, the function f is immediately positive in the whole domain $\mathbb{R}^d$. Thus, at the numerical level, some nonphysical condition has to be imposed to keep the support of the function in velocity uniformly bounded. In order to do this there are two main strategies.

- One can remove the physical binary collisions that will lead outside the bounded velocity domain, which means a possible increase of the number of local invariants (that is, the functions φ such that $(\varphi'_* + \varphi' - \varphi_* - \varphi)$ is zero everywhere on the domain). If this is done properly (i.e., without removing too many collisions), the scheme remains conservative (and without spurious invariants). However, this truncation breaks down the convolution-like structure of the collision operator, which requires the translation invariance in velocity. Indeed, the modified collision kernel depends on v through the boundary conditions. This truncation is the starting point of most schemes based on DVM in a bounded domain.
- One can add some nonphysical binary collisions by periodizing the function and the collision operator. This implies the loss of some local invariants (some nonphysical collisions are added). Thus the scheme is no longer conservative, except for the mass if the periodization is done carefully. In this way the structural properties of the collision operator are maintained and thus they can be exploited to derive fast algorithms. This periodization is the basis of the spectral methods.

Therefore, we consider the space homogeneous Boltzmann equation in a bounded domain in velocity $\mathcal{D}_T = [T;T]^d$ $(0 < T < \infty)$. We need to truncate the integration in y and z, since periodization would yield an infinite result if not. Thus we set y and z to belong to some truncated domain $\mathcal{C}_R \subset \mathcal{C}$ (the parameter R refers to its size and will be defined later). For a compactly supported function with support included in B_S, the ball centered at 0 with radius $S > 0$, one has to prescribe suitable relations (depending on the precise change of variable and truncation chosen) between S, R, T in order to retain all possible collisions and at the same time prevent intersections of the regions, where f is different from zero (dealiasing condition). Then the [illegible] collision operator reads

$$Q^R(f,f) = \int_{\mathcal{C}_R} \mathcal{B}(y,z) \left(f'_* f' - f_* f\right) dy\, dz \tag{3.3}$$

for $v \in \mathcal{D}_T$ (the expression for $v \in \mathbb{R}^d$ is deduced by periodization). By making some changes of variable on v, one can easily prove for the two choices of variables y, z in the next subsections that for any function φ periodic on $\mathcal{D}_T$ the following weak form is satisfied:

$$\int_{\mathcal{D}_T} Q^R(f,f)\,\varphi(v)\,dv = \frac{1}{4}\int_{\mathcal{D}_T}\int_{\mathcal{C}_R} \mathcal{B}(y,z)\, f_* f\, (\varphi'_* + \varphi' - \varphi_* - \varphi)\, dy\, dz\, dv. \tag{3.4}$$

Now we use the representation Q^R to derive spectral methods. Hereafter we use just one index to denote the d-dimensional sums with respect to the vector $k = (k_1, \ldots, k_d) \in \mathbb{Z}^d$, and hence we set

$$\sum_{k=-N}^{N} := \sum_{k_1,\ldots,k_d=-N}^{N}.$$

The approximate function f_N is represented as the truncated Fourier series

$$f_N(v) = \sum_{k=-N}^{N} \hat{f}_k\, e^{i\frac{\pi}{T}k\cdot v}, \tag{3.5}$$

$$\hat{f}_k = \frac{1}{(2T)^d}\int_{\mathcal{D}_T} f(v)\, e^{-i\frac{\pi}{T}k\cdot v}\, dv.$$

In a Fourier–Galerkin method the fundamental unknowns are the coefficients $\hat{f}_k$, $k = -N, \dots, N$. We obtain a set of ODEs for the coefficients $\hat{f}_k$ by requiring that the residual of (3.3) be orthogonal to all trigonometric polynomials of degree less than N. Hence for $k = -N, \dots, N$

$$\int_{\mathcal{D}_T} \left(\frac{\partial f_N}{\partial t} - Q^R(f_N, f_N) \right) e^{-i\frac{\pi}{T}k\cdot v}\, dv = 0. \tag{3.6}$$

By substituting expression (3.5) in (3.4) we get

$$Q^R(f_N, f_N) = Q^{R,+}(f_N, f_N) - L^R(f_N)\, f_N,$$

with

$$L^R(f_N)\, f_N = \sum_{l=-N}^{N} \sum_{m=-N}^{N} \beta(m,m)\, \hat{f}_l\, \hat{f}_m e^{i\frac{\pi}{T}(l+m)\cdot v}, \tag{3.7}$$

$$Q^{R,+}(f_N, f_N) = \sum_{l=-N}^{N} \sum_{m=-N}^{N} \beta(l,m)\, \hat{f}_l\, \hat{f}_m e^{i\frac{\pi}{T}(l+m)\cdot v}, \tag{3.8}$$

where

$$\beta(l,m) = \int_{\mathcal{C}_R} \mathcal{B}(y,z) e^{i\frac{\pi}{T}(l\cdot\Theta'(y,z) + m\cdot\Theta'_*(y,z))}\, dy\, dz. \tag{3.9}$$

The [illegible] is the projection of the collision equation in $\mathbb{P}_N$, the $(2N+1)^d$-dimensional vector space of trigonometric polynomials of degree at most N in each direction, i.e.,

$$\frac{\partial f_N}{\partial t} = \mathcal{P}_N\, Q^R(f_N, f_N),$$

where $\mathcal{P}_N$ denotes the orthogonal projection on $\mathbb{P}_N$ in $L^2(\mathcal{D}_T)$. A straightforward computation leads to the following set of ODEs on the Fourier coefficients

$$\frac{\partial \hat{f}_k}{\partial t} = \sum_{\substack{l+m=k \\ l,m=-N}}^{N} \hat{\beta}(l,m)\, \hat{f}_l\, \hat{f}_m, \tag{3.10}$$

where $\hat{\beta}(l,m)$ are the so-called [illegible], given by

$$\hat{\beta}(l,m) = \beta(l,m) - \beta(m,m),$$

with the initial condition

$$\hat{f}_k(0) = \frac{1}{(2T)^d} \int_{\mathcal{D}_T} f_0(v)\, e^{-i\frac{\pi}{T}k\cdot v}\, dv. \tag{3.11}$$

3.2. Classical spectral methods. In the classical spectral method [41], a simple change of variables in (2.1) permits us to write

$$Q(f,f) = \int_{\mathbb{R}^d} \int_{S^{d-1}} \mathcal{B}^c(g,\omega)\big(f(v')f(v'_*) - f(v)f(v_*)\big)\, d\omega\, dg, \tag{3.12}$$

with $g = v - v_* \in \mathbb{R}^d$, $\omega \in \mathbb{S}^{d-1}$, and

$$\begin{cases} v' = v - \frac{1}{2}(g - |g|\omega) =: v + \Theta'(g,\omega), \\ v'_* = v - \frac{1}{2}(g + |g|\omega) =: v + \Theta'_*(g,\omega), \\ v_* = v + g =: v + \Theta_*(g,\omega). \end{cases} \tag{3.13}$$

Finally, $\mathcal{B}^c$ is defined by

$$\mathcal{B}^c(g,\omega) = 2^{d-1}\left(1 - (\hat{g}\cdot\omega)\right)^{d/2-1} B\big(|g|, 2(\hat{g}\cdot\omega)^2 - 1\big). \tag{3.14}$$

The Boltzmann operator (3.12) is now written in the form (3.2) with $(y,z) = (g,\omega) \in \mathbb{R}^d \times S^{d-1} =: \mathcal{C}$. Moreover, from the conservation of the momentum $v'_* + v' = v_* + v$ and the energy $|v'_*|^2 + |v'|^2 = |v_*|^2 + |v|^2$, we get the following results [40], assuming supp $f \subset B_S$:

- We have supp $Q(f,f) \subset B_{\sqrt{2}\,S}$;
- the collision operator is then given by

$$Q(f,f)(v) = \int_{B_{2S}} \int_{S^{d-1}} B(|g|, \cos\theta)\big(f(v')f(v'_*) - f(v_*)f(v)\big)\, d\omega\, dg,$$

with $v', v'_*, v_* \in B_{(2+\sqrt{2})R}$.

As a consequence of this result, in order to write a spectral approximation to (3.1) we consider the distribution function f restricted on $[-T,T]^d$ $(0 < T < +\infty)$, assuming $f(v) = 0$ on $[-T,T]^d \setminus B_S$, and extend it by periodicity to a periodic function on $[-T,T]^d$. We truncate the domain for $(y,z) = (g,\omega)$ as $\mathcal{C}_R = B_R \times \mathbb{S}^{d-1}$ for $R > 0$ (thus defining Q^R). Following the previous discussion on the dealiasing condition, we take $R = 2S$, and the shortest period can be restricted to $[-T,T]^d$, with $T \geq (3+\sqrt{2})S/2$ (for a more detailed discussion see [41]).

Then we apply the spectral algorithm (3.7), (3.8) and get the coefficients $\beta^c(l,m)$

$$\beta^c(l,m) = \int_{B_R} \int_{S^{d-1}} B(|g|, \cos\theta)\, e^{-i\frac{\pi}{T}\left(g\cdot\frac{(l+m)}{2} - i|g|\omega\cdot\frac{(m-l)}{2}\right)}\, d\omega\, dg. \tag{3.15}$$

We refer to [41, 22] for the explicit computation of Fourier coefficients $\beta^c(l,m)$ in the VHS case, where B is given by (2.3). Now the evaluation of the right-hand side of (3.10) requires exactly $O(N^{2d})$ operations. We emphasize that the usual cost for a DVM method based on N^d parameters for f in the velocity space is $O(N^{2d}M)$, where M is the numbers of angle discretizations.

3.3. Fast spectral methods. Here we shall approximate the collision operator starting from a representation which conserves more symmetries of the collision operator when one truncates it in a bounded domain. This representation was used

in [7, 26] to derive finite differences schemes and is close to the classical Carleman representation (cf. [12]). The basic identity we shall need is (for $u \in \mathbb{R}^d$)

$$\frac{1}{2}\int_{S^{d-1}} F(|u|\sigma - u)\, d\sigma = \frac{1}{|u|^{d-2}}\int_{\mathbb{R}^d} \delta(2\, y\cdot u + |y|^2)\, F(y)\, dy. \tag{3.16}$$

Using (3.16) the collision operator (2.1) can be written as

$$\begin{aligned} Q(f,f)(v) = 2^{d-1}\int_{x\in\mathbb{R}^d}\int_{y\in\mathbb{R}^d} \mathcal{B}^f(y,z)\,\delta(y\cdot z) \\ \times\big(f(v+z)f(v+y) - f(v+y+z)f(v)\big)\, dy\, dz, \end{aligned} \tag{3.17}$$

with

$$\mathcal{B}^f(y,z) = 2^{d-1}\, B\left(|y+z|, -\frac{y\cdot(y+z)}{|y||y+z|}\right)\, |y+z|^{-(d-2)}.$$

Thus the collision operator is now written in the form (3.2), with $(y,z)\in\mathbb{R}^d\times\mathbb{R}^d =: \mathcal{C}$, $\mathcal{B}(y,z) = \mathcal{B}^f(y,z)\,\delta(y\cdot z)$, and $v'_* = v + z =: v + \Theta'_*(y,z)$, $v' = v + y =: v + \Theta'(y,z)$, $v_* = v + y + z =: v + \Theta_*(y,z)$.

Now we consider the bounded domain $\mathcal{D}_T = [T,T]^d$ $(0 < T < \infty)$ for the distribution f, and the bounded domain $B_R \times B_R$ for (y,z) (for some $R > 0$). If f has support included in B_S, $S > 0$, geometrical arguments similar to the one for the classical spectral methods (see [41, 33, 32]) show that we can take $R = 2S$ and $T \geq (1+3\sqrt{2})S/2$ to get all collisions and prevent intersections of the regions, where f is different from zero. The (truncated) operator now reads

$$Q^R(f,f)(v) = \int_{y\in B_R}\int_{z\in B_R} \mathcal{B}^f(y,z)\,\delta(y\cdot z)\,(f(v+z)f(v+y) - f(v+y+z)f(v))\, dy\, dz \tag{3.18}$$

for $v\in\mathcal{D}_T$. This representation of the collision kernel yields better decoupling properties between the arguments of the operator. From now on we can apply the spectral algorithm (3.7), (3.8) to this collision operator, and the corresponding kernel modes are given by

$$\beta^f(l,m) = \int_{y\in B_R}\int_{z\in B_R} \tilde{B}(y,z)\,\delta(y\cdot z)\, e^{i\frac{\pi}{T}(l\cdot y + m\cdot z)}\, dy\, dz.$$

In what follows we shall focus on β^f, and one easily checks that $\beta^f(l,m)$ depends only on $|l|$, $|m|$, and $|l\cdot m|$.

3.1. Note that the classical spectral method originates the following form of the kernel modes in the y, z notation:

$$\beta^c(l,m) = \int_{y\in B_R}\int_{z\in B_R} \mathcal{B}^f(y,z)\,\delta(y\cdot z)\,\chi_{|y+z|\leq R}\, e^{i\frac{\pi}{T}(l\cdot y + m\cdot z)}\, dy\, dz.$$

One can notice that the condition $|y+z|^2 = |y|^2 + |z|^2 \leq R^2$ couples the modulus of y and z such that the ball is not completely covered (for instance, if y and z are orthogonal both with modulus R, the condition is not satisfied, since $|y+z| = \sqrt{2R}$). This explains the better decoupling properties between the arguments of the collision operator of this representation.

4. Fast algorithms. The search for fast deterministic algorithms for the collision operator, i.e., algorithms with a cost lower than $O(N^{2d+\epsilon})$ (with typically $\epsilon = 1$ for DVM, or $\epsilon = 0$ for the classical spectral method), consists mainly of identifying some convolution structure in the operator (see, for example, [8, 42]). The aim is to approximate each $\beta^f(l,m)$ by a sum

$$\beta^f(l,m) = \sum_{p=1}^{A} \alpha_p(l)\alpha'_p(m). \tag{4.1}$$

This gives a sum of A discrete convolutions, and thus the algorithm can be computed in $O(A\,N^d \log_2 N)$ operations by means of standard fast Fourier transform (FFT) techniques [11]. To this purpose we shall use a further approximated collision operator, where the number of possible directions of collision is reduced to a finite set. We start from representation (3.18) and write y and z in spherical coordinates

$$Q^R(f,f)(v) = \frac{1}{4}\int_{e\in S^{d-1}}\int_{e'\in S^{d-1}} \delta(e\cdot e')\,de\,de'$$
$$\times\left[\int_{-R}^{R}\int_{-R}^{R} \rho^{d-2}(\rho')^{d-2}\mathcal{B}^f(\rho,\rho')\left(f(v+\rho' e')\,f(v+\rho e) - f(v+\rho' e'+\rho e)f(v)\right)d\rho\,d\rho'\right]$$

(note that thanks to the orthogonality condition imposed by the Dirac mass on y and z, $\mathcal{B}^f$ depends only on the modulus of y and z). Let us denote by $\mathcal{A}$ a discrete set of orthogonal couples of unit vectors (e,e'), which is even; i.e., $(e,e')\in\mathcal{A}$ implies that $(-e,e')$, $(e,-e')$, and $(-e,-e')$ belong to $\mathcal{A}$ (this property on the set $\mathcal{A}$ is required to preserve the conservation properties of the operator). Now we define $Q^{R,\mathcal{A}}$ to be

$$Q^{R,\mathcal{A}}(f,f)(v) = \frac{1}{4}\int_{(e,e')\in\mathcal{A}} d\mathcal{A}$$
$$\times\left[\int_{-R}^{R}\int_{-R}^{R} \rho^{d-2}(\rho')^{d-2}\mathcal{B}^f(\rho,\rho')\left(f(v+\rho' e')\,f(v+\rho e) - f(v+\rho' e'+\rho e)f(v)\right)d\rho\,d\rho'\right],$$

where $d\mathcal{A}$ denotes a discrete measure on $\mathcal{A}$, which is also even in the sense that $d\mathcal{A}(e,e') = d\mathcal{A}(-e,e') = d\mathcal{A}(e,-e') = d\mathcal{A}(-e,-e')$. It is easy to check that $Q^{R,\mathcal{A}}$ has the same conservation properties as Q^R. We make the decoupling assumption that

$$\forall\, y\perp z, \quad \mathcal{B}^f(y,z) = a(|y|)\,b(|z|). \tag{4.2}$$

This assumption is obviously satisfied if $\mathcal{B}^f$ is constant. This is the case of Maxwellian molecules in dimension $d=2$ and hard spheres in dimension $d=3$ (the most relevant kernel for applications). Extensions to more general interactions are discussed in [33].

Let us describe the method in dimension $d=3$ with $\mathcal{B}^f$ satisfying the decoupling assumption (4.2) (see [33] for other dimensions). First we change to spherical coordinates

$$\beta^f(l,m) = \frac{1}{4}\int_{e\in S^2}\int_{e'\in S^2}\delta(e\cdot e')$$
$$\times\left[\int_{-R}^{R}|\rho|\,a(\rho)\,e^{i\frac{\pi}{T}\rho(l\cdot e)}\,d\rho\right]\left[\int_{-R}^{R}|\rho'|\,b(\rho')\,e^{i\frac{\pi}{T}\rho'(m\cdot e')}\,d\rho'\right]de\,de'$$

and then we integrate first e' on the intersection of the unit sphere with the plane $e^\perp$

$$\beta^f(l,m) = \frac{1}{4}\int_{e\in S^2} \phi^3_{R,a}(l\cdot e)\,\left[\int_{e'\in S^2\cap e^\perp} \phi^3_{R,b}(m\cdot e')\,de'\right]\,de,$$

where

$$\phi^3_{R,a}(s) = \int_{-R}^{R} |\rho| a(\rho) e^{i\frac{\pi}{T}\rho s}\,d\rho, \quad \phi^3_{R,b}(s) = \int_{-R}^{R} |\rho| b(\rho) e^{i\frac{\pi}{T}\rho s}\,d\rho.$$

Thus we get the following decoupling formula with two degrees of freedom:

$$\beta_R(l,m) = \int_{e\in S^2_+} \phi^3_{R,a}(l\cdot e)\,\psi^3_{R,b}(\Pi_{e^\perp}(m))\,de,$$

where S^2_+ denotes the half-sphere and

$$\psi^3_{R,b}(\Pi_{e^\perp}(m)) = \int_0^\pi \phi^3_{R,b}\big(|\Pi_{e^\perp}(m)|\,\cos\theta\big)\,d\theta,$$

and $\Pi_{e^\perp}$ is the orthogonal projection on the plane $e^\perp$. In the particular case where $\mathcal{B}^f = 1$ (HS model), we can compute the functions ϕ^3_R and ψ^3_R

$$\phi^3_R(s) = R^2\big(2\,\mathrm{Sinc}(Rs) - \mathrm{Sinc}^2(Rs/2)\big), \quad \psi^3_R(s) = \int_0^\pi \phi^3_R\big(s\,\cos\theta\big)\,d\theta.$$

Now the function $e\mapsto \phi^3_{R,a}(l\cdot e)\,\psi^3_{R,b}(\Pi_{e^\perp}(m))$ is periodic on S^+_2. Taking a spherical parametrization (θ,φ) of $e\in S^2_+$ and taking for the set $\mathcal{A}$ uniform grids of respective size M_1 and M_2 for θ and φ we get

$$\beta^f(l,m) \simeq \frac{\pi^2}{M_1\,M_2}\sum_{p,q=0}^{M_1,M_2} \alpha_{p,q}(m)\,\alpha'_{p,q}(l),$$

where

$$\alpha_{p,q}(l) = \phi^3_{R,a}(l\cdot e_{\theta_p,\varphi_q}), \quad \alpha'_{p,q}(m) = \psi^3_{R,b}\left(\Pi_{e^\perp_{\theta_p,\varphi_q}}(m)\right)$$

and $(\theta_p,\varphi_q) = (p\pi/M_1, q\pi/M_2)$.

We consider this expansion with $M = M_1 = M_2$ to avoid anisotropy in the computational grid. By using the FFT algorithm, the computational cost of the algorithm is then $O(M^2N^3\log N)$ (compared to $O(M^2N^6)$ of a direct discretization on the grid for a DVM method, and to $O(N^6)$ of the classical spectral method).

Let us finally mention that the mathematical analysis of the fast algorithm in [33] provides the following results:

- It is spectrally accurate according to the parameters N and M;
- the error on the conservation laws of momentum and energy is spectrally small according to the parameter N, and no additional error (according to the speed-up parameter M) is made.

These two properties were the main motivations for the development of the method of [33] described above to obtain the decomposition (4.1). Other advantages of this particular decomposition is that it does not introduce instability in the equation (see [33, Theorem 3.1], for instance) and it is naturally adaptive (as it is based on the rectangular quadrature rule for approximating integrals of periodic functions). Finally, another advantage of the proposed method is that it is still easy to implement since it is based only on FFT.

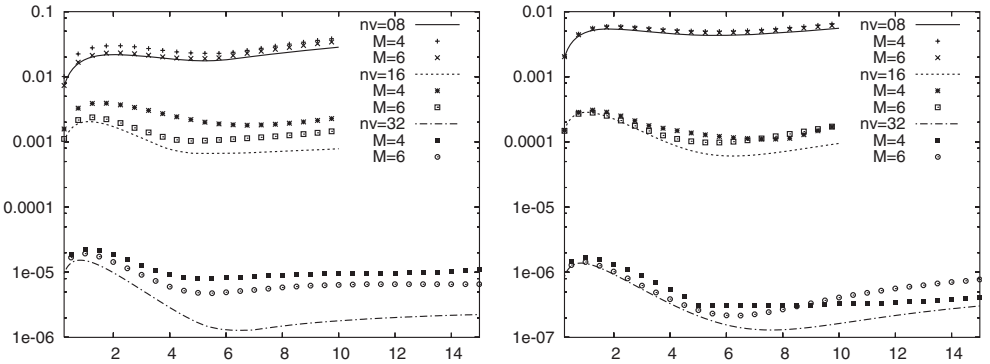


FIG. 1. *Two-dimensional homogeneous case: evolution of the numerical L^1 and L^∞ relative error of $f(t, v)$.*

5. Numerical results in the homogeneous case. In this section we will present several numerical results for the space homogeneous equation which show the improvement of the fast spectral algorithms with respect to the classical spectral methods and how they compare with Monte Carlo methods. The time discretization is performed by suitable Runge–Kutta methods.

5.1. Spatially homogeneous Maxwell molecules in dimension 2.

Comparison to exact solutions. We consider two-dimensional pseudo-Maxwell molecules (i.e., the VHS model with $\gamma = 0$). In this case we have an exact solution given by

$$f(t, v) = \frac{\exp(-v^2/2S)}{2\pi\, S^2} \left[2\, S - 1 + \frac{1 - S}{2\, S}\, v^2\right],$$

with $S = 1 - \exp(-t/8)/2$, which corresponds to the well-known exact solution [4]. This test is performed to check spectral accuracy, by comparing the error at a given time, when using $n_v = 8$, 16, and 32 Fourier modes for each coordinate. We present the results obtained by the classical spectral method and the fast spectral method with different numbers of discrete angles.

Figure 1 shows the relative L^∞, L^1, and L^2 norms of the difference between the numerical and exact solutions as functions of time. These errors are computed according to the formula

$$\mathcal{E}_p = \left(\frac{\sum_{i=-N}^{N} |f_i(t) - f(v_i, t)|^p}{\sum_{i=-N}^{N} |f(v_i, t)|^p} \right)^{1/p},$$

with $i = (i_1, i_2)$ and $N = n_v/2$ for $p = 1$ and $p = 2$. A similar expression is used for the L^∞ error. Note that the error increases initially and then decreases almost monotonically in time. After a long time the error starts increasing again. This effect is due to aliasing. Indeed, for a fixed computational domain, when the number of Fourier modes increases, the effect of aliasing becomes dominant over the error due to the spectral approximation. For this reason, the size of the domain is chosen in order to minimize the aliasing error. A trade-off should be obtained between aliasing and spectral error, which means that the size of the domain should be increased when

TABLE 1
Comparison of the L^1 error in two dimensions between the classical spectral method and the fast spectral method with different numbers of discrete angles and with a second-order Runge–Kutta time discretization at time $T_{end} = 1$.

Number of points	Classical spectral	Fast spectral with $M = 4$	Fast spectral with $M = 6$	Fast spectral with $M = 8$
8	0.02013	0.02778	0.02129	0.02112
16	0.00204	0.00329	0.00238	0.00224
32	1.405E-5	2.228E-5	1.861E-5	1.772E-5

increasing the number of Fourier modes. Roughly speaking, the period should be chosen in such a way that the two contributions of the error are of the same order of magnitude. In this test, the radius of the ball, which defines the computational domain, is $T = 4$ for $n_v = 8$, $T = 5$ for $n_v = 16$, and $T = 7$ for $n_v = 32$. We refer to [41] for a more detailed discussion about aliasing.

Concerning the comparison between the classical and fast spectral methods, we observe that for a fixed value of n_v, the numerical error of the classical spectral method and of the fast algorithm is of the same order. Moreover, the influence of the number of discrete angles is very weak. Indeed, with only $M = 4$, the results are quite similar even for large n_v and as expected the number of discrete angles does not affect the variations of energy, which are of the same order of magnitude as the numerical error (note that there is no variation for the momentum since in the special case of even solutions, it is preserved at 0 by the spectral scheme). In Table 1, we give a quantitative comparison of the numerical error $\mathcal{E}_1$ at time $T_{end} = 1$. We can also observe the spectral accuracy for the classical and fast methods: the order of accuracy is about 3 between 8 and 16 grid points, whereas it becomes 7 between 16 to 32 points.

Efficiency and accuracy. Now we still consider two-dimensional pseudo-Maxwell molecules (i.e., $\gamma = 0$) with the initial datum

$$f(0,v) = \frac{1}{4\,\pi}\left[\exp\left(-\frac{|v-v_0|^2}{2}\right) + \exp\left(-\frac{|v+v_0|^2}{2}\right)\right], \quad v \in \mathbb{R}^2,$$

where $v_0 = (1,2)$. In this case, we do not know the exact solution but we want to study the influence of the number of discrete angles on a nonisotropic solution. Thus, this test is used to check the energy conservation and the evolution of high-order moments of the solution.

The time step is chosen small enough to reduce the influence of the time discretization, i.e., $\Delta t = 0.025$. Moreover, the computational domain is taken large enough with respect to the number of grid points in order to reduce the aliasing error due to the periodization of the solution. Simulations are performed with $n_v = 16$, 32, and 64 points.

In Figure 2 the relaxation of the entropy and the temperature components for the fast and classical spectral methods is shown. The energy is conserved by the continuous collision operator, but when using the spectral method the total energy can change with time, which is a good indicator of the accuracy of the numerical solution. Indeed, the total discrete energy is not exactly conserved over time, but if the aliasing error is small, it is conserved within spectral accuracy; typically here the variations are about 10^{-4} when $n_v = 32$. Moreover, we observe that the number of discrete angles does not significantly affect the transient regime. For instance, with only four angles

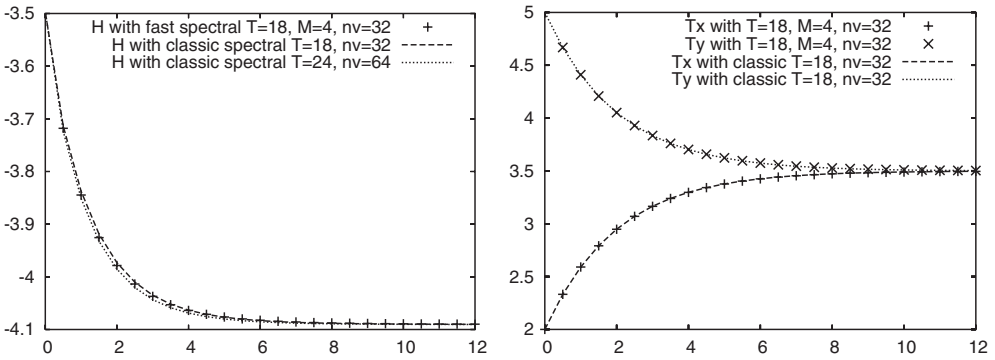


FIG. 2. *Two-dimensional homogeneous case: relaxation of the entropy and the temperature components for the fast and classical spectral methods with respect to the number of modes per direction n_v and the length box T.*

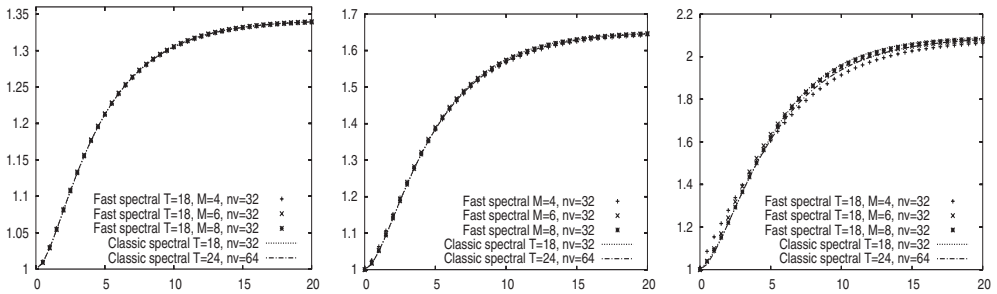


FIG. 3. *Two-dimensional homogeneous case: time evolution of the variations of high-order normalized moments $\mathcal{M}_4$, $\mathcal{M}_5$, and $\mathcal{M}_6$ of $f(t,v)$ for the fast and classical spectral methods with respect to the number of modes per direction n_v and the length box T.*

on the half-sphere, the relaxation of entropy and temperature components is very close to the numerical solution obtained by the classical spectral method. Finally, we plot in Figure 3 the time evolution of high-order moments of $f_N(t,v)$ given in discrete form by

$$\mathcal{M}_k(t) = \Delta v^2 \sum_{l=-N}^{N} |v_l|^k \, f_N(t, v_l).$$

High-order moments give information on the accuracy of the approximate distribution function tail. Once again, we observe that the number of angles does not affect the results even if the solution is nonisotropic.

To conclude, we observe that in dimension $d = 2$, the fast algorithm is really efficient in terms of accuracy and computational cost compared to the classical spectral method. In Table 2 we report the computational times of the methods which show a speed-up of the fast solver independently of the number of points used in our tests and with a maximum speed-up reached for $N = 64$, where the fast methods with $M = 4$ is more than 17 times faster than the classical method.

5.2. Spatially homogeneous hard spheres in dimension 3. In this section we consider the three-dimensional HS molecules model. The initial condition is chosen

TABLE 2
Comparison of the computational time in two dimensions between the classical spectral method and the fast spectral method with different numbers of discrete angles and with a second-order Runge–Kutta time discretization.

Number of points	Classical spectral	Fast spectral with $M = 4$	Fast spectral with $M = 6$	Fast spectral with $M = 8$
16	2 sec. 40	1 sec. 15	1 sec. 70	2 sec. 30
32	38 sec. 01	5 sec. 55	8 sec. 47	11 sec. 10
64	616 sec.	35 sec. 50	54 sec. 66	71 sec. 27

TABLE 3
Comparison of the computational time in three dimensions between the classical spectral method and the fast spectral method with different numbers of discrete angles and with a second-order Runge–Kutta time discretization.

Number of points	Classical spectral	Fast spectral with $M = 4$	Fast spectral with $M = 6$	Fast spectral with $M = 8$
16	1 min. 14 sec.	3 min. 31 sec.	7 min. 45 sec.	13 min. 44 sec.
32	118 min. 02 sec.	50 min. 31 sec.	105 min. 19 sec.	186 min. 18 sec.
64	125 h 54 min.	8 h 45 min. 22 sec.	21 h 39 min.	35 h 01 min. 28 sec.

as the sum of two Gaussians

$$f(v,0) = \frac{1}{2(2\pi\sigma^2)}\left[\exp\left(-\frac{|v-v_0|^2}{2\sigma^2}\right) + \exp\left(-\frac{|v+v_0|^2}{2\sigma^2}\right)\right],$$

with $\sigma = 1$ and $v_0 = (2,1,0)$. The final time of the simulation is $T_{end} = 3$ and corresponds approximately to the time for which the steady state of the solution is reached. The time step is $\Delta t = 0.1$ and the length box is taken as $T = 12$ when $n_v = 16$ and as $T = 15$ when $n_v = 32$.

This test is used to check the evolution of moments and particularly the stress tensor $P_{i,j}$, $i,j = 1,\dots,3$ defined as

$$P_{i,j} = \int_{\mathbb{R}^3} f(v)(v_i - u_i)\,(v_j - u_j)\,dv, \quad (i,j) \in \{1,2,3\}^2,$$

where $(u_i)_i$ are the components of the mean velocity. As in the previous case, we compare the classical and fast methods in terms of computational time (see Table 3) and accuracy. In Figure 4, we propose the evolution of the temperature for the two methods using 32 grid points in each direction. This solution is also compared with the solution obtained from the Monte Carlo method. The discrete temperatures agree well in this case and the efficiency of the fast algorithm is verified since the computational time is highly reduced using only $M_1 = M_2 = 4$ discrete angles without affecting the accuracy of the distribution function. We remark that in dimension $d = 3$ the speed-up of the methods becomes really evident for large values of N. Again for $N = 64$ and $M = 4$ the fast method is more than 14 times faster.

Comparison with Monte Carlo. Finally, we compare the results obtained with our spectral method with those obtained by a Monte Carlo scheme. We use a standard version of Monte Carlo, which may be referred to as the Nanbu–Babovsky scheme [36, 2].

In the case of Monte Carlo methods, the moments are computed by using unbiased estimators averaged over several runs. The number of runs N_{runs}, number of particles N_p, and the time step Δt, have been chosen in such a way to balance the time discretization error with statistical fluctuations.

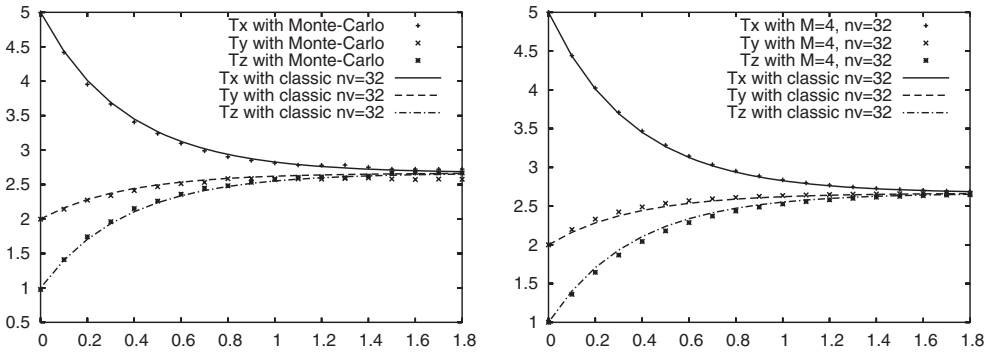


FIG. 4. *Three-dimensional homogeneous case: comparison between the fast and classical spectral methods and the Monte Carlo method for the temperature components relaxation.*

The first run consists of taking a large number of particles and making time averaging in order to minimize the fluctuation errors: $N_{\text{runs}} = 10^3$, $N_p = 10^4$, and $\Delta t = 0.01$. The total computational time for computing the evolution of moments in this case is 113 min. 23 sec. On the other hand we perform a second run, where we use more averaging to minimize fluctuation errors: $N_{\text{runs}} = 5.\,10^3$, $N_p = 5.\,10^3$, and $\Delta t = 0.01$. The computational time is now 100 min.

We remark that the computations have been obtained by using the HS model (VHS with $\gamma = 1$), which is the most realistic. In this case, the computational time of the Monte Carlo methods becomes larger than in the case when we consider pseudo-Maxwell molecules (VHS with $\gamma = 0$) for which the collision kernel is constant and no rejection is needed. For the spectral method the computational cost is independent of the collision kernel.

From the comparison, it is obvious that, for three-dimensional computations, the greater cost of the classical spectral scheme (with the same number of degrees of freedom) is compensated by a much greater accuracy, allowing for better results with the same computational cost. Moreover, with the fast algorithm, the spectral scheme really becomes competitive in terms of computational time since the accuracy is not affected when we use fewer discrete angles (for instance, $M = 4$). In Figure 5, we compare the accuracy on the evolution of high-order moments with the different methods: the fourth-order moments are very close, but the results obtained with the Monte Carlo methods are affected by fluctuations on the tail of the distribution function, which are difficult to remove (see the evolution of the eighth-order moment). Note that for two- and three-dimensional pseudo-Maxwell molecules, comparisons had also been performed between Monte Carlo methods and the classical spectral method in [41, section 6.3].

5.3. Stability of spectral methods with respect to nonsmooth data. In this subsection we perform some numerical simulations in order to study the behavior of the spectral methods when applied to nonsmooth data. We consider the distribution

$$f_0(v) = \begin{cases} 1 & \text{if } |v|^2 \leq 1, \\ 0 & \text{else} \end{cases}$$

and use a mollified initial datum, which suitably approximates moments. We perform two numerical simulations using the fast spectral method with 32^2 and 64^2 grid points and plot the evolution of the entropy and the fourth-order moment in Figure 6. Even

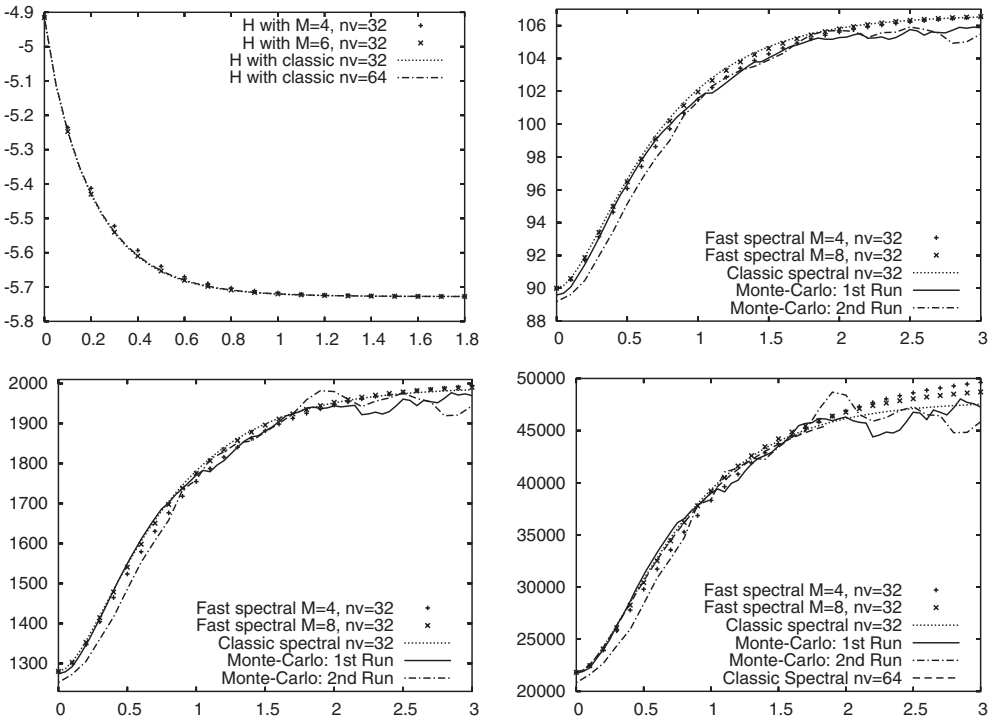


FIG. 5. *Three-dimensional homogeneous case: time evolution of the kinetic entropy H and high-order moments $\mathcal{M}_4$, $\mathcal{M}_6$, and $\mathcal{M}_8$ of $f(t,v)$ for the fast and classical spectral methods and the Monte Carlo methods.*

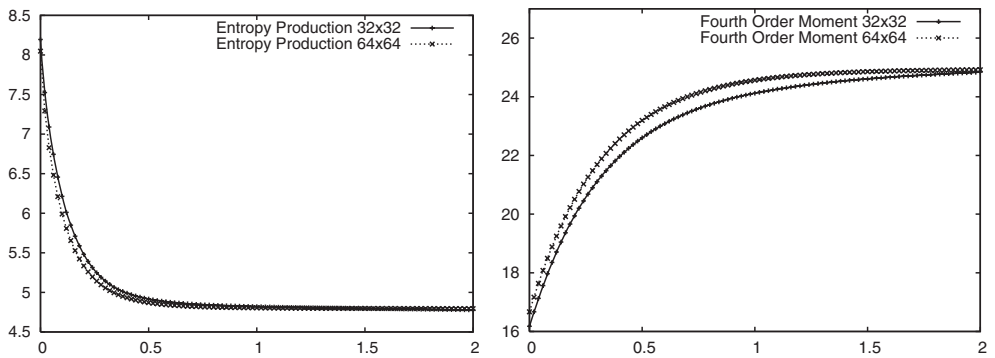


FIG. 6. *Time evolution of the kinetic entropy and the fourth-order moment with respect to the number of grid points 32^2 and 64^2.*

for this discontinuous initial datum, we observe that for the two configurations the numerical entropy is decreasing and both numerical solutions converge to the same steady state. Moreover, we plot the evolution of the distribution function with respect to time in Figure 7: the fast spectral method is very stable for this numerical test even if spurious oscillations are first generated; the distribution becomes smooth and converges to an approximated Maxwellian. As expected from a Fourier–Galerkin method, the accuracy degenerates in the discontinuity region. However, surprisingly the method seems to remain stable.

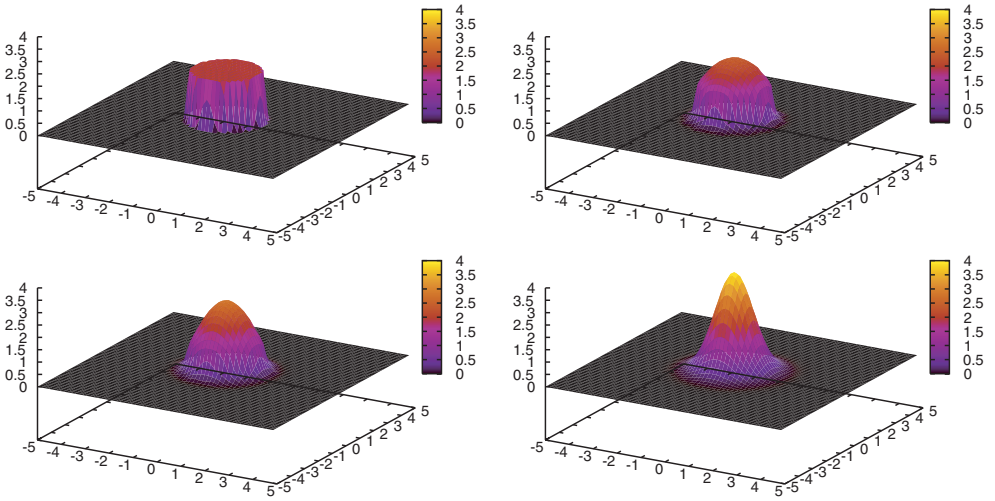


FIG. 7. *Time evolution of the distribution function for* 64×64 *grid points.*

Finally, let us mention that we have also performed some numerical tests when the initial datum approximates a Dirac distribution. The spectral method is still stable even if spurious oscillations are generated, but this problem is inherent to gridded methods and we refer to [23] for a rescaling method, which follows the variation of the distribution function and allows us to treat concentrated distributions.

6. Application to the nonhomogeneous case. In [20, 22], we performed several numerical simulations to compare the spectral scheme with Monte Carlo methods and showed that when we are interested in the transient regime, the deterministic method becomes very efficient. Obviously, the fast spectral method will still improve the computational cost.

In what follows, since we will be interested in the study of the trend to a global equilibrium state of the kinetic equation, we avoid the use of a splitting method by solving the whole nonhomogeneous equation in time by a second-order Runge–Kutta method. Clearly the spectral methods apply straightforwardly to the collision operator also in this situation. The transport part is treated by the positive flux conservative method (see [18, 21, 20] for further details).

6.1. Definition of the problem. We consider the full Boltzmann equation in dimension $d = 2$ on the torus

$$\frac{\partial f}{\partial t} + v \cdot \nabla_x f = Q(f, f), \qquad x \in [0, L]^2, \quad v \in \mathbb{R}^2,$$

with periodic boundary conditions in x. We first introduce the hydrodynamical fields associated with a kinetic distribution $f(t, x, v)$. These are the $(d + 2)$ scalar fields of density ρ (scalar), mean velocity u (vector valued), and temperature T (scalar) defined by the formulas (2.7). Whenever $f(t, x, v)$ is a smooth solution to the Boltzmann equation with periodic boundary conditions, one has the global conservation laws for

mass, momentum, and energy

$$\frac{d}{dt}\int_{[0,L]^2\times\mathbb{R}^2} f(t,x,v)\,dx\,dv = 0,$$
$$\frac{d}{dt}\int_{[0,L]^2\times\mathbb{R}^2} f(t,x,v)\,v\,dx\,dv = 0,$$
$$\frac{d}{dt}\int_{[0,L]^2\times\mathbb{R}^2} f(t,x,v)\,\frac{|v|^2}{2}\,dx\,dv = 0.$$

Therefore, without loss of generality we shall impose

$$\int_{[0,L]^2\times\mathbb{R}^2} f(t,x,v)\,dx\,dv = 1, \quad \int_{[0,L]^2\times\mathbb{R}^2} f(t,x,v)\,v\,dx\,dv = 0,$$

and

$$\int_{[0,L]^2\times\mathbb{R}^2} f(t,x,v)\,\frac{|v|^2}{2}\,dx\,dv = 1.$$

These conservation laws are then enough to uniquely determine the stationary state of the Boltzmann equation: the normalized global Maxwellian distribution

$$M_g(v) = \frac{1}{2\pi}\exp\left(-\frac{|v|^2}{2}\right). \tag{6.1}$$

We shall use the following terminology: a velocity distribution of the form (6.1) will be called a [illegible], whereas a distribution of the form

$$M_l(x,v) = \frac{\rho(x)}{2\pi T(x)}\exp\left(-\frac{|v-u(x)|^2}{2T(x)}\right) \tag{6.2}$$

will be called a [illegible] (in the sense that the constants ρ, u, and T appearing there depend on the position x). We also define the notion of [illegible] H_l, the entropy relative to the local Maxwellian, and the [illegible] H_g, the entropy relative to the global Maxwellian distribution, respectively, by

$$H_l(t) = \int f\,\log\left(\frac{f}{M_l}\right)\,dx\,dv, \quad H_g(t) = \int f\,\log\left(\frac{f}{M_g}\right)\,dx\,dv.$$

Our goal here is to investigate numerically the long-time behavior of the solution f. If f is any reasonable solution of the Boltzmann equation, satisfying certain a priori bounds of compactness (in particular, ensuring that no kinetic energy is allowed to leak at large velocities), then it is possible to prove that f does indeed converge to the global Maxwellian distribution M_g as t goes to $+\infty$. Of course, obtaining these a priori bounds is extremely difficult; as a matter of fact, they have been established only in the spatially homogeneous situation (which means that the distribution function does not depend on the position variable x; see the survey in [49]) or in a close-to-equilibrium setting (see in particular [25] for the torus), and it still constitutes a famous open problem for spatially inhomogeneous initial data far from equilibrium. More recently, Desvillettes and Villani [16] and Strain and Guo [47] were interested in the study of rates of convergence for the full Boltzmann equation. Roughly speaking,

in [16] it was proved that if the solution to the Boltzmann equation is smooth enough and satisfies bounds from below of the form

$$\forall\, t \geq 0,\ x \in [0, L]^2,\ v \in \mathbb{R}^2, \quad f(t, x, v) \geq K_0\, e^{-A_0\, |v|^{q_0}} \quad (A_0, K_0 > 0,\ q_0 \geq 2),$$

(although this bound can be shown to be a consequence of the regularity bounds; see [31]), then (with constructive bounds)

$$\|f(t) - M_g\| = O(t^{-\infty}),$$

which means that the solution converges almost exponentially fast to the global equilibrium (namely, with polynomial rate $O(t^{-r})$ with r as large as desired).

The solution f to the Boltzmann equation satisfies the formula of additivity of the entropy: the entropy can be decomposed into the sum of a purely hydrodynamic part and (by contrast) of a purely kinetic part. In terms of the H functional, one can write

$$H_g(t) = H_l(t) + \int_0^L \rho_l(t, x)\, \log\left(\frac{\rho_l(t, x)}{T_l(t, x)}\right) dx.$$

In fact, we can also show that

$$H_l(t) \leq H_g(t) \quad \forall t \geq 0.$$

Moreover, the Csiszár–Kullback–Pinsker inequality asserts that (when the total mass of the solution is normalized to 1)

$$H(f|M) \geq \frac{1}{2}\|f - M\|_{L^1}^2.$$

In other words, controlling the speed of convergence of the entropy to its equilibrium value is enough to control the speed of convergence of the solution to its equilibrium in a very strong sense.

Moreover, in [16] Desvillettes and Villani conjectured that time oscillations should occur on the evolution of the relative local entropy. In fact their proof does not rule out the possibility that the entropy production undergoes important oscillations in time, and actually most of the technical work is caused by this possibility.

6.2. Description and interpretation of the results. Here we performed simulations on the full Boltzmann equation in a simplified geometry (one dimension of space, two dimensions of velocity, periodic boundary conditions, fixed Knudsen number) with the fast spectral method to observe the evolution of the entropy and to check numerically if such oscillations occur. Clearly this test is challenging for a numerical method due to the high accuracy required to capture such oscillating behavior.

Then we considered an initial datum as a perturbation of the global equilibrium M_g

$$f_0(x, v) = \frac{1}{2\pi}\,(1 + A_0\, \cos(k_0\, x))\, \exp(-|v|^2/2), \qquad x \in [0, L], \quad v \in \mathbb{R}^2, \tag{6.3}$$

for some constants $A_0 > 0$ and $k_0 = 2\pi/L$.

In Figures 8 and 9, we are indeed able to observe oscillations in the entropy production and in the hydrodynamic entropy. The strength of the oscillations depends a lot on the length L of the domain, which is consistent with the fact that

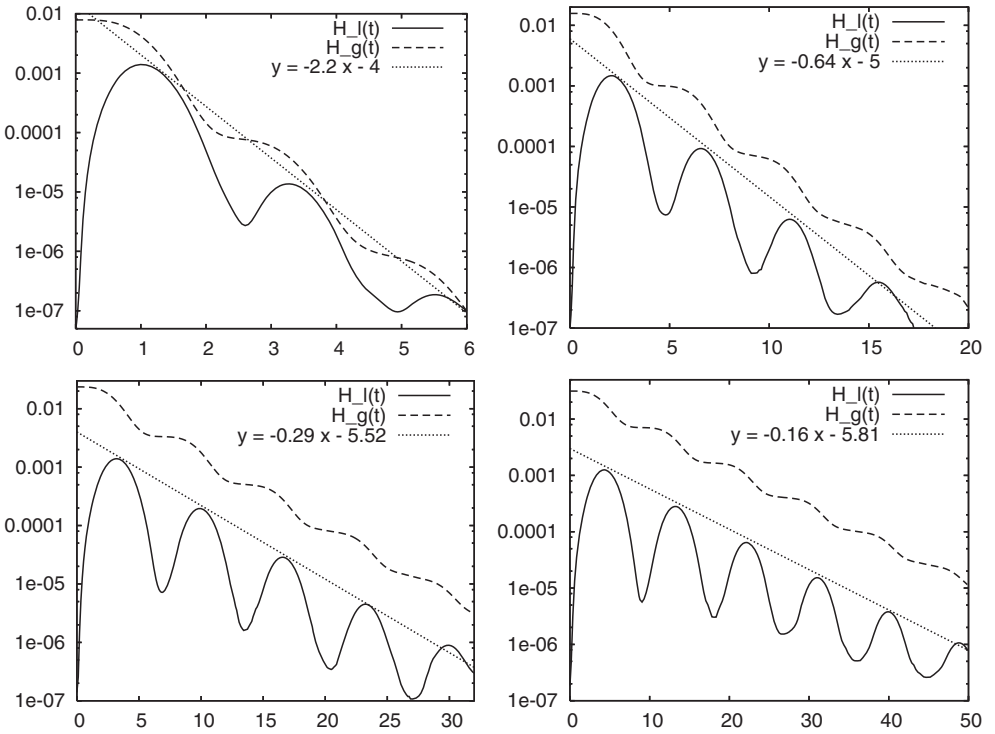


FIG. 8. *Influence of the length box: relative entropy with respect to the local Maxwellian* $H_l(t)$ *using* $64 \times 64 \times 64$ *for* $L = \pi$, 2π, 3π, *and* 4π *with* $A_0 = 0.1$ *in* (6.3).

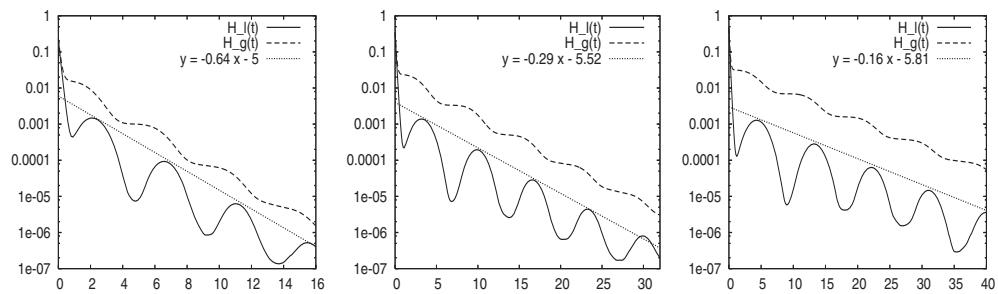


FIG. 9. *Influence of the length box: relative entropy with respect to the local Maxwellian* $H_l(t)$ *using* $64 \times 64 \times 64$ *for* $L = 2\pi$, 3π, *and* 4π *with* $A_0 = 0.1$ *in* (6.4).

such oscillations are never observed in the spatially homogeneous case ($L = 0$). The superimposed curves yield the time evolution of the total H functional and of its kinetic part, respectively. In all cases, a local Maxwellian distribution is chosen for the initial datum; the first plot corresponds to $L = 1$ and the second one to $L = 4$. Some slight oscillations can be seen in the case $L = 1$, but what is most striking is that after a short while, the kinetic entropy is very close to the total entropy, which is an indication that the solution evolves basically in a spatially homogeneous way (contrary to the intuition of the hydrodynamic regime). On the contrary, in the case $L = 4$, the oscillations are much more important in frequency and amplitude (note that this is a logarithmic plot): the solution "hesitates" between states in which it

TABLE 4
Influence of the length box: damping rate and oscillation frequency for the relative entropy with respect to the local Maxwellian $H_l(t)$ using $64 \times 64 \times 64$ with $A_0 = 0.1$ in (6.3).

Length box	Oscillation frequency ω	ω/L	Damping rate α	$-\alpha L^2$
$L = \pi/2$	01.10	0.701	-6.521	16.04
$L = \pi$	02.25	0.716	-2.202	21.71
$L = 2\pi$	04.50	0.716	-0.641	25.26
$L = 3\pi$	06.61	0.701	-0.285	25.31
$L = 4\pi$	08.78	0.699	-0.160	25.27
$L = 8\pi$	17.57	0.699	-0.040	25.35

is very close to hydrodynamic and states in which it is not at all. Further note that the equilibration is much more rapid when the box is small and that the convergence seems to be exponential.

It is in fact possible to give a simple interpretation of these oscillations thanks to the work [17]. Since this effect is observed near the global equilibrium one can replace the Boltzmann collision operator with the linearized Boltzmann collision operator (moreover, oscillation is effectively observed for the linearized Boltzmann collision operator as well). Then it is straightforward that the computations above correspond to observing the time evolution of one Fourier mode in x (here with frequency k_0 and amplitude A_0). Hence by an obvious rescaling, this evolution is given by the semigroup $\tilde{T}_{k_0/L}(t)$, where $\tilde{T}_k(t)$ is defined in [17] (this is the semigroup for the $\cdot$th Fourier mode in x for the linearized equation). An asymptotic study of the spectrum of its infinitesimal generator for small frequencies k was done in [17]. The dominant term in terms of long-time behavior (i.e., the one with the lower rate of decrease) is given by the $(d+2)$ "hydrodynamical eigenvalues." Moreover, explicit computations are available for the expansions of these eigenvalues according to $\varepsilon = |k|$ near $k = 0$.

At first order in ε, the eigenvalues vanish, except for two of them, which are purely imaginary. They are given by

$$\begin{cases} I_1 = i\,\varepsilon\,\sqrt{1+2/d} + O(\varepsilon^2), \\ I_2 = -i\,\varepsilon\,\sqrt{1+2/d} + O(\varepsilon^2), \\ I_3 = \cdots = I_{d+2} = 0. \end{cases}$$

Therefore for $|k_0|/L << 1$ (realized, for instance, when k_0 is fixed and L is large enough), this analysis gives us the dominant imaginary term in the eigenvalues. In this regime, one should thus observe oscillations with frequency $\sqrt{1+2/d}\,|k_0|/L$. Thus, the period of oscillations should be given by $2\pi(1+2/d)^{-1/2}\,L/|k_0|$, which can be checked with the numerical simulations. Indeed, in Table 4 we give the ratio of the period of oscillations ω with the length box L. The numerical results agree well with the analytical computations $\omega/L \simeq 1/\sqrt{2}$.

We also observe that the damping rate is related to the length box and is proportional to $1/L^2$ when L becomes large (see Table 4; $\alpha\,L^2 \simeq$ constant). This is coherent with the fact that no real value occurs in the hydrodynamical eigenvalues until the second order in $\varepsilon = |k|$. The coefficients for the order 2 in the expansion are computed in [17]; they are purely real and can be expressed simply in terms of the dimension d, the viscosity coefficient η, and the heat conductivity λ of the gas (indeed, these coefficients are related to the Navier–Stokes limit of the Boltzmann

equation). Namely, they are given by

$$\begin{cases} R_1 = R_2 = -\dfrac{\lambda}{d+2} - \dfrac{\eta}{2}, \\ R_3 = \cdots = R_{d+1} = -\dfrac{\eta\, d}{2(d-1)}, \\ R_{d+2} = -\dfrac{\lambda\, d}{d+2}. \end{cases}$$

Therefore for $|k_0|/L << 1$, the damping rate is given by the minimum among these values.

To conclude these tests, we performed a final numerical experiment to evaluate the robustness of the theory of the trend to equilibrium. We have chosen an initial datum which is far from the equilibrium

(6.4)
$$f_0(x,v) = \frac{1}{2\pi v_{th}^2}\,(1 + A_0\,\cos(k_0\,x))\,\left[\exp\left(-\frac{|v-v_0|^2}{2v_{th}^2}\right) + \exp\left(-\frac{|v+v_0|^2}{2v_{th}^2}\right)\right],$$

with $v_0 = (1/2, 1/2)$ and $v_{th} = \sqrt{3}/2$. We present the time evolution of the relative entropies H_l and H_g in log scale and observe that initially the entropy is strongly decreasing, and when the distribution function becomes close to a local equilibrium, some oscillations appear with the good frequency $\omega/L = 1/\sqrt{2}$ and damping rate α (see Figure 9).

1. Now numerical methods for the Boltzmann equation, such as the one presented in this paper, become able to provide very accurate simulations of the transient regime towards equilibrium with reasonable cost, even in the inhomogeneous case. This could be used to explore the spectrum of the linearized Boltzmann collision operator (in the homogeneous case), or, more interestingly, the spectrum of the linearized Boltzmann collision operator together with the transport term in the inhomogeneous case. For instance, the exponential rate of convergence is directly readable on the figures above and provides a numerical estimation of the (that is, the real part of the first nonzero eigenvalue) of this operator (it has been known since Ukai [48] that this operator has a spectral gap in the torus; see also [13]). Moreover, by a frequency analysis of the curve of the time evolution of the relative entropy or the L^1 distance to the equilibrium, it could be possible also to describe other eigenvalues; as long as they have a different imaginary part, it should be possible (in principle) to extract from the frequency analysis the curve corresponding to their contribution in the evolution semigroup, and then to compute their real part which corresponds to the exponential rate of decay of this curve.

2. A recent work [27] gave a detailed pointwise study of the Green function for the linearized Boltzmann equation in the domain $x \in \Omega = \mathbb{R}$. This particular study shows that the long-time behavior is governed by "fluid-like waves" (corresponding to the waves of the linearized Euler and Navier–Stokes equations) whose amplitude decreases polynomially, whereas the amplitude of the "kinetic part" of the Green function decreases exponentially. We think it is likely that this study could be extended to the torus, where the amplitude of both the fluid and the kinetic parts of the Green function should decrease exponentially. Moreover, the rate of decay of the kinetic part should not depend on the size of the box, whereas the rate of decay of the

fluid part should. Hence for a box small enough, the long-time behavior should be governed by the kinetic part of the Green function (that is, like the spatially homogeneous Boltzmann equation), whereas for a box big enough, the long-time behavior should be governed by the fluid-like waves. This is precisely what we observe numerically, and thus this theoretical study could provide a rigorous proof of the numerical observations above, at least in the linearized regime.

7. Conclusions. In this paper we have introduced and deeply tested a class of new fast algorithms for the computation of the Boltzmann collision operator. These methods allow us to reduce the computational cost from $O(n^2)$ to $O(n \log_2 n)$. We give computational evidence of the great performance of the schemes which can provide a dramatic speed-up in computing time of deterministic schemes by making them competitive with Monte Carlo methods, where higher accuracy is required. A first numerical application to a nontrivial problem in the space nonhomogeneous case confirms the strong computing potential of the new schemes.

Other methods such as singular value decomposition, fast multipole methods [9], separated representations in high-dimensional problems (see, for instance, [1]), wavelets, etc. could have been used to search for a decomposition of the form (4.1). However, to our knowledge it is not known at present how to obtain the properties described above for our decomposition with these methods.

Acknowledgment. We would like to thank Cédric Villani for suggesting the numerical study of possible oscillations in the relaxation to equilibrium.

REFERENCES

[1] B. Alpert, G. Beylkin, R. Coifman, and V. Rokhlin, *Wavelet-like bases for the fast solution of second-kind integral equations*, SIAM J. Sci. Comput., 14 (1993), pp. 159–184.
[2] H. Babovsky, *On a simulation scheme for the Boltzmann equation*, Math. Methods Appl. Sci., 8 (1986), pp. 223–233.
[3] G. A. Bird, *Molecular Gas Dynamics*, Clarendon Press, Oxford, UK, 1994.
[4] A. V. Bobylev, *Exact solutions of the Boltzmann equation*, Dokl. Akad. Nauk. SSSR, 225 (1975), pp. 1296–1299 (in Russian).
[5] A. V. Bobylev, *The theory of the nonlinear spatially uniform Boltzmann equation for Maxwell molecules*, Math. Phys. Reviews, 7 (1988), pp. 111–233.
[6] A. V. Bobylev, A. Palczewski, and J. Schneider, *On approximation of the Boltzmann equation by discrete velocity models*, C. R. Acad. Sci. Paris Sér. I Math., 320 (1995), pp. 639–644.
[7] A. V. Bobylev and S. Rjasanow, *Difference scheme for the Boltzmann equation based on the fast Fourier transform*, Eur. J. Mech. B Fluids, 16 (1997), pp. 293–306.
[8] A. V. Bobylev and S. Rjasanow, *Fast deterministic method of solving the Boltzmann equation for hard spheres*, Eur. J. Mech. B Fluids, 18 (1999), pp. 869–887.
[9] O. Bokanowski and M. Lemou, *Fast multipole method for multivariables integrals*, SIAM J. Numer. Anal., 42 (2005), pp. 2098–2117.
[10] C. Buet, *A discrete velocity scheme for the Boltzmann operator of rarefied gas dynamics*, Transp. Theory Statist. Phys., 25 (1996), pp. 33–60.
[11] C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang, *Spectral Methods in Fluid Dynamics*, Springer Ser. Comput. Phys., Springer-Verlag, New York, 1988.
[12] T. Carleman, *Sur la théorie de l'équation intégrodifférentielle de Boltzmann*, Acta Math., 60 (1933), pp. 91–146.
[13] C. Cercignani, R. Illner, and M. Pulvirenti, *The Mathematical Theory of Dilute Gases*, Appl. Math. Sci. 106, Springer-Verlag, New York, 1994.
[14] A. J. Chorin, *Numerical solution of Boltzmann's equation*, Comm. Pure Appl. Math., 25 (1972), pp. 171–186.
[15] P. Degond, L. Pareschi, and G. Russo, eds., *Modeling and Computational Methods for Kinetic Equations*, Model. Simul. Sci. Eng. Technol. Birkhäuser Boston, Boston, MA, 2004.

[16] L. Desvillettes and C. Villani, *On the trend to global equilibrium for spatially inhomogeneous kinetic systems: The Boltzmann equation*, Invent. Math., 159 (2005), pp. 245–316.
[17] R. S. Ellis and M. A. Pinsky, *The first and second fluid approximations to the linearized Boltzmann equation*, J. Math. Pures Appl., 54 (1975), pp. 125–156.
[18] F. Filbet, E. Sonnendrücker, and P. Bertrand, *Conservative numerical schemes for the Vlasov equation*, J. Comput. Phys., 172 (2001), pp. 166–187.
[19] F. Filbet and L. Pareschi, *A numerical method for the accurate solution of the Fokker-Planck-Landau equation in the nonhomogeneous case*, J. Comput. Phys., 179 (2002), pp. 1–26.
[20] F. Filbet and G. Russo, *High order numerical methods for the space non-homogeneous Boltzmann equation*, J. Comput. Phys., 186 (2003), pp. 457–480.
[21] F. Filbet, L. Pareschi, and G. Toscani, *Accurate numerical methods for the collisional motion of (heated) granular flows*, J. Comput. Phys., 202 (2005), pp. 216–235.
[22] F. Filbet and G. Russo, *Accurate numerical methods for the Boltzmann equation*, in Modeling and Computational Methods for Kinetic Equations, Model Sumil. Sci Eng. Technol., Birkhäuser Boston, Boston, MA, 2004, pp. 117–145.
[23] F. Filbet and G. Russo, *A rescaling velocity method for kinetic equations: The homogeneous case*, in Proceedings of the Workshop on Modelling and Numerics of Kinetic Dissipative Systems (Lipari, Italy), L. Pareschi, G. Russo, and G. Toscani, eds., Nova Science Publishers, Inc., Hauppauge, NY, 2006.
[24] E. Gabetta, L. Pareschi, and G. Toscani, *Relaxation schemes for nonlinear kinetic equations*, SIAM J. Numer. Anal., 34 (1997), pp. 2168–2194.
[25] Y. Guo, *The Vlasov-Maxwell-Boltzmann system near Maxwellians*, Invent. Math., 153 (2003), pp. 593–630.
[26] I. Ibragimov and S. Rjasanow, *Numerical solution of the Boltzmann equation on the uniform grid*, Computing, 69 (2002), pp. 163–186.
[27] T.-P. Liu and S.-H. Yu, *The Green's function and large-time behavior of solutions for the one-dimensional Boltzmann equation*, Comm. Pure Appl. Math., 57 (2004), pp. 1543–1608.
[28] P. A. Markowich and L. Pareschi, *Fast conservative and entropic methods for the Boson Boltzmann equation*, Numer. Math., 99 (2005), pp. 509–532.
[29] Y.-L. Martin, F. Rogier, and J. Schneider, *Une méthode déterministe pour la résolution de l'équation de Boltzmann inhomogène*, C. R. Acad. Sci. Paris Sér. I Math., 314 (1992), pp. 483–487.
[30] S. Mischler and C. Mouhot, *Cooling process for inelastic Boltzmann equations for hard spheres, Part* II: *Self-similar solutions and tail behavior*, J. Statist. Phys., to appear.
[31] C. Mouhot, *Quantitative lower bounds for the full Boltzmann equation, Part* I: *Periodic boundary conditions*, Comm. Partial Differential Equations, 30 (2005), pp. 881–917.
[32] C. Mouhot and L. Pareschi, *Fast methods for the Boltzmann collision operator*, C. R. Math. Acad. Sci. Paris, 339 (2004), pp. 71–76.
[33] C. Mouhot and L. Pareschi, *Fast algorithms for computing the Boltzmann collision operator* Math. Comp., to appear.
[34] C. Mouhot and L. Pareschi, *An* $O(N \log_2 N)$ *Algorithm for Computing Discrete Velocity Models*, in preparation.
[35] G. Naldi, L. Pareschi, and G. Toscani, *Spectral methods for one-dimensional kinetic models of granular flows and numerical quasi elastic limit*, M2AN Math. Model. Numer. Anal., 37 (2003), pp. 73–90.
[36] K. Nanbu, *Direct simulation scheme derived from the Boltzmann equation.* I. *Monocomponent Gases*, J. Phys. Soc. Japan, 52 (1083), pp. 2042–2049.
[37] A. Palczewski, J. Schneider, and A. V. Bobylev, *A consistency result for a discrete-velocity model of the Boltzmann equation*, SIAM J. Numer. Anal., 34 (1997), pp. 1865–1883.
[38] A. Palczewski and J. Schneider, *Existence, stability, and convergence of solutions of discrete velocity models to the Boltzmann equation*, J. Statist. Phys., 91 (1998), pp. 307–326.
[39] V. A. Panferov and A. G. Heintz, *A new consistent discrete-velocity model for the Boltzmann equation*, Math. Methods Appl. Sci., 25 (2002), pp. 571–593.
[40] L. Pareschi and B. Perthame, *A Fourier spectral method for homogeneous Boltzmann equations*, Transp. Theory Statist. Phys., 25 (1996), pp. 369–382.
[41] L. Pareschi and G. Russo, *Numerical solution of the Boltzmann equation* I. *Spectrally accurate approximation of the collision operator*, SIAM J. Numer. Anal., 37 (2000), pp. 1217–1245.
[42] L. Pareschi, G. Russo, and G. Toscani, *Fast spectral methods for the Fokker-Planck-Landau collision operator*, J. Comput. Phys., 165 (2000), pp. 216–236.

[43] L. Pareschi and G. Russo, *On the stability of spectral methods for the homogeneous Boltzmann equation*, Transp. Theory Statist. Phys., 29 (2000), pp. 431–447.

[44] A. Pulvirenti and B. Wennberg, *A Maxwellian lower bound for solutions to the Boltzmann equation*, Comm. Math. Phys., 183 (1997), pp. 145–160.

[45] F. Rogier and J. Schneider, *A direct method for solving the Boltzmann equation*, Transp. Theory Statist. Phys., 23 (1994), pp. 313–338.

[46] G. A. Sod, *A numerical solution of Boltzmann's equation*, Comm. Pure Appl. Math., 30 (1977), pp. 391–419.

[47] R. Strain and Y. Guo, *Almost exponential decay near Maxwellian*, Comm. Partial Differential Equations, 31 (2006), pp. 417–429.

[48] S. Ukai, *On the existence of global solutions of mixed problem for non-linear Boltzmann equation*, Proc. Japan Acad., 50 (1974), pp. 179–184.

[49] C. Villani, *A review of mathematical topics in collisional kinetic theory*, in Handbook of Mathematical Fluid Mechanics, Vol. 1, S. Friedlander and D. Serre, eds., North–Holland, Amsterdam, 2002, pp. 91–309.

SIAM J. SCI. COMPUT.
Vol. 28, No. 3, pp. 1054–1077
© 2006 Society for Industrial and Applied Mathematics

AN ALTERNATING-DIRECTION IMPLICIT ORTHOGONAL SPLINE COLLOCATION SCHEME FOR NONLINEAR PARABOLIC PROBLEMS ON RECTANGULAR POLYGONS*

BERNARD BIALECKI† AND RYAN I. FERNANDES‡

Abstract. We consider a nonlinear parabolic initial-boundary value problem on a rectangular polygon with the solution satisfying Robin boundary conditions with variable coefficients. An approximation to the solution at the desired time value is obtained using an alternating-direction implicit extrapolated Crank–Nicolson scheme in which orthogonal spline collocation with piecewise polynomials of an arbitrary degree is used for spatial discretization. At each time level, the scheme determines the intermediate solution along horizontal lines and the approximate solution along vertical lines passing through the Gauss points. Only at the last time level is the approximate solution along vertical lines converted into the approximate solution defined on the entire rectangular polygon. This property of our approach leads to its efficient implementation and its applicability to rectangular polygons.

Key words. parabolic problems, alternating-direction implicit, extrapolation, orthogonal spline collocation, Gauss points, interpolants, implementation

AMS subject classification. 65M70

DOI. 10.1137/050627885

1. Introduction.

We assume that Ω is a rectangular polygon, that is, a polygon whose sides are parallel to the x- and y-coordinate axes, and consider the numerical solution of the nonlinear parabolic initial-boundary value problem consisting of the differential equation

$$u_t + (L_1 + L_2)u = f(x, y, t, u, u_x, u_y), \qquad (x, y, t) \in \Omega \times [0, d], \tag{1.1}$$

the initial condition

$$u(x, y, 0) = g_0(x, y), \qquad (x, y) \in \overline{\Omega}, \tag{1.2}$$

and Robin boundary conditions on $\partial\Omega$. The differential operators L_1 and L_2 in (1.1) are given by

$$L_1 u = -p_1(x, y, t, u, u_x, u_y)u_{xx}, \qquad L_2 u = -p_2(x, y, t, u, u_x, u_y)u_{yy}, \tag{1.3}$$

where p_1, p_2 are assumed to be continuous functions for which there exist positive constants p_* and p^* so that

$$p_* \le p_i(x, y, t, v, w, z) \le p^*, \quad i = 1, 2, \quad (x, y) \in \Omega, \quad t \in (0, d], \quad v, w, z \in R.$$

In order to demonstrate numerically the expected convergence rates, we also assume that the solution u is sufficiently smooth. Although the paper is primarily concerned

*Received by the editors March 24, 2005; accepted for publication (in revised form) February 14, 2006; published electronically June 30, 2006.
http://www.siam.org/journals/sisc/28-3/62788.html
†Department of Mathematical and Computer Sciences, Colorado School of Mines, Golden, CO 80401 (bbialeck@mines.edu).
‡Department of Mathematics, St. Xaviers College, Mapusa, Goa 403507, India (rfernand@mines.edu).

with the development of a scheme for (1.1), we also discuss a simple extension of the scheme to the equation

$$\begin{aligned} &u_t + (L_1 + L_2)u - p(x,t,u,u_x,u_y)u_{xy} \\ &= f(x,y,t,u,u_x,u_y), \qquad (x,y,t) \in \Omega \times [0,d]. \end{aligned} \tag{1.4}$$

On page 11 of [24], (1.4) is called the quasi-linear equation of general form, while

$$\begin{aligned} &u_t - (q_1(x,y,t,u,u_x,u_y))_x - (q_2(x,y,t,u,u_x,u_y))_y \\ &= f(x,y,t,u,u_x,u_y), \qquad (x,y,t) \in \Omega \times [0,d], \end{aligned} \tag{1.5}$$

is called the quasi-linear equation with principal part in divergence form; see also section 30.4 of [28]. Assuming that q_1 and q_2 of (1.5) are continuously differentiable, we have (see the corresponding formula on page 3 of [24])

$$\begin{aligned} &(q_1(x,y,t,u,u_x,u_y))_x = (q_1)_x(x,y,t,u,u_x,u_y) + (q_1)_u(x,y,t,u,u_x,u_y)u_x \\ &+ (q_1)_{u_x}(x,y,t,u,u_x,u_y)u_{xx} + (q_1)_{u_y}(x,y,t,u,u_x,u_y)u_{xy}. \end{aligned} \tag{1.6}$$

Hence an equation of the form (1.5) can be transformed into an equation of the form (1.4). In the special case when

$$q_1(x,y,t,u,u_x,u_y) = p_1(x,y,t,u)u_x, \quad q_2(x,y,t,u,u_x,u_y) = p_2(x,y,t,u)u_y, \tag{1.7}$$

where p_1 and p_2 are continuously differentiable, (1.5) is transformed into an equation of the form (1.1). It is stated on page 417 of [24] that "all of the widely known equations of mechanics of parabolic type belong to the class (1.5)." Other applications of (1.5) can be found in magnetic fields, heat conduction processes, fluid mechanics, nuclear physics (see page 236 of [29]), and combustion theory (see page 438 of [22]). We consider equations of the form (1.4) since they are more natural for collocation spatial discretization than equations of the form (1.5). Of course, equations of the form (1.5) are more suitable for the finite element Galerkin (FEG) spatial discretization.

The first alternating-direction implicit (ADI) method was introduced in [25] for the finite difference (FD) spatial discretization of a simple linear parabolic problem on a rectangle. The development of ADI methods at that time (see [26]) was motivated by the need to solve multidimensional problems efficiently by reducing them to a collection of one-dimensional problems, each of which can be solved with an optimal cost proportional to the number of unknowns. Several first and second order in time ADI FEG schemes for linear and nonlinear parabolic problems on rectangles were introduced in [17]. The FEG schemes of [17] have direct ADI factorizations only in the case of a linear and separable parabolic operator. For general nonlinear parabolic problems, the ADI FEG methods of [17] are obtained from Laplace modified schemes which include additional truncation error terms and hence, in general, may be less accurate. Moreover, the ADI FEG schemes of [17] require stability parameters whose optimal selections for nonlinear parabolic problems in particular may not be easy. For nonlinear parabolic problems, the first and second order in time ADI schemes of [17] were extended in [12] to certain rectangular polygons. In some cases, implementations of the schemes in [12] were based on the ideas of [18] involving disintegrating difference operators. A generalization of the ADI FEG methods of [17] was carried out in [20] to general curved regions which can be mapped isoparametrically onto a rectangle or certain rectangular polygons. The first order in time ADI methods of [20] were based on the knowledge of the isoparametric mapping and approximation of its Jacobian. In [6], extrapolated backward differentiation formulas of orders 1, 2, and 3 for time

approximation and the FEG spatial discretization were used for nonlinear parabolic problems on rectangles. Since there are no direct ADI factorizations of the proposed perturbation schemes in [6], matrices from ADI schemes with constant coefficients were used as preconditioners at each time level.

In this paper, we use a two level ADI extrapolated Crank–Nicolson scheme for the second order discretization in time and orthogonal spline collocation (OSC) with piecewise polynomials of an arbitrary degree for spatial discretization to solve nonlinear parabolic problems on rectangular polygons. The present paper extends our previous work [4, 5] on ADI OSC schemes for linear parabolic problems on rectangles. For elliptic problems, OSC is based on replacing the exact solution by its piecewise polynomial approximation and on satisfying the partial differential equation at the Gauss points (see, e.g., [2] and references therein). There are several reasons for using OSC in place of FEG or FD for spatial discretization. First, since the idea behind ADI methods is to reduce the solution of a multidimensional problem to the solution of a collection of one-dimensional problems, it is natural to use a spatial discretization which is most efficient for solving one-dimensional problems. As it was shown in the computational study [21] of five finite element methods for the solution of second order, linear, two-point boundary value problems, OSC is the cheapest method for given accuracy and the easiest to implement. Second, in comparison to the FEG schemes in [17] and [6], which have direct ADI factorizations only for linear separable parabolic operators, our OSC scheme has a direct ADI factorization for nonlinear equation (1.1) and hence it does not require a stability parameter or preconditioning. While the FD scheme of [11] also has a direct ADI factorization for (1.1), it is only second order accurate in space and provides only approximations at discrete points of $\overline{\Omega}$. In comparison, our OSC scheme is of arbitrary order in space and yields an approximate solution defined on $\overline{\Omega}$. Third, in comparison to ADI FEG and ADI FD schemes, our ADI OSC scheme is easily applicable to arbitrary rectangular polygons.

Our ADI OSC scheme is most efficient for an equation of the form (1.1). In this case, at each time level, the scheme determines a piecewise polynomial intermediate solution along horizontal lines and a piecewise polynomial approximate solution along vertical lines passing through the Gauss points. Only at the last time level is the approximate solution along vertical lines converted into the approximate solution defined on $\overline{\Omega}$. It is this property of our approach that leads to its efficient implementation and its applicability to rectangular polygons. For an equation of the form (1.4), a modification of our scheme requires the computation, at each time level, of an extrapolated approximation of the mixed derivative. This, in turn, involves converting a function defined along vertical lines passing through the Gauss points into a function defined on $\overline{\Omega}$. If p_1 or p_2 of (1.7) were discontinuous along horizontal or vertical lines in Ω, then, as in the case of FD discretization, the corresponding interface conditions along these lines involving the solution and its flux would have to be incorporated into the definition of our OSC scheme.

If Ω is an arbitrary region, (1.5) can be solved using the extrapolated Crank–Nicolson FEG scheme; cf. (13.21) of [27]. At each time level, a linear elliptic FEG problem is usually solved by preconditioned iterative methods, with domain decomposition or multigrid preconditioners gaining in popularity in recent years. Of course, the smaller the number of iterations at each time level, the smaller the total cost of the scheme is. If Ω is a rectangle, the approach of [6] is based on a similar idea with the matrices from ADI schemes with constant coefficients being used as preconditioners at each time level. At each time level, the number of iterations in [6] is proportional to the logarithm of the number of all time levels. In comparison, our

ADI OSC scheme, applicable to a rectangular polygon Ω and equations of the form (1.1) or (1.4), does not involve iterations at any time level and, consequently, has the optimal cost proportional to the total number of unknowns.

The ADI FD and ADI FEG methods were developed and analyzed extensively in the 1960s, '70s, and '80s. However, even in recent years these methods have been used to solve specific problems [7, 8] or to derive more accurate schemes [23]. The general purpose of our paper is to draw attention to the fact that the ADI OSC methods offer some advantages over ADI FD and ADI FEG schemes in the solution of nonlinear parabolic problems.

A brief outline of the paper is as follows. Section 2 embodies the heart of the paper. While the purpose of this paper is to develop the ADI OSC scheme for any rectangular polygon, in the interest of giving a simple presentation of the scheme and its implementation, in section 2.1 we first formulate the ADI OSC scheme for a rectangle and then comment on its comparison with other schemes in [1, 4, 5, 11] and show how to extend it to solve (1.4). In section 2.2, we present an efficient implementation and give the computational cost of the scheme. Section 3 demonstrates how the ADI OSC scheme can be applied to an L-shaped region. Comparison of our scheme with other ADI FD and ADI FEG schemes on a rectangular polygon is discussed in section 4. Finally, section 5 contains numerical results obtained using our ADI OSC scheme.

2. The ADI OSC scheme for a rectangle. Assume that $\Omega = (a_1, b_1)\times(a_2, b_2)$ and consider the nonlinear parabolic initial-boundary value problem consisting of the differential equation (1.1), (1.3), the initial condition (1.2), Robin boundary conditions on the horizontal sides,

$$\begin{aligned} &\alpha_1(x,t)u(x,a_2,t) - \beta_1(x,t)u_y(x,a_2,t) = g_1(x,t),\\ &\alpha_2(x,t)u(x,b_2,t) + \beta_2(x,t)u_y(x,b_2,t) = g_2(x,t), \end{aligned} \qquad t \in (0,d], \tag{2.1}$$

where $x \in [a_1, b_1]$, and Robin boundary conditions on the vertical sides,

$$\begin{aligned} &\alpha_3(y,t)u(a_1,y,t) - \beta_3(y,t)u_x(a_1,y,t) = g_3(y,t),\\ &\alpha_4(y,t)u(b_1,y,t) + \beta_4(y,t)u_x(b_1,y,t) = g_4(y,t), \end{aligned} \qquad t \in (0,d], \tag{2.2}$$

where $y \in [a_2, b_2]$. The functions α_i and β_i, $i = 1, 2, 3, 4$, in (2.1) and (2.2) are assumed to be nonnegative and not simultaneously equal to zero.

2.1. The scheme. Let $\{x_k\}_{k=0}^{N_x}$ and $\{y_l\}_{l=0}^{N_y}$ be partitions (in general nonuniform) of $[a_1, b_1]$ and $[a_2, b_2]$, respectively, such that

$$a_1 = x_0 < x_1 < \cdots < x_{N_x-1} < x_{N_x} = b_1, \quad a_2 = y_0 < y_1 < \cdots < y_{N_y-1} < y_{N_y} = b_2.$$

Let r be an integer ≥ 3, and let $\mathcal{M}_x$ and $\mathcal{M}_y$ be spaces of piecewise polynomials defined by

$$\mathcal{M}_x = \{v \in C^1[a_1, b_1] : v|_{[x_{k-1},x_k]} \in P_r, k = 1, \ldots, N_x\},$$
$$\mathcal{M}_y = \{v \in C^1[a_2, b_2] : v|_{[y_{l-1},y_l]} \in P_r, l = 1, \ldots, N_y\},$$

where P_r denotes the set of polynomials of degree $\leq r$. The dimensions of $\mathcal{M}_x$ and $\mathcal{M}_y$ are $M_x = N_x(r-1)+2$ and $M_y = N_y(r-1)+2$, respectively. Let $\mathcal{M}$ be the space of piecewise polynomials in x and y given by

$$\mathcal{M} = \mathcal{M}_x \otimes \mathcal{M}_y.$$

($\mathcal{M}$ is the set of all functions that are finite linear combinations of products $v^x(x)v^y(y)$, where $v^x \in \mathcal{M}_x$ and $v^y \in \mathcal{M}_y$.) Let $\{\sigma_j\}_{j=1}^{r-1}$ be the nodes of the $(r-1)$-point Gauss–Legendre quadrature on $[0,1]$, and let $\mathcal{G}_x = \{\xi_k^x\}_{k=1}^{M_x-2}$, $\mathcal{G}_y = \{\xi_l^y\}_{l=1}^{M_y-2}$ be the collocation points defined by

$$\xi^x_{(k-1)(r-1)+j} = x_{k-1} + h^x_k\sigma_j, \qquad k = 1,\dots,N_x, \qquad j = 1,\dots,r-1,$$
$$\xi^y_{(l-1)(r-1)+j} = y_{l-1} + h^y_k\sigma_j, \qquad l = 1,\dots,N_y, \qquad j = 1,\dots,r-1,$$

where $h^x_k = x_k - x_{k-1}$, $h^y_l = y_l - y_{l-1}$. Let

$$\mathcal{G} = \{(x,y) : x \in \mathcal{G}_x, y \in \mathcal{G}_y\}.$$

Let $\{t_n\}_{n=0}^{N_t}$ be a partition of $[0,d]$ such that $t_n = n\tau$, where $\tau = d/N_t$, and let $t_{n+1/2} = (t_n + t_{n+1})/2$, $n = 0,\dots,N_t-1$. For $i = 1,2$, and given functions v and w, let $L_i^m(v,w)$ be the differential operators defined by

$$L_1^m(v,w)u = -p_1(x,y,t_m,v,w_x,v_y)u_{xx}, \qquad L_2^m(v,w)u = -p_2(x,y,t_m,v,w_x,v_y)u_{yy}.$$

In the following, $\phi^m(\cdot) = \phi(\cdot,t_m)$, and $g^m_{1,h}, g^m_{2,h} \in \mathcal{M}_x$ and $g^m_{3,h}, g^m_{4,h} \in \mathcal{M}_y$ are Hermite or Gauss interpolants of g_1^m, g_2^m and g_3^m, g_4^m, respectively. For $v \in C^1[a_1,b_1]$ its Hermite interpolant $v_h \in \mathcal{M}_x$ is defined by (cf. definition of $T_{r,\delta}v$ on page 9 of [15])

$$v_h^{(j)}(x_k) = v^{(j)}(x_k), \qquad j = 0,1, \qquad k = 0,\dots,N_x,$$
$$v_h(x_{k-1} + h^x_k\eta_j) = v(x_{k-1} + h^x_k\eta_j), \qquad k = 1,\dots,N_x, \qquad j = 1,\dots,r-1,$$

where $0 < \eta_1 < \eta_2 < \cdots < \eta_{r-3} < 1$ are the simple roots of $[x^{r-1}(x-1)^{r-1}]^{(r-3)}$. For $v \in C[a_1,b_1]$ its Gauss interpolant $v_h \in \mathcal{M}_x$ is defined by (cf. definition of $S_{r,\delta}v$ on page 10 of [15])

$$v_h(x) = v(x), \qquad x \in \mathcal{G}_x \cup \{a_1,b_1\}.$$

Our ADI OSC scheme to compute $u_h \in \mathcal{M}$ approximating the solution $u(x,y,t)$ of (1.1)–(1.3), (2.1), and (2.2) at $t = d$ consists of the following three stages.

In the first stage, we set up some initial approximations to the exact solution u on vertical and horizontal lines in $\overline{\Omega}$ passing through the points of $\mathcal{G}$. Specifically, for each $x \in \mathcal{G}_x$, the function $u_h^0(x,\cdot) \in \mathcal{M}_y$ approximating $u(x,y,0)$, $y \in [a_2,b_2]$, is obtained by collocating the initial condition; that is, we require that

$$u_h^0(x,y) = g_0(x,y), \qquad y \in \mathcal{G}_y \cup \{a_2,b_2\}. \tag{2.3}$$

Similarly, for each $y \in \mathcal{G}_y$, the function $\hat{u}_h^0(\cdot,y) \in \mathcal{M}_x$ approximating $u(x,y,0)$, $x \in [a_1,b_1]$, is determined using the collocation equations

$$\hat{u}_h^0(x,y) = g_0(x,y), \qquad x \in \mathcal{G}_x \cup \{a_1,b_1\}. \tag{2.4}$$

The functions $u_h^0(x,\cdot) \in \mathcal{M}_y$, $x \in \mathcal{G}_x$, and $\hat{u}_h^0(\cdot,y) \in \mathcal{M}_x$, $y \in \mathcal{G}_y$, are well defined since it can be shown that any $v^y \in \mathcal{M}_y$ and any $v^x \in \mathcal{M}_x$ are uniquely determined by their values at $\mathcal{G}_y \cup \{a_2,b_2\}$ and $\mathcal{G}_x \cup \{a_1,b_1\}$, respectively.

In order to begin advancing the ADI OSC procedure in the t direction, we also require, second order accurate in t, initial approximations to $u(x,y,t_{1/2})$ on vertical

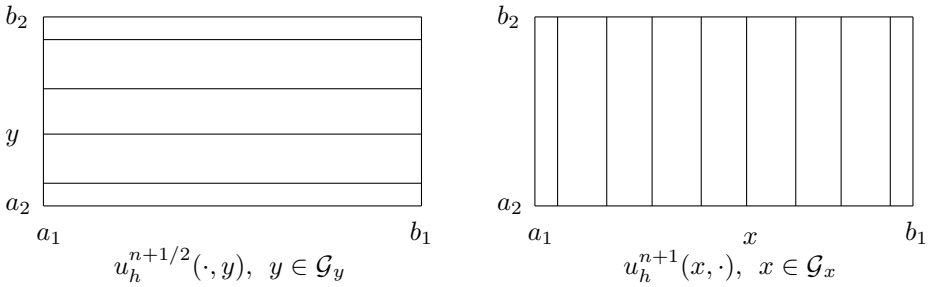


FIG. 2.1. *Rectangle.*

and horizontal lines in $\overline{\Omega}$ passing through the points of $\mathcal{G}$. Using Taylor's theorem, (1.1) with $t = 0$, (1.2), and (1.3), we have

$$u(x, y, t_{1/2}) = g_0(x, y) + \frac{\tau}{2}\left[f(x, y, 0, g_0(x, y), (g_0)_x(x, y), (g_0)_y(x, y)) - \sum_{i=1}^{2} L_i^0(g_0, g_0) g_0(x, y)\right] + O(\tau^2), \qquad (x, y) \in \Omega.$$

Replacing g_0, in the terms on the right-hand side enclosed in the brackets, with u_h^0 and $\hat{u}_h^0$, we introduce

$$g(x, y) = \frac{\tau}{2}\left[f(x, y, 0, u_h^0(x, y), (\hat{u}_h^0)_x(x, y), (u_h^0)_y(x, y)) - L_1^0(u_h^0, \hat{u}_h^0)\hat{u}_h^0(x, y) - L_2^0(u_h^0, \hat{u}_h^0)u_h^0(x, y)\right], \qquad (x, y) \in \mathcal{G}. \tag{2.5}$$

For each $x \in \mathcal{G}_x$, the function $\tilde{u}_h^{1/2}(x, \cdot) \in \mathcal{M}_y$ approximating $u(x, y, t_{1/2})$, $y \in [a_2, b_2]$, is determined using

$$\begin{aligned} &\tilde{u}_h^{1/2}(x, y) = g_0(x, y) + g(x, y), \quad y \in \mathcal{G}_y, \\ &\alpha_1^{1/2}(x)\tilde{u}_h^{1/2}(x, a_2) - \beta_1^{1/2}(x)(\tilde{u}_h^{1/2})_y(x, a_2) = g_1^{1/2}(x), \\ &\alpha_2^{1/2}(x)\tilde{u}_h^{1/2}(x, b_2) + \beta_2^{1/2}(x)(\tilde{u}_h^{1/2})_y(x, b_2) = g_2^{1/2}(x). \end{aligned} \tag{2.6}$$

In a similar manner, for each $y \in \mathcal{G}_y$, the function $\hat{u}_h^{1/2}(\cdot, y) \in \mathcal{M}_x$ approximating $u(x, y, t_{1/2})$, $x \in [a_1, b_1]$, is determined using

$$\begin{aligned} &\hat{u}_h^{1/2}(x, y) = g_0(x, y) + g(x, y), \quad x \in \mathcal{G}_x, \\ &\alpha_3^{1/2}(y)\hat{u}_h^{1/2}(a_1, y) - \beta_3^{1/2}(y)(\hat{u}_h^{1/2})_x(a_1, y) = g_3^{1/2}(y), \\ &\alpha_4^{1/2}(y)\hat{u}_h^{1/2}(b_1, y) + \beta_4^{1/2}(y)(\hat{u}_h^{1/2})_x(b_1, y) = g_4^{1/2}(y). \end{aligned} \tag{2.7}$$

In the second stage, we advance in the t direction. Corresponding to the time levels $t_{n+1/2}$ and t_{n+1}, we compute $u_h^{n+1/2}$ along horizontal lines and u_h^{n+1} along vertical lines, respectively, passing through the Gauss points (see Figure 2.1). Specifically, for $n = 0, \ldots, N_t - 1$ and $x \in \mathcal{G}_x$, we determine $u_h^{n+1}(x, \cdot) \in \mathcal{M}_y$ approximating

$u(x,y,t_{n+1})$, $y\in[a_2,b_2]$, using the scheme

$$\begin{aligned}&\left[\frac{u_h^{n+1/2}-u_h^n}{0.5\tau}+L_1^{n+1/2}(\tilde u_h^{n+1/2},\hat u_h^{n+1/2})u_h^{n+1/2}\right.\\&\qquad\left.+L_2^{n+1/2}(\tilde u_h^{n+1/2},\hat u_h^{n+1/2})u_h^n\right](x,y)\\&=f(x,y,t_{n+1/2},\tilde u_h^{n+1/2}(x,y),(\hat u_h^{n+1/2})_x(x,y),(\tilde u_h^{n+1/2})_y(x,y)),\quad (x,y)\in\mathcal{G},\end{aligned}\tag{2.8}$$

$$\begin{aligned}&\left[\frac{u_h^{n+1}-u_h^{n+1/2}}{0.5\tau}+L_1^{n+1/2}(\tilde u_h^{n+1/2},\hat u_h^{n+1/2})u_h^{n+1/2}\right.\\&\qquad\left.+L_2^{n+1/2}(\tilde u_h^{n+1/2},\hat u_h^{n+1/2})u_h^{n+1}\right](x,y)\\&=f(x,y,t_{n+1/2},\tilde u_h^{n+1/2}(x,y),(\hat u_h^{n+1/2})_x(x,y),(\tilde u_h^{n+1/2})_y(x,y)),\quad (x,y)\in\mathcal{G}.\end{aligned}\tag{2.9}$$

For $n=0$, $x\in\mathcal{G}_x$, and $y\in\mathcal{G}_y$, the functions $u_h^0(x,\cdot)$, $\tilde u_h^{1/2}(x,\cdot)\in\mathcal{M}_y$, and $\hat u_h^{1/2}(\cdot,y)\in\mathcal{M}_x$ are defined by (2.3), (2.6), and (2.7), respectively, and for $n=1,\dots,N_t-1$, $x\in\mathcal{G}_x$, and $y\in\mathcal{G}_y$, the functions $\tilde u_h^{n+1/2}(x,\cdot)\in\mathcal{M}_y$ and $\hat u_h^{n+1/2}(\cdot,y)\in\mathcal{M}_x$ are given by

$$\begin{aligned}&\tilde u_h^{n+1/2}(x,\cdot)=(3/2)u_h^n(x,\cdot)-(1/2)u_h^{n-1}(x,\cdot),\qquad n=1,\dots,N_t-1,\\&\hat u_h^{3/2}(\cdot,y)=3u_h^{1/2}(\cdot,y)-2\hat u_h^0(\cdot,y),\\&\hat u_h^{n+1/2}(\cdot,y)=2u_h^{n-1/2}(\cdot,y)-u_h^{n-3/2}(\cdot,y),\qquad n=2,\dots,N_t-1,\end{aligned}\tag{2.10}$$

where $\hat u_h^0(\cdot,y)\in\mathcal{M}_x$, $y\in\mathcal{G}_y$, is defined by (2.4). For each $y\in\mathcal{G}_y$, the function $u_h^{n+1/2}(\cdot,y)\in\mathcal{M}_x$ is determined using (2.8) and the boundary conditions

$$\begin{aligned}&\alpha_3^{n+1/2}(y)u_h^{n+1/2}(a_1,y)-\beta_3^{n+1/2}(y)(u_h^{n+1/2})_x(a_1,y)\\&=(1/2)(g_{3,h}^{n+1}+g_{3,h}^n)(y)+(\tau/4)L_2^{n+1/2}(\overline{u}_h^{n+1/2},\hat u_h^{n+1/2})(g_{3,h}^{n+1}-g_{3,h}^n)(a_1,y),\\&\alpha_4^{n+1/2}(y)u_h^{n+1/2}(b_1,y)+\beta_4^{n+1/2}(y)(u_h^{n+1/2})_x(b_1,y)\\&=(1/2)(g_{4,h}^{n+1}+g_{4,h}^n)(y)+(\tau/4)L_2^{n+1/2}(\overline{u}_h^{n+1/2},\hat u_h^{n+1/2})(g_{4,h}^{n+1}-g_{4,h}^n)(b_1,y),\end{aligned}\tag{2.11}$$

where for $\gamma=a_1,b_1$, the function $\overline{u}_h^{n+1/2}(\gamma,\cdot)\in\mathcal{M}_y$ is defined by

$$\begin{aligned}&\overline{u}_h^{n+1/2}(\gamma,y)=\hat u_h^{n+1/2}(\gamma,y),\qquad y\in\mathcal{G}_y,\\&\alpha_1^{n+1/2}(\gamma)\overline{u}_h^{n+1/2}(\gamma,a_2)-\beta_1^{n+1/2}(\gamma)(\overline{u}_h^{n+1/2})_y(\gamma,a_2)=g_1^{n+1/2}(\gamma),\\&\alpha_2^{n+1/2}(\gamma)\overline{u}_h^{n+1/2}(\gamma,b_2)+\beta_2^{n+1/2}(\gamma)(\overline{u}_h^{n+1/2})_y(\gamma,b_2)=g_2^{n+1/2}(\gamma).\end{aligned}\tag{2.12}$$

For each $x\in\mathcal{G}_x$, the function $u_h^{n+1}(x,\cdot)\in\mathcal{M}_y$ is determined using (2.9) and the boundary conditions

$$\begin{aligned}&\alpha_1^{n+1}(x)u_h^{n+1}(x,a_2)-\beta_1^{n+1}(x)(u_h^{n+1})_y(x,a_2)=g_{1,h}^{n+1}(x),\\&\alpha_2^{n+1}(x)u_h^{n+1}(x,b_2)+\beta_2^{n+1}(x)(u_h^{n+1})_y(x,b_2)=g_{2,h}^{n+1}(x).\end{aligned}\tag{2.13}$$

Since for $x \in \mathcal{G}_x$ and $y \in \mathcal{G}_y$, the functions $u_h^n(x,\cdot) \in \mathcal{M}_y$ and $u_h^{n-1/2}(\cdot,y) \in \mathcal{M}_x$ are expected to be second order accurate approximations in t to $u(x,y,t_n)$, $y \in [a_2,b_2]$, and $u(x,y,t_{n-1/2})$, $x \in [a_1,b_1]$, respectively, the extrapolation formulas of (2.10) are obtained from the Taylor expansions

$$u^{n+1/2}(x,y) = (3/2)u^n(x,y) - (1/2)u^{n-1}(x,y) + O(\tau^2),$$
$$u^{3/2}(x,y) = 3u^{1/2}(x,y) - 2u^0(x,y) + O(\tau^2),$$
$$u^{n+1/2}(x,y) = 2u^{n-1/2}(x,y) - u^{n-3/2}(x,y) + O(\tau^2).$$

In the third stage, we convert the computed functions $u_h^{N_t}(x,\cdot) \in \mathcal{M}_y$, $x \in \mathcal{G}_x$, into a piecewise polynomial defined on $\overline{\Omega}$. Specifically, the function $u_h \in \mathcal{M}$, approximating $u(x,y,d)$ for $(x,y) \in \overline{\Omega}$, is obtained from the conditions

$$u_h(x,y) = u_h^{N_t}(x,y), \qquad x \in \mathcal{G}_x, \qquad y \in [a_2,b_2], \tag{2.14}$$

and

$$\begin{aligned} &\alpha_3^{N_t}(y)u_h(a_1,y) - \beta_3^{N_t}(y)(u_h)_x(a_1,y) = g_{3,h}^{N_t}(y), \\ &\alpha_4^{N_t}(y)u_h(b_1,y) + \beta_4^{N_t}(y)(u_h)_x(b_1,y) = g_{4,h}^{N_t}(y), \end{aligned} \qquad y \in \mathcal{G}_y \cup \{a_2,b_2\}. \tag{2.15}$$

We now give several remarks concerning the scheme (2.8)–(2.9).

2.1. For nonlinear parabolic problems the scheme (2.8)–(2.9) is similar to the corresponding ADI FD scheme of [11].

2.2. It is possible to obtain the initial functions $u_h^0(x,\cdot) \in \mathcal{M}_y$ and $\hat{u}_h^0(\cdot,y) \in \mathcal{M}_x$ (see (2.3) and (2.4)) using Hermite interpolation rather than collocation.

2.3. For a linear parabolic problem, that is, a problem in which the given functions f and p_1, p_2 of (1.1) and (1.3) depend only on x, y, t, the scheme (2.8)–(2.9) coincides with the scheme (3.1) of [5]. The scheme of [5] is formulated for a linear parabolic problem with

$$L_1u = -p_1(x,y,t)u_{xx} + q_1(x,y,t)u_x + c(x,y,t)u, \quad L_2u = -p_2(x,y,t)u_{yy} + q_2(x,y,t)u_y,$$

and the solution u subject to Dirichlet boundary conditions. It follows from (3.2) of [5] that if, for the nonlinear parabolic problem, Robin boundary conditions on the vertical sides of $\overline{\Omega}$ reduce to Dirichlet boundary conditions, that is, $\alpha_3 = \alpha_4 = 1$ and $\beta_3 = \beta_4 = 0$ in (2.2), then (2.11) with $y \in \mathcal{G}_y$ can be replaced by

$$\begin{aligned} u_h^{n+1/2}(\gamma,y) &= [(1/2)(u_h^{n+1} + u_h^n) \\ &+ (\tau/4)L_2^{n+1/2}((u_h^{n+1} + u_h^n)/2, \hat{u}_h^{n+1/2})(u_h^{n+1} - u_h^n)](\gamma,y), \quad \gamma = a_1, b_1, \end{aligned}$$

where $u_h^n(\gamma,\cdot) \in \mathcal{M}_y$, $n = 0,\dots,N_t$, $\gamma = a_1, b_1$, are given by

$$u_h^n(a_1,\cdot) = g_{3,h}^n, \qquad u_h^n(b_1,\cdot) = g_{4,h}^n.$$

2.4. While the functions $\tilde{u}_h^{n+1/2}$, given by the first equation of (2.10), are defined on the vertical lines passing through the points of $\mathcal{G}$, the functions $\hat{u}_h^{n+1/2}$, given by the last two equations in (2.10), are defined on the horizontal lines passing through the points of $\mathcal{G}$. This is why the $\hat{u}_h^{n+1/2}$ are used at those places in (2.8)–(2.9) that require differentiation with respect to x. On the other hand, $\tilde{u}_h^{n+1/2}$ are used in

(2.8)–(2.9) at all remaining places, including those which involve differentiation with respect to y. It should be noted that for $n = 2, \dots, N_t - 1$ and $y \in \mathcal{G}_y$, the function $\hat{u}_h^{n+1/2}(\cdot, y) \in \mathcal{M}_x$ given by the last equation in (2.10) could be defined in terms of u_h^n and u_h^{n-1} rather than $u_h^{n-1/2}$ and $u_h^{n-3/2}$, using, for example, the equations

$$
\begin{aligned}
&\hat{u}_h^{n+1/2}(x,y) = (3/2)u_h^n(x,y) - (1/2)u_h^{n-1}(x,y), \qquad x \in \mathcal{G}_x,\\
&\alpha_3(y, t_{n+1/2})\hat{u}_h^{n+1/2}(a_1,y) - \beta_3(y, t_{n+1/2})(\hat{u}_h^{n+1/2})_x(a_1,y) = g_3(y, t_{n+1/2}),\\
&\alpha_4(y, t_{n+1/2})\hat{u}_h^{n+1/2}(b_1,y) + \beta_4(y, t_{n+1/2})(\hat{u}_h^{n+1/2})_x(b_1,y) = g_4(y, t_{n+1/2}),
\end{aligned}
\tag{2.16}
$$

the first of which is similar to the first equation of (2.10) defining $\tilde{u}_h^{n+1/2}$. However, we choose to use the last equation of (2.10) for the definition of $\hat{u}_h^{n+1/2}$ since its implementation is less expensive than that of (2.16) which requires solving a collection of independent one-dimensional OSC two-point boundary value problems.

REMARK 2.5. In (2.9), it seems natural to use the function $u_h^{n+1/2}$ determined from (2.8) and (2.11) in place of $\hat{u}_h^{n+1/2}$. However, such a modification of the scheme would require an additional evaluation of the function f and a more involved implementation than the one presented in this paper.

REMARK 2.6. The scheme (2.8)–(2.9) can be viewed as the one obtained from linearization of the nonlinear ADI OSC scheme

$$
\begin{aligned}
&\left[\frac{u_h^{n+1/2} - u_h^n}{0.5\tau} + L_1^{n+1/2}(u_h^{n+1/2}, u_h^{n+1/2})u_h^{n+1/2} \right.\\
&\qquad \left. + L_2^{n+1/2}(u_h^{n+1/2}, u_h^{n+1/2})u_h^n\right](x,y)\\
&= f(x, y, t_{n+1/2}, u_h^{n+1/2}(x,y), (u_h^{n+1/2})_x(x,y), (u_h^{n+1/2})_y(x,y)), \quad (x,y) \in \mathcal{G},\\
&\left[\frac{u_h^{n+1} - u_h^{n+1/2}}{0.5\tau} + L_1^{n+1/2}(u_h^{n+1/2}, u_h^{n+1/2})u_h^{n+1/2} \right.\\
&\qquad \left. + L_2^{n+1/2}(u_h^{n+1/2}, u_h^{n+1/2})u_h^{n+1}\right](x,y)\\
&= f(x, y, t_{n+1/2}, u_h^{n+1/2}(x,y), (u_h^{n+1/2})_x(x,y), (u_h^{n+1/2})_y(x,y)), \quad (x,y) \in \mathcal{G},
\end{aligned}
\tag{2.17}
$$

in which the unknown function $u_h^{n+1/2}$ in $L_1^{n+1/2}$, $L_2^{n+1/2}$, and f is replaced with its corresponding extrapolated approximations given in (2.10). In comparison to (2.8)–(2.9), the scheme (2.17) would require solving a collection of one-dimensional nonlinear, rather than linear, OSC two-point boundary value problems.

REMARK 2.7. An alternative to (2.8)–(2.9) is the scheme

$$
\begin{aligned}
&\left[\frac{u_h^{n+1/2} - u_h^n}{0.5\tau} + L_1^{n+1/2}(\tilde{u}_h^{n+1/2}, \hat{u}_h^{n+1/2})u_h^{n+1/2} + L_2^n(u_h^n, \hat{u}_h^n)u_h^n\right](x,y)\\
&= f(x, y, t_{n+1/2}, \tilde{u}_h^{n+1/2}(x,y), (\hat{u}_h^{n+1/2})_x(x,y), (\tilde{u}_h^{n+1/2})_y(x,y)), \quad (x,y) \in \mathcal{G},\\
&\left[\frac{u_h^{n+1} - u_h^{n+1/2}}{0.5\tau} + L_1^{n+1/2}(\tilde{u}_h^{n+1/2}, \hat{u}_h^{n+1/2})u_h^{n+1/2} + L_2^{n+1}(\tilde{u}_h^{n+1}, \hat{u}_h^{n+1})u_h^{n+1}\right](x,y)\\
&= f(x, y, t_{n+1/2}, \tilde{u}_h^{n+1/2}(x,y), (\hat{u}_h^{n+1/2})_x(x,y), (\tilde{u}_h^{n+1/2})_y(x,y)), \quad (x,y) \in \mathcal{G},
\end{aligned}
\tag{2.18}
$$

where the additional functions $\tilde{u}_h^{n+1}(x,\cdot)$ and $\hat{u}_h^{n+1}(x,\cdot)$ in L_2^{n+1} are given by

$$\tilde{u}_h^{n+1}(x,\cdot) = 2u_h^n(x,\cdot) - u_h^{n-1}(x,\cdot), \quad \hat{u}_h^{n+1}(x,\cdot) = (3/2)u_h^{n+1/2}(x,\cdot) - (1/2)u_h^{n-1/2}(x,\cdot).$$

The scheme (2.18) is also second order accurate in t and is obtained from (2.8)–(2.9) by replacing the operator $L_2^{n+1/2}(\tilde{u}_h^{n+1/2}, \hat{u}_h^{n+1/2})$ in (2.8) and (2.9) with the operators $L_2^n(u_h^n, \hat{u}_h^n)$ and $L_2^{n+1}(\tilde{u}_h^{n+1}, \hat{u}_h^{n+1})$, respectively. For a linear parabolic problem, the scheme (2.18) coincides with the scheme (3.41) of [4].

REMARK 2.8. For the operators L_1, L_2 in the divergence forms

$$L_1 u = -(p_1(x,y,t,u,u_x,u_y)u_x)_x, \qquad L_2 u = -(p_2(x,y,t,u,u_x,u_y)u_y)_y$$

and zero Neumann boundary conditions, an ADI OSC scheme similar to ours is discussed in [1]. The authors of [1] formulate (2.8)–(2.9) with $(1/2)(u_h^{n+1} + u_h^n)$ in place of $\tilde{u}_h^{n+1/2}$, $\hat{u}_h^{n+1/2}$ and suggest, but do not seem to carry out, the extrapolation procedure of [16] in the implementation of their scheme for nonlinear problems. The application of the extrapolation procedure of [16] in the formulation of the scheme yields the formula for $\tilde{u}_h^{n+1/2}$ in (2.10), which along with the other formulas of (2.10) leads to (2.8)–(2.9). The main difference between the two schemes is that in [1] the intermediate and approximate solutions $u_h^{n+1/2}$ and u_h^{n+1} are required to be piecewise Hermite bicubics defined on $\overline{\Omega}$, whereas in (2.8)–(2.9) they are, respectively, piecewise polynomials along horizontal and vertical lines passing through the points of $\mathcal{G}$. This property of our scheme leads to its efficient implementation and its generalization to rectangular polygons.

REMARK 2.9. Our ADI OSC scheme for (1.1) can be easily extended to (1.4) by extrapolating the term $p(x,y,t,u,u_x,u_y)u_{xy}$. Specifically, for $n = 0, \ldots, N_t - 1$, we include on the right-hand sides of (2.8) and (2.9) the additional term

$$p(x,y,t_{n+1/2},\tilde{u}_h^{n+1/2}(x,y),(\hat{u}_h^{n+1/2})_x(x,y),(\tilde{u}_h^{n+1/2})_y(x,y))(\tilde{u}_h^{n+1/2})_{xy}(x,y),$$

where, for each n, $\tilde{u}_h^{n+1/2}$ defined along vertical lines passing through the Gauss points is converted into a function defined on $\overline{\Omega}$. A similar modification is also made on the right-hand side of (2.5) by including the additional term

$$\frac{\tau}{2}p(x,y,0,u_h^0(x,y),(\hat{u}_h^0)_x(x,y),(u_h^0)_y(x,y))(u_h^0)_{xy}(x,y),$$

where u_h^0 defined along vertical lines passing through the Gauss points is converted into a function defined on $\overline{\Omega}$.

2.2. Implementation and cost. To discuss implementation of the ADI OSC scheme of section 2.1, we assume that $\{\phi_i^x\}_{i=1}^{M_x}$ and $\{\phi_j^y\}_{j=1}^{M_y}$ are B-spline basis functions for $\mathcal{M}_x$ and $\mathcal{M}_y$, respectively. This implies (see [10, page 445]), that

$$\begin{gathered}\phi_1^x(a_1) \neq 0, \quad [\phi_1^x]'(a_1) \neq 0, \qquad \phi_2^x(a_1) = 0, \quad [\phi_2^x]'(a_1) \neq 0,\\ \phi_i^x(a_1) = [\phi_i^x]'(a_1) = 0, \quad i = 3, \ldots, M_x,\end{gathered} \tag{2.19}$$

$$\begin{gathered}\phi_i^x(b_1) = [\phi_i^x]'(b_1) = 0, \quad i = 1, \ldots, M_x - 2,\\ \phi_{M_x-1}^x(b_1) = 0, \quad [\phi_{M_x-1}^x]'(b_1) \neq 0, \qquad \phi_{M_x}^x(b_1) \neq 0, \quad [\phi_{M_x}^x]'(b_1) \neq 0,\end{gathered} \tag{2.20}$$

and that $\{\phi_j^y\}_{j=1}^{M_y}$ have similar properties at a_2 and b_2.

Since for each ξ_k^x, $k = 1, \ldots, M_x - 2$, the functions $u_h^n(\xi_k^x, \cdot)$, $\tilde{u}_h^{n+1/2}(\xi_k^x, \cdot) \in \mathcal{M}_y$, and for each ξ_l^y, $l = 1, \ldots, M_y - 2$, the functions $\hat{u}_h^0(\cdot, \xi_l^y)$, $\hat{u}_h^{n+1/2}(\cdot, \xi_l^y)$, $u_h^{n+1/2}(\cdot, \xi_l^y) \in \mathcal{M}_x$, we have

$$u_h^n(\xi_k^x, y) = \sum_{j=1}^{M_y} u_{k,j}^n \phi_j^y(y), \tag{2.21}$$

$$\tilde{u}_h^{n+1/2}(\xi_k^x, y) = \sum_{j=1}^{M_y} \tilde{u}_{k,j}^{n+1/2} \phi_j^y(y), \tag{2.22}$$

$$\hat{u}_h^0(x, \xi_l^y) = \sum_{i=1}^{M_x} \hat{u}_{i,l}^0 \phi_i^x(x), \tag{2.23}$$

$$\hat{u}_h^{n+1/2}(x, \xi_l^y) = \sum_{i=1}^{M_x} \hat{u}_{i,l}^{n+1/2} \phi_i^x(x), \tag{2.24}$$

$$u_h^{n+1/2}(x, \xi_l^y) = \sum_{i=1}^{M_x} u_{i,l}^{n+1/2} \phi_i^x(x). \tag{2.25}$$

Implementation of the first stage of the scheme requires the computation of the expansion coefficients of the initial approximations $u_h^0(\xi_k^x, \cdot)$, $\hat{u}_h^0(\cdot, \xi_l^y)$, $\tilde{u}_h^{1/2}(\xi_k^x, \cdot)$, and $\hat{u}_h^{1/2}(\cdot, \xi_l^y)$. It follows from (2.3) and (2.21) that the coefficients $\{u_{k,j}^0\}_{j=1}^{M_y}$ of $u_h^0(\xi_k^x, \cdot)$ satisfy the equations

$$\sum_{j=1}^{M_y} \phi_j^y(\xi_l^y) u_{k,j}^0 = g_0(\xi_k^x, \xi_l^y), \qquad l = 0, \ldots, M_y - 1, \tag{2.26}$$

where

$$\xi_0^y = a_2, \qquad \xi_{M_y-1}^y = b_2.$$

Therefore, for fixed $k = 1, \ldots, M_x - 2$, the coefficients $\{u_{k,j}^0\}_{j=1}^{M_y}$ are obtained by solving a linear system with the matrix

$$C^y = (c_{l,j}^y)_{l=0,j=1}^{M_y-1,M_y}, \qquad c_{l,j}^y = \phi_j^y(\xi_l^y). \tag{2.27}$$

Similarly, it follows from (2.4) and (2.23) that for fixed $l = 1, \ldots, M_y - 2$, the coefficients $\{\hat{u}_{i,l}^0\}_{i=1}^{M_x}$ of $\hat{u}_h^0(\cdot, \xi_l^y)$ are determined by solving a linear system with the matrix

$$C^x = (c_{k,i}^x)_{k=0,i=1}^{M_x-1,M_x}, \qquad c_{k,i}^x = \phi_i^x(\xi_k^x), \tag{2.28}$$

where

$$\xi_0^x = a_1, \qquad \xi_{M_x-1}^x = b_1.$$

The matrices C^x and C^y have an almost block diagonal (ABD) structure [9]. It follows from (2.19) and (2.20) that the first element in the first rows of C_x and C_y is the only

nonzero element in these rows and that the last element in the last rows of C_x and C_y is the only nonzero element in these rows. The remaining nonzero elements of C^x and C^y are located along the diagonals in $(r-1)\times(r+1)$ blocks which overlap in two columns; there are N_x such blocks in C^x and N_y blocks in C^y. Linear systems with C^x and C^y can be solved efficiently using the package COLROW [13], [14] at costs $O(r^3N_x)$ and $O(r^3N_y)$, respectively.

Equations (2.6) and (2.22) imply that for fixed $k=1,\dots,M_x-2$, the coefficients $\{\tilde u^{1/2}_{k,j}\}_{j=1}^{M_y}$ of $\tilde u^{1/2}_h(\xi^x_k,\cdot)$ are computed by solving an ABD linear system with the matrix that differs from C^y in the first and last rows; the first two and, respectively, the last two elements being the only nonzero elements in these rows. Similarly, (2.7) and (2.24) imply that for fixed $l=1,\dots,M_y-2$, the coefficients $\{\hat u^{1/2}_{i,l}\}_{i=1}^{M_x}$ of $\hat u^{1/2}_h(\cdot,\xi^y_l)$ are obtained by solving an ABD linear system with the matrix that differs from C^x in the first and last rows. Again, with the use of COLROW, linear systems with the modified C^x and C^y are solved at costs $O(r^3N_x)$ and $O(r^3N_y)$, respectively. It follows from (2.6), (2.7), (2.5), (2.23), and (2.21) that the computation of the right-hand sides in the systems for $\{\tilde u^{1/2}_{k,j}\}_{j=1}^{M_y}$ and $\{\hat u^{1/2}_{i,l}\}_{i=1}^{M_x}$ involves, among other things, matrix-vector multiplications with the matrices

$$B^{1,l}=(b^{1,l}_{k,i})_{k=1,i=1}^{M_x-2,M_x},\qquad b^{1,l}_{k,i}=L^0_1(u^0_h,\hat u^0_h)\phi^x_i(\xi^x_k,\xi^y_l),\qquad l=1,\dots,M_y-2,$$

and

$$B^{2,k}=(b^{2,k}_{l,j})_{l=1,j=1}^{M_y-2,M_y},\qquad b^{2,k}_{l,j}=L^0_2(u^0_h,\hat u^0_h)\phi^y_j(\xi^x_k,\xi^y_l),\qquad k=1,\dots,M_x-2.$$

Since $B^{1,l}$ and $B^{2,k}$ are ABD matrices, multiplying $B^{1,l}$ and $B^{2,k}$ by a vector requires $O(r^2N_x)$ and $O(r^2N_y)$ operations, respectively.

To describe implementation of the second stage of the scheme, we first rewrite (2.8) in the form

$$\begin{aligned}&\left[I+0.5\tau L_1^{n+1/2}(\tilde u_h^{n+1/2},\hat u_h^{n+1/2})\right]u_h^{n+1/2}(x,y)\\&\qquad=\left[I-0.5\tau L_2^{n+1/2}(\tilde u_h^{n+1/2},\hat u_h^{n+1/2})\right]u_h^n(x,y)\\&+0.5\tau f(x,y,t_{n+1/2},\tilde u_h^{n+1/2}(x,y),(\hat u_h^{n+1/2})_x(x,y),(\tilde u_h^{n+1/2})_y(x,y)),\quad (x,y)\in\mathcal{G},\end{aligned}$$

where I is the identity operator. Subtracting (2.8) from (2.9) and multiplying by 0.5τ, we obtain

$$\begin{aligned}&\left[I+0.5\tau L_2^{n+1/2}(\tilde u_h^{n+1/2},\hat u_h^{n+1/2})\right]u_h^{n+1}(x,y)\\&\qquad=2u_h^{n+1/2}(x,y)-\left[I-0.5\tau L_2^{n+1/2}(\tilde u_h^{n+1/2},\hat u_h^{n+1/2})\right]u_h^n(x,y),\qquad (x,y)\in\mathcal{G}.\end{aligned}$$

Thus, for fixed $n=0,\dots,N_t-1$, the second stage of the scheme is implemented using equations

$$\begin{aligned}&\left[I+0.5\tau L_1^{n+1/2}(\tilde u_h^{n+1/2},\hat u_h^{n+1/2})\right]u_h^{n+1/2}(x,y)=v_h^n(x,y)\\&\qquad+0.5\tau f(x,y,t_{n+1/2},\tilde u_h^{n+1/2}(x,y),(\hat u_h^{n+1/2})_x(x,y),(\tilde u_h^{n+1/2})_y(x,y)),\end{aligned}\tag{2.29}$$

$$\left[I+0.5\tau L_2^{n+1/2}(\tilde u_h^{n+1/2},\hat u_h^{n+1/2})\right]u_h^{n+1}(x,y)=2u_h^{n+1/2}(x,y)-v_h^n(x,y),\tag{2.30}$$

$$v_h^n(x,y) = \left[I - 0.5\tau L_2^{n+1/2}(\tilde{u}_h^{n+1/2}, \hat{u}_h^{n+1/2})\right] u_h^n(x,y), \tag{2.31}$$

with $(x,y) \in \mathcal{G}$, where for $y \in \mathcal{G}_y$ and $x \in \mathcal{G}_x$, the functions $u_h^{n+1/2}(\cdot,y) \in \mathcal{M}_x$ and $u_h^{n+1}(x,\cdot) \in \mathcal{M}_y$ satisfy the boundary conditions (2.11) and (2.13), respectively. Using (2.25) and (2.21) in (2.29)–(2.31), and setting $(x,y) = (\xi_k^x, \xi_l^y)$, we obtain

$$\begin{aligned}
&\sum_{i=1}^{M_x} \left[\phi_i^x(\xi_k^x) + 0.5\tau L_1^{n+1/2}(\tilde{u}_h^{n+1/2}, \hat{u}_h^{n+1/2})\phi_i^x(\xi_k^x, \xi_l^y)\right] u_{i,l}^{n+1/2} = v_h^n(\xi_k^x, \xi_l^y) \\
&\quad + 0.5\tau f(\xi_k^x, \xi_l^y, t_{n+1/2}, \tilde{u}_h^{n+1/2}(\xi_k^x, \xi_l^y), (\hat{u}_h^{n+1/2})_x(\xi_k^x, \xi_l^y), (\tilde{u}_h^{n+1/2})_y(\xi_k^x, \xi_l^y)),
\end{aligned} \tag{2.32}$$

$$\begin{aligned}
&\sum_{j=1}^{M_y} \left[\phi_j^y(\xi_l^y) + 0.5\tau L_2^{n+1/2}(\tilde{u}_h^{n+1/2}, \hat{u}_h^{n+1/2})\phi_j^y(\xi_k^x, \xi_l^y)\right] u_{k,j}^{n+1} \\
&\qquad = 2\sum_{i=1}^{M_x} \phi_i^x(\xi_k^x) u_{i,l}^{n+1/2} - v_h^n(\xi_k^x, \xi_l^y),
\end{aligned} \tag{2.33}$$

$$v_h^n(\xi_k^x, \xi_l^y) = \sum_{j=1}^{M_y} \left[\phi_j^y(\xi_l^y) - 0.5\tau L_2^{n+1/2}(\tilde{u}_h^{n+1/2}, \hat{u}_h^{n+1/2})\phi_j^y(\xi_k^x, \xi_l^y)\right] u_{k,j}^n,$$

where $k = 1, \ldots, M_x - 2$ and $l = 1, \ldots, M_y - 2$. For fixed $k = 1, \ldots, M_x - 2$, the computation of $\{v_h^n(\xi_k^x, \xi_l^y)\}_{l=1}^{M_y-2}$ involves matrix-vector multiplication with the matrix

$$A^{2,k} = (a_{l,j}^{2,k})_{l=1,j=1}^{M_y-2,M_y}, \quad a_{l,j}^{2,k} = \phi_j^y(\xi_l^y) - 0.5\tau L_2^{n+1/2}(\tilde{u}_h^{n+1/2}, \hat{u}_h^{n+1/2})\phi_j^y(\xi_k^x, \xi_l^y). \tag{2.34}$$

Since $A^{2,k}$ is an ABD matrix, the cost of multiplying $A^{2,k}$ by a vector is $O(r^2 N_y)$. For fixed $l = 1, \ldots, M_y - 2$, (2.32) with $k = 1, \ldots, M_x - 2$ and both equations in (2.11) with $y = \xi_l^y$ involve solving an ABD linear system in the x-direction for the coefficients $\{u_{i,l}^{n+1/2}\}_{i=1}^{M_x}$ of $u_h^{n+1/2}(\cdot, \xi_l^y)$. Similarly, for fixed $k = 1, \ldots, M_x - 2$, (2.33) with $l = 1, \ldots, M_y - 2$ and both equations in (2.13) with $x = \xi_k^x$ involve solving an ABD linear system in the y-direction for the coefficients $\{u_{k,j}^{n+1}\}_{j=1}^{M_y}$ of $u_h^{n+1}(\xi_k^x, \cdot)$. The corresponding matrices in these systems are given by formulas similar to those for $A_{2,k}$ of (2.34). With the help of COLROW, the systems in the x- and y-directions are solved at costs $O(r^3 N_x)$ and $O(r^3 N_y)$, respectively. Setting up the matrix $A^{2,k}$ of (2.34), the matrix and right-hand side in the linear system corresponding to (2.32), and the matrix in the linear system corresponding to (2.33) involves multiplications of ABD matrices by the vectors $\{\tilde{u}_{k,j}^{n+1/2}\}_{j=1}^{M_y}$ and $\{\hat{u}_{i,l}^{n+1/2}\}_{i=1}^{M_x}$ whose components, according to (2.10) and (2.21)–(2.25), are computed using formulas

$$\begin{aligned}
\tilde{u}_{k,j}^{n+1/2} &= (3/2)u_{k,j}^n - (1/2)u_{k,j}^{n-1}, \qquad n = 1, \ldots, N_t - 1, \\
\hat{u}_{i,l}^{3/2} &= 3u_{i,l}^{1/2} - 2\hat{u}_{i,l}^0, \qquad \hat{u}_{i,l}^{n+1/2} = 2u_{i,l}^{n-1/2} - u_{i,l}^{n-3/2}, \qquad n = 2, \ldots, N_t - 1.
\end{aligned}$$

In the third phase of the ADI OSC scheme, the coefficients $u_{i,j}$ in

$$u_h(x,y) = \sum_{i=1}^{M_x} \sum_{j=1}^{M_y} u_{i,j}\phi_i^x(x)\phi_j^y(y), \qquad (x,y) \in \overline{\Omega}, \tag{2.35}$$

are determined using (2.14) and (2.15). It follows from (2.14) and (2.21) with $n = N_t$ that

$$u_h(\xi_k^x, y) = \sum_{j=1}^{M_y} u_{k,j}^{N_t} \phi_j^y(y), \qquad k = 1, \ldots, M_x - 2, \qquad y \in [a_2, b_2], \tag{2.36}$$

where for each $k = 1, \ldots, M_x - 2$, the coefficients $\{u_{k,j}^{N_t}\}_{j=1}^{M_y}$ are known. Setting $x = \xi_k^x$, $k = 1, \ldots, M_x - 2$, in (2.35) and equating the coefficients of ϕ_j^y in the resulting right-hand side with those on the right-hand sides in (2.36), we obtain

$$\sum_{i=1}^{M_x} \phi_i^x(\xi_k^x) u_{i,j} = u_{k,j}^{N_t}, \qquad k = 1, \ldots, M_x - 2, \tag{2.37}$$

for $j = 1, \ldots, M_y$. For fixed j, (2.37) forms a system of $M_x - 2$ equations in the unknown coefficients $\{u_{i,j}\}_{i=1}^{M_x}$. Since the number of the unknown coefficients is M_x, we require two additional equations or knowledge of two coefficients. In the following we show how to determine $\{u_{1,j}\}_{j=1}^{M_y}$ and $\{u_{M_x,j}\}_{j=1}^{M_y}$. (It follows from (2.37) and (2.28) that once $u_{1,j}$ and $u_{M_x,j}$ are known, the remaining unknown coefficients $\{u_{i,j}\}_{i=2}^{M_x-1}$ are obtained with cost $O(r^3 N_x)$ by solving, using COLROW, an ABD linear system with the matrix that differs from C^x in the first and last rows whose first and last elements, respectively, are set to 1.) Since $u_h(\cdot, y) \in \mathcal{M}_x$ for fixed $y \in [a_2, b_2]$, there are coefficients $\{z_{i,l}\}_{i=1}^{M_x}$, $l = 0, \ldots, M_y - 1$, such that

$$u_h(x, \xi_l^y) = \sum_{i=1}^{M_x} z_{i,l} \phi_i^x(x), \qquad x \in [a_1, b_1], \qquad l = 0, \ldots, M_y - 1. \tag{2.38}$$

Then it follows from (2.35) with $x = a_1, b_1$, $y = \xi_l^y$, $l = 0, \ldots, M_y - 1$, (2.38) with $x = a_1, b_1$, (2.19), and (2.20) that

$$\sum_{j=1}^{M_y} \phi_j^y(\xi_l^y) u_{1,j} = z_{1,l}, \qquad l = 0, \ldots, M_y - 1,$$

$$\sum_{j=1}^{M_y} \phi_j^y(\xi_l^y) u_{M_x,j} = z_{M_x,l}, \qquad l = 0, \ldots, M_y - 1.$$

Solving these two systems using COLROW with cost $O(r^3 N_y)$ (cf. definition (2.27) of the ABD matrix C^y), we obtain the required coefficients $\{u_{1,j}\}_{j=1}^{M_y}$ and $\{u_{M_x,j}\}_{j=1}^{M_y}$. Thus, it remains to show how $\{z_{1,l}\}_{l=0}^{M_y-1}$ and $\{z_{M_x,l}\}_{l=0}^{M_y-1}$ are computed.

It follows from (2.38) with $x = \xi_k^x$, $k = 1, \ldots, M_x - 2$, and (2.36) with $y = \xi_l^y$, $l = 0, \ldots, M_y - 1$, that

$$\sum_{i=1}^{M_x} \phi_i^x(\xi_k^x) z_{i,l} = \sum_{j=1}^{M_y} \phi_j^y(\xi_l^y) u_{k,j}^{N_t}, \qquad k = 1, \ldots, M_x - 2, \tag{2.39}$$

for $l = 0, \ldots, M_y - 1$. Substituting (2.38) into (2.15) with $y = \xi_l^y$ and using (2.19), (2.20), we also obtain

$$\left\{\alpha_3^{N_t}(\xi_l^y)\phi_1^x(a_1) - \beta_3^{N_t}(\xi_l^y)[\phi_1^x]'(a_1)\right\} z_{1,l} - \beta_3^{N_t}(\xi_l^y)[\phi_2^x]'(a_1) z_{2,l} = g_{3,h}^{N_t}(\xi_l^y) \tag{2.40}$$

and

$$\left\{\alpha_4^{N_t}(\xi_l^y)\phi_{M_x}^x(b_1)+\beta_4^{N_t}(\xi_l^y)[\phi_{M_x}^x]'(b_1)\right\}z_{M_x,l} + \beta_4^{N_t}(\xi_l^y)[\phi_{M_x-1}^x]'(b_1)z_{M_x-1,l}=g_{4,h}^{N_t}(\xi_l^y) \tag{2.41}$$

for $l=0,\dots,M_y-1$. Therefore, for fixed $l=0,\dots,M_y-1$, the coefficients $\{z_{i,l}\}_{i=1}^{M_x}$, and hence $z_{1,l}$ and $z_{M_x,l}$, are obtained by solving the ABD linear system consisting of (2.39), (2.40), and (2.41). By (2.39) and (2.28), the matrix in this system differs from C^x in the first and last rows. It follows from (2.39) and (2.27) that the right-hand sides in these systems are first computed by multiplying C^y by $\{u_{k,j}^{N_t}\}_{j=1}^{M_y}$ for $k=1,\dots,M_x-2$.

If the Dirichlet boundary conditions are imposed on the vertical sides of $\overline{\Omega}$, the systems (2.39)–(2.41) need not be solved since $z_{1,l}$ and $z_{M_x,l}$ are known explicitly. Indeed, it follows from (2.40) and (2.41) with $\alpha_3=\alpha_4=1$ and $\beta_3=\beta_4=0$ that

$$z_{1,l}=\frac{g_{3,h}^{N_t}(\xi_l^y)}{\phi_1^x(a_1)},\qquad z_{M_x,l}=\frac{g_{4,h}^{N_t}(\xi_l^y)}{\phi_{M_x}^x(b_1)},\qquad l=0,\dots,M_y-1.$$

Assume $N_x=N_y=N$. Since M_x and M_y are proportional to rN_x and rN_y, respectively, it follows that the costs of stages 1 and 3 and, for fixed n, stage 2 are $O(r^4N^2)$ each. Hence the total cost of computing the approximation u_h is $O(r^4N^2N_t)$.

3. The ADI OSC scheme for an *L*-shaped region. Assume $a_1<c_1<b_1$, $a_2<c_2<b_2$ and consider the L-shaped region $\Omega=\{(a_1,b_1)\times(a_2,b_2)\}\setminus\{[c_1,b_1)\times[c_2,b_2)\}$ (see Figure 3.1).

Consider problem (1.1)–(1.2) with L_1 and L_2 as in (1.3), the solution u subject to the conditions in (2.1) with $x\in[a_1,b_1]$, $x\in[a_1,c_1]$, respectively, the conditions in (2.2) with $y\in[a_2,b_2]$, $y\in[a_2,c_2]$, respectively, and the conditions

$$\begin{aligned}&\alpha_5(x,t)u(x,c_2,t)+\beta_5(x,t)u_y(x,c_2,t)=g_5(x,t),\quad x\in[c_1,b_1],\\ &\alpha_6(y,t)u(c_1,y,t)+\beta_6(y,t)u_x(c_1,y,t)=g_6(y,t),\quad y\in[c_2,b_2],\end{aligned}\qquad t\in(0,d]. \tag{3.1}$$

Let $\{x_k\}_{k=0}^{N_x}$ and $\{y_l\}_{l=0}^{N_y}$ of section 2 be such that

$$c_1=x_i,\qquad c_2=y_j$$

for some integers i and j. Let $\mathcal{M}_x$, $\mathcal{M}_y$, $\mathcal{G}_x$, $\mathcal{G}_y$, and $\mathcal{G}$ be as in section 2. Let $\mathcal{M}_x^{(1)}$, $\mathcal{M}_x^{(2)}$ be restrictions of $\mathcal{M}_x$ to $[a_1,c_1]$, $[c_1,b_1]$, respectively, and let $\mathcal{M}_y^{(1)}$, $\mathcal{M}_y^{(2)}$ be restrictions of $\mathcal{M}_y$ to $[a_2,c_2]$, $[c_2,b_2]$, respectively. Let $\mathcal{G}_x^{(1)}=\mathcal{G}_x\cap(a_1,c_1)$, $\mathcal{G}_x^{(2)}=\mathcal{G}_x\cap(c_1,b_1)$, $\mathcal{G}_y^{(1)}=\mathcal{G}_y\cap(a_2,c_2)$, and $\mathcal{G}_y^{(2)}=\mathcal{G}_y\cap(c_2,b_2)$. Let $g_{1,h}^m\in\mathcal{M}_x$, $g_{2,h}^m\in\mathcal{M}_x^{(1)}$, $g_{3,h}^m\in\mathcal{M}_y$, $g_{4,h}^m\in\mathcal{M}_y^{(1)}$, $g_{5,h}^m\in\mathcal{M}_x^{(2)}$, and $g_{6,h}^m\in\mathcal{M}_y^{(2)}$ be Hermite or Gauss interpolants of $g_1^m,\dots,g_6^m$, respectively.

In the first stage of the ADI OSC scheme, the functions $u_h^0(x,\cdot)\in\mathcal{M}_y$, $x\in\mathcal{G}_x^{(1)}$, and $\hat{u}_h^0(\cdot,y)\in\mathcal{M}_x$, $y\in\mathcal{G}_y^{(1)}$, are defined by (2.3) and (2.4), respectively. The functions $u_h^0(x,\cdot)\in\mathcal{M}_y^{(1)}$, $x\in\mathcal{G}_x^{(2)}$, and $\hat{u}_h^0(\cdot,y)\in\mathcal{M}_x^{(1)}$, $y\in\mathcal{G}_y^{(2)}$, are defined by

$$u_h^0(x,y)=g_0(x,y),\qquad y\in\mathcal{G}_y^{(1)}\cup\{a_2,c_2\},$$

and

$$\hat{u}_h^0(x,y)=g_0(x,y),\qquad x\in\mathcal{G}_x^{(1)}\cup\{a_1,c_1\}.$$

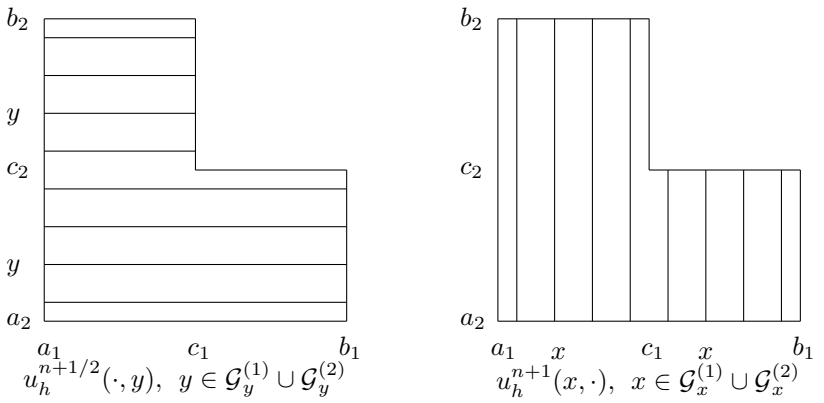


FIG. 3.1. *L-shaped region.*

The functions $\tilde{u}_h^{1/2}(x, \cdot) \in \mathcal{M}_y$, $x \in \mathcal{G}_x^{(1)}$, and $\hat{u}_h^{1/2}(\cdot, y) \in \mathcal{M}_x$, $y \in \mathcal{G}_y^{(1)}$, are defined by (2.6) and (2.7), respectively. The functions $\tilde{u}_h^{1/2}(x, \cdot) \in \mathcal{M}_y^{(1)}$, $x \in \mathcal{G}_x^{(2)}$, and $\hat{u}_h^{1/2}(\cdot, y) \in \mathcal{M}_x^{(1)}$, $y \in \mathcal{G}_y^{(2)}$, are defined by

$$\begin{aligned}
&\tilde{u}_h^{1/2}(x, y) = g_0(x, y) + g(x, y), \quad y \in \mathcal{G}_y^{(1)},\\
&\alpha_1^{1/2}(x)\tilde{u}_h^{1/2}(x, a_2) - \beta_1^{1/2}(x)(\tilde{u}_h^{1/2})_y(x, a_2) = g_1^{1/2}(x),\\
&\alpha_5^{1/2}(x)\tilde{u}_h^{1/2}(x, c_2) + \beta_5^{1/2}(x)(\tilde{u}_h^{1/2})_y(x, c_2) = g_5^{1/2}(x)
\end{aligned}$$

and

$$\begin{aligned}
&\hat{u}_h^{1/2}(x, y) = g_0(x, y) + g(x, y), \quad x \in \mathcal{G}_x^{(1)},\\
&\alpha_3^{1/2}(y)\hat{u}_h^{1/2}(a_1, y) - \beta_3^{1/2}(y)(\hat{u}_h^{1/2})_x(a_1, y) = g_3^{1/2}(y),\\
&\alpha_6^{1/2}(y)\hat{u}_h^{1/2}(c_1, y) + \beta_6^{1/2}(y)(\hat{u}_h^{1/2})_x(c_1, y) = g_6^{1/2}(y),
\end{aligned}$$

where g is given in (2.5).

In the second stage of the scheme, corresponding to the time levels $t_{n+1/2}$ and t_{n+1}, we compute, respectively, $u_h^{n+1/2}$ along horizontal lines and u_h^{n+1} along vertical lines passing through the Gauss points (see Figure 3.1). Specifically, for $n = 0, \ldots, N_t - 1$, we use (2.8)–(2.9) with $(x, y) \in \mathcal{G} \cap \Omega$ to determine $u_h^{n+1}(x, \cdot) \in \mathcal{M}_y$, $x \in \mathcal{G}_x^{(1)}$, and $u_h^{n+1}(x, \cdot) \in \mathcal{M}_y^{(1)}$, $x \in \mathcal{G}_x^{(2)}$. The functions $u_h^{n+1/2}(\cdot, y) \in \mathcal{M}_x$, $y \in \mathcal{G}_y^{(1)}$, satisfy the boundary conditions (2.11). The functions $u_h^{n+1/2}(\cdot, y) \in \mathcal{M}_x^{(1)}$, $y \in \mathcal{G}_y^{(2)}$, satisfy the first equation of (2.11) and

$$\begin{aligned}
&\alpha_6^{n+1/2}(y)u_h^{n+1/2}(c_1, y) + \beta_6^{n+1/2}(y)(u_h^{n+1/2})_x(c_1, y)\\
&= (1/2)(g_{6,h}^{n+1} + g_{6,h}^n)(y) + (\tau/4)L_2^{n+1/2}(\overline{u}_h^{n+1/2}, \hat{u}_h^{n+1/2})(g_{6,h}^{n+1} - g_{6,h}^n)(c_1, y).
\end{aligned}$$

The functions $u_h^{n+1}(x, \cdot) \in \mathcal{M}_y$, $x \in \mathcal{G}_x^{(1)}$, satisfy the boundary conditions (2.13). The functions $u_h^{n+1}(x, \cdot) \in \mathcal{M}_y^{(1)}$, $x \in \mathcal{G}_x^{(2)}$, satisfy the first equation in (2.13) and

$$\alpha_5^{n+1}(x)u_h^{n+1}(x, c_2) + \beta_5^{n+1}(x)(u_h^{n+1})_y(x, c_2) = g_{5,h}^{n+1}(x).$$

As in section 2, for $n = 1, \dots, N_t - 1$, the functions $\tilde{u}_h^{n+1/2}(x, \cdot) \in \mathcal{M}_y$, $x \in \mathcal{G}_x^{(1)}$; $\tilde{u}_h^{n+1/2}(x, \cdot) \in \mathcal{M}_y^{(1)}$, $x \in \mathcal{G}_x^{(2)}$; $\hat{u}_h^{n+1/2}(\cdot, y) \in \mathcal{M}_x$, $y \in \mathcal{G}_y^{(1)}$; and $\hat{u}_h^{n+1/2}(\cdot, y) \in \mathcal{M}_x^{(1)}$, $y \in \mathcal{G}_y^{(2)}$, are defined by (2.10), and $\overline{u}_h^{n+1/2}(a_1, \cdot) \in \mathcal{M}_y$ is defined by (2.12) with $\gamma = a_1$. The functions $\overline{u}_h^{n+1/2}(b_1, \cdot) \in \mathcal{M}_y^{(1)}$, $\overline{u}_h^{n+1/2}(c_1, \cdot) \in \mathcal{M}_y^{(2)}$ are defined by

$$\begin{aligned}
&\overline{u}_h^{n+1/2}(b_1, y) = \hat{u}_h^{n+1/2}(b_1, y), \qquad y \in \mathcal{G}_y^{(1)},\\
&\alpha_1^{n+1/2}(b_1)\overline{u}_h^{n+1/2}(b_1, a_2) - \beta_1^{n+1/2}(b_1)(\overline{u}_h^{n+1/2})_y(b_1, a_2) = g_1^{n+1/2}(b_1),\\
&\alpha_5^{n+1/2}(b_1)\overline{u}_h^{n+1/2}(b_1, c_2) + \beta_5^{n+1/2}(b_1)(\overline{u}_h^{n+1/2})_y(b_1, c_2) = g_5^{n+1/2}(b_1)
\end{aligned}$$

and

$$\begin{aligned}
&\overline{u}_h^{n+1/2}(c_1, y) = \hat{u}_h^{n+1/2}(c_1, y), \qquad y \in \mathcal{G}_y^{(2)},\\
&\alpha_5^{n+1/2}(c_1)\overline{u}_h^{n+1/2}(c_1, c_2) - \beta_5^{n+1/2}(c_1)(\overline{u}_h^{n+1/2})_y(c_1, c_2) = g_5^{n+1/2}(c_1),\\
&\alpha_2^{n+1/2}(c_1)\overline{u}_h^{n+1/2}(c_1, b_2) + \beta_2^{n+1/2}(c_1)(\overline{u}_h^{n+1/2})_y(c_1, b_2) = g_2^{n+1/2}(c_1).
\end{aligned}$$

In the third stage we convert $u_h^{N_t}(x, \cdot) \in \mathcal{M}_y$, $x \in \mathcal{G}_x^{(1)}$, and $u_h^{N_t}(x, \cdot) \in \mathcal{M}_y^{(1)}$, $x \in \mathcal{G}_x^{(2)}$, into $u_h^I \in \mathcal{M}_x^{(1)} \otimes \mathcal{M}_y^{(2)}$ and $u_h^{II} \in \mathcal{M}_x \otimes \mathcal{M}_y^{(1)}$ which are to be approximations to $u(x, y, d)$ on $\overline{\Omega}_I = [a_1, c_1] \times [c_2, b_2]$ and $\overline{\Omega}_{II} = [a_1, b_1] \times [a_2, c_2]$, respectively. Specifically we require that

$$\begin{aligned}
&u_h^I(x, y) = u_h^{N_t}(x, y), \qquad x \in \mathcal{G}_x^{(1)}, \qquad y \in [c_2, b_2],\\
&\alpha_3^{N_t}(y)u_h^I(a_1, y) - \beta_3^{N_t}(y)(u_h^I)_x(a_1, y) = g_{3,h}^{N_t}(y),\\
&\alpha_6^{N_t}(y)u_h^I(c_1, y) + \beta_6^{N_t}(y)(u_h^I)_x(c_1, y) = g_{6,h}^{N_t}(y),
\end{aligned} \qquad y \in \mathcal{G}_y^{(2)} \cup \{c_2, b_2\}, \tag{3.2}$$

and

$$\begin{aligned}
&u_h^{II}(x, y) = u_h^{N_t}(x, y), \qquad x \in \mathcal{G}_x, \qquad y \in [a_2, c_2],\\
&\alpha_3^{N_t}(y)u_h^{II}(a_1, y) - \beta_3^{N_t}(y)(u_h^{II})_x(a_1, y) = g_{3,h}^{N_t}(y),\\
&\alpha_4^{N_t}(y)u_h^{II}(b_1, y) + \beta_4^{N_t}(y)(u_h^{II})_x(b_1, y) = g_{4,h}^{N_t}(y),
\end{aligned} \qquad y \in \mathcal{G}_y^{(1)} \cup \{a_2, c_2\}.$$

We note that, in general, $u_h^I(c_1, c_2) \neq u_h^{II}(c_1, c_2)$, and hence u_h^I and u_h^{II} do not match in a continuous fashion along the line segment $\{(x, c_2) : x \in [a_1, c_1]\}$. This is a consequence of the fact that if $v_h \in \mathcal{M}_x$ is the Gauss interpolant of v, then, in general, $v(x_i) \neq v_h(x_i)$, even though $v(x_i) - v_h(x_i)$ is expected to be close to 0.

With $\overline{\Omega}_I$ and $u_h^I \in \mathcal{M}_x^{(1)} \otimes \mathcal{M}_y^{(2)}$ as above, we describe below an alternative to the above conversion of $u_h^{N_t}$ into u_h^{II}. With $\overline{\Omega}_{II} = [a_1, c_1] \times [a_2, c_2]$ and $\overline{\Omega}_{III} = [c_1, b_1] \times [a_2, c_2]$, we seek approximations $u_h^{II} \in \mathcal{M}_x^{(1)} \otimes \mathcal{M}_y^{(1)}$ and $u_h^{III} \in \mathcal{M}_x^{(2)} \otimes \mathcal{M}_y^{(1)}$ to $u(x, y, d)$ on $\overline{\Omega}_{II}$ and $\overline{\Omega}_{III}$, respectively. First, for each $y \in \mathcal{G}_y^{(1)} \cup a_2$, we determine $v_h(\cdot, y) \in \mathcal{M}_x$ such that

$$\begin{aligned}
&v_h(x, y) = u_h^{N_t}(x, y), \qquad x \in \mathcal{G}_x,\\
&\alpha_3^{N_t}(y)v_h(a_1, y) - \beta_3^{N_t}(y)(v_h)_x(a_1, y) = g_{3,h}^{N_t}(y),\\
&\alpha_4^{N_t}(y)v_h(b_1, y) + \beta_4^{N_t}(y)(v_h)_x(b_1, y) = g_{4,h}^{N_t}(y).
\end{aligned}$$

Then we compute $z_h \in \mathcal{M}_y^{(1)}$ such that

$$z_h(y) = v_h(c_1, y), \qquad y \in \mathcal{G}_y^{(1)} \cup a_2,$$
$$z_h(c_2) = u_h^I(c_1, c_2).$$

Finally, we obtain the desired u_h^{II} and u_h^{III} from

$$u_h^{II}(x, y) = u_h^{N_t}(x, y), \qquad x \in \mathcal{G}_x^{(1)}, \qquad y \in [a_2, c_2],$$
$$\left.\begin{aligned} \alpha_3^{N_t}(y) u_h^{II}(a_1, y) - \beta_3^{N_t}(y)(u_h^{II})_x(a_1, y) &= g_{3,h}^{N_t}(y), \\ u_h^{II}(c_1, y) &= z_h(y), \end{aligned}\right\} \quad y \in \mathcal{G}_y^{(1)} \cup \{a_2, c_2\},$$

and

$$u_h^{III}(x, y) = u_h^{N_t}(x, y), \qquad x \in \mathcal{G}_x^{(2)}, \qquad y \in [a_2, c_2],$$
$$\left.\begin{aligned} u_h^{III}(c_1, y) &= z_h(y), \\ \alpha_4^{N_t}(y) u_h^{III}(b_1, y) + \beta_4^{N_t}(y)(u_h^{III})_x(b_1, y) &= g_{4,h}^{N_t}(y), \end{aligned}\right\} \quad y \in \mathcal{G}_y^{(1)} \cup \{a_2, c_2\}.$$

This approach guarantees that

$$u_h^I(x, c_2) = u_h^{II}(x, c_2), \qquad x \in [a_1, c_1],$$
$$u_h^{II}(c_1, y) = u_h^{III}(c_1, y), \qquad y \in [a_2, c_2],$$

which means that u_h^I, u_h^{II} are continuous across the interface $\overline{\Omega}_I \cap \overline{\Omega}_{II}$ and that u_h^{II}, u_h^{III} are continuous across the interface $\overline{\Omega}_{II} \cap \overline{\Omega}_{III}$.

Implementation of the ADI OSC scheme for the L-shaped region is similar to that for a rectangular region (see section 2.2). For example, finding $u_h^{n+1/2}(\cdot, y) \in \mathcal{M}_x$, $y \in \mathcal{G}_y^{(1)}$, and $u_h^{n+1/2}(\cdot, y) \in \mathcal{M}_x^{(1)}$, $y \in \mathcal{G}_y^{(2)}$, amounts to solving a collection of independent two-point OSC boundary value problems in the x-directions. Similarly, $u_h^{n+1}(x, \cdot) \in \mathcal{M}_y$, $x \in \mathcal{G}_x^{(1)}$, and $u_h^{n+1}(x, \cdot) \in \mathcal{M}_y^{(1)}$, $x \in \mathcal{G}_x^{(2)}$, are obtained by solving a collection of independent two-point OSC boundary value problems in the y-directions. In our implementation of the third stage of the ADI OSC scheme we use the original conversion of $u_h^{N_t}$ into u_h^I, u_h^{II} since its implementation is simpler than that of converting $u_h^{N_t}$ into u_h^I, u_h^{II}, u_h^{III}. Computing u_h^I and u_h^{II} involves performing two independent conversions on rectangles $\overline{\Omega}_I$ and $\overline{\Omega}_{II}$.

4. ADI schemes for rectangular polygons. For parabolic problems on rectangular polygons, ADI FD methods were considered in [18] and [19], and ADI FEG methods were presented in [12]. It should be noted that for a nonlinear parabolic problem, the second order in time method (2.6) in [12] is the three level ADI Laplace-modified scheme. Implementation of the scheme for Dirichlet boundary conditions uses a technique similar to that of [18] involving disintegrating difference operators. To describe possible complications in ADI FD methods on polygons, we consider the rectangular polygon Ω depicted in Figure 4.1. Suppose that Ω is partitioned with a stepsize h in the x- and y-directions, and that $x_k = kh$, $y_l = lh$ for $k, l = 0, \pm 1, \pm 2, \ldots$. Let Ω_h be the collection of all (k, l) such that $(x_k, y_l) \in \Omega$ and let $\partial\Omega_h$ be the collection of all (k, l) such that $(x_k, y_l) \in \partial\Omega$. Let $\partial\Omega_h^x$ be the collection of all (k, l) such that (x_k, y_l) is a point on a vertical side of $\partial\Omega$, excluding vertices but including the reentrant corners B and C of Ω.

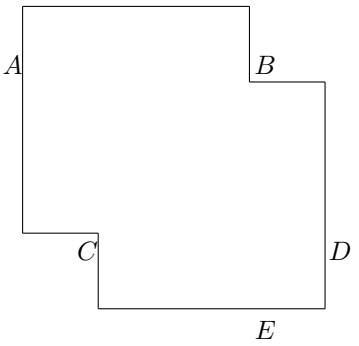


FIG. 4.1. *Rectangular polygon* Ω.

The ADI FD method for the linear problem

$$u_t - u_{xx} - u_{yy} = f(x,y,t), \qquad (x,y,t) \in \Omega \times [0,d],$$
$$u(x,y,0) = g_0(x,y), \quad (x,y) \in \overline{\Omega}, \qquad u(x,y,t) = g_1(x,y,t), \quad (x,y) \in \partial\Omega \times [0,d],$$

is defined for $n = 0, \ldots, N_t - 1$ and $(k,l) \in \Omega_h$ by

$$\frac{u_{k,l}^{n+1/2} - u_{k,l}^{n}}{0.5\tau} - \partial_x^2 u_{k,l}^{n+1/2} - \partial_y^2 u_{k,l}^{n} = f^{n+1/2}(x_k, y_l), \tag{4.1}$$

$$\frac{u_{k,l}^{n+1} - u_{k,l}^{n+1/2}}{0.5\tau} - \partial_x^2 u_{k,l}^{n+1/2} - \partial_y^2 u_{k,l}^{n+1} = f^{n+1/2}(x_k, y_l), \tag{4.2}$$

$$u_{k,l}^{n+1/2} = \frac{1}{2}\left(u_{k,l}^{n+1} + u_{k,l}^{n}\right) - \frac{\tau}{4}\partial_y^2\left(u_{k,l}^{n+1} - u_{k,l}^{n}\right), \qquad (k,l) \in \partial\Omega_h^x, \tag{4.3}$$

where

$$\partial_x^2 v_{k,l} = \frac{v_{k-1,l} - 2v_{k,l} + v_{k+1,l}}{h^2}, \qquad \partial_y^2 v_{k,l} = \frac{v_{k,l-1} - 2v_{k,l} + v_{k,l+1}}{h^2},$$
$$u_{k,l}^{0} = g_0(x_k, y_l), \qquad (k,l) \in \Omega_h \cup \partial\Omega_h,$$
$$u_{k,l}^{n} = g_1^n(x_k, y_l), \qquad n = 1, \ldots, N_t, \qquad (k,l) \in \partial\Omega_h. \tag{4.4}$$

Using (4.1), (4.3), and (4.4), we can compute $u_{k,l}^{n+1/2}$ along all horizontal partition lines except the lines AB and CD. We cannot compute $u_{k,l}^{n+1/2}$ along, for example, the line AB since we do not know u_{k^*,l^*}^{n+1} corresponding to the partition point directly below B and we cannot determine u_{k^*,l^*}^{n+1} by solving (4.2) along the line BE. (There is no such difficulty if Ω is an L-shaped region.) In contrast to the FD scheme, our ADI OSC scheme for Ω in Figure 4.1 does not have any such complications described above since it involves finding $u_h^{n+1/2}$ along horizontal lines passing through the Gauss points and since $L_2^{n+1/2}$ in (2.11) is a differential, rather than difference, operator.

For any rectangular polygon Ω, using our ADI OSC scheme, at each time level we can solve for $u_h^{n+1/2}$ along horizontal lines and then solve for u_h^{n+1} along vertical lines passing through the Gauss points as explained for an L-shaped region in section 3. Once the approximate solution at the final time level is obtained along vertical lines, it is converted into global piecewise polynomials on subrectangles in a similar way as described in section 3; for the polygon Ω in Figure 4.1, the subrectangles of $\overline{\Omega}$ would be $\overline{\Omega}_I$, $\overline{\Omega}_{II}$, and $\overline{\Omega}_{III}$ with the lines AB and CD as two interfaces.

5. Numerical results. For the rectangle Ω of section 2 with $a_1 = a_2 = 0$, $b_1 = b_2 = 1$, and the L-shaped region Ω of section 3 with $a_1 = a_2 = 0$, $b_1 = b_2 = 1$, and $c_1 = c_2 = 1/2$, we present numerical results obtained using our ADI OSC scheme. For $r = 3, 4, 5$ and several increasing values of integer N_i, we use the uniform partitions $\{x_k\}_{k=0}^{N_x^{(i)}} = \{y_l\}_{l=0}^{N_y^{(i)}}$ of $[0,1]$ with $N_x^{(i)} = N_y^{(i)} = N_i$. We give spatial errors in various norms and the corresponding convergence rates determined by the formula

$$\text{Convergence rate} \approx \frac{\log(\varepsilon_i/\varepsilon_{i+1})}{\log(h_i/h_{i+1})},$$

where the stepsize $h_i = 1/N_i$ and ε_i is the error corresponding to h_i. For the L-shaped region, $\epsilon_i = \max(\epsilon_i^I, \epsilon_i^{II})$, where ϵ_i^I and ϵ_i^{II} are the errors corresponding to $\Omega^I = [0, 0.5] \times [0.5, 1]$ and $\Omega^{II} = [0, 1] \times [0, 0.5]$, respectively. For all tests, we give the errors and the rates corresponding to the approximation of $u(\cdot, \cdot, d)$.

For the rectangle and the L-shaped region, we solve problem (1.1)–(1.3), (2.1), (2.2), (3.1), with $d = 1$, the exact solution

$$u(x, y, t) = e^{(1+t)^{-1}+xy}, \tag{5.1}$$

$$\begin{aligned}
&\alpha_1(x,t) = x + t + 2, && \beta_1(x,t) = x^2 + t^2,\\
&\alpha_2(x,t) = \log(x + t + 2), && \beta_2(x,t) = \sin(x + t),\\
&\alpha_3(y,t) = y + t, && \beta_3(y,t) = e^y + t^2 + 3,\\
&\alpha_4(y,t) = 1 + t\sin y, && \beta_4(y,t) = 3 + t\cos y,\\
&\alpha_5(x,t) = \alpha_2(x,t), && \beta_5(x,t) = \beta_2(x,t),\\
&\alpha_6(y,t) = \alpha_4(y,t), && \beta_6(y,t) = \beta_4(y,t),
\end{aligned} \tag{5.2}$$

$$\begin{aligned}
p_1(x, y, t, v, w, z) &= (1 + x^2 + y^2 + t^2)/4 + \tan^{-1}(v^2 + 1) + \cos(w + z) + 2,\\
p_2(x, y, t, v, w, z) &= \sin(x + y) + (t + 4)/3 + \cos v + (1 + \sqrt{w^2 + z^2})^{-1} + 1,
\end{aligned}$$

and

$$f(x, y, t, v, w, z) = \tilde{f}(x, y, t) + v\sin(w) + z^2/(1 + z^2),$$

where

$$\tilde{f} = u_t - p_1(x, y, t, u, u_x, u_y)u_{xx} - p_2(x, y, t, u, u_x, u_y)u_{yy} - u\sin(u_x) - u_y^2/(1 + u_y^2).$$

In Tables 5.1 and 5.2, we give the L^∞ and L^2 errors and the corresponding rates for the rectangle and the L-shaped region, respectively. The boundary conditions are approximated using Gauss interpolants of $g_1, \ldots, g_6$. We select $\tau_i = h_i^{(r+1)/2}$ ($N_t^{(i)} = N_i^{(r+1)/2}$) to demonstrate numerically the expected second order accuracy in t and $r+1$ order accuracy in space. In Table 5.1, we also give CPU times (in seconds) which confirm the theoretical analysis of section 2.2, showing that the total cost of our scheme is $O(r^4 N_i^{2+(r+1)/2})$. The time results also indicate that to achieve a given accuracy, the scheme takes less time for a larger value of r.

In Tables 5.3 and 5.4, we give the L^∞ errors in u_x, u_y, H^1 errors, and the corresponding rates for the rectangle and the L-shaped region, respectively. The boundary conditions are approximated using Gauss interpolants of $g_1, \ldots, g_6$. We choose $\tau_i = h_i^{r/2}$ ($N_t^{(i)} = N_i^{r/2}$) to demonstrate numerically the expected second order accuracy in t and r order accuracy in space.

TABLE 5.1
Rectangle; L^∞, L^2 errors in $u(\cdot,\cdot,d)$; CPU time.

r	N_i	L^∞ error	Rate	L^2 error	Rate	CPU time
3	4	0.119–02		0.649–03		0.019
	16	0.450–05	4.024	0.248–05	4.017	3.528
	36	0.176–06	3.999	0.967–07	4.001	89.922
	64	0.176–07	3.999	0.968–08	4.000	873.877
	100	0.295–08	4.003	0.162–08	4.005	5336.772
4	4	0.293–03		0.162–03		0.087
	16	0.280–06	5.015	0.156–06	5.008	37.740
	36	0.485–08	5.000	0.271–08	5.000	1498.947
	64	0.274–09	4.993	0.153–09	4.990	19163.175
5	4	0.718–04		0.401–04		0.357
	16	0.175–07	6.003	0.975–08	6.003	324.394
	36	0.136–09	5.985	0.764–10	5.980	19125.276

TABLE 5.2
L-shaped region; L^∞, L^2 errors in $u(\cdot,\cdot,d)$.

r	N_i	L^∞ error	Rate	L^2 error	Rate
3	4	0.110–02		0.416–03	
	16	0.481–05	3.915	0.162–05	4.000
	36	0.192–06	3.971	0.633–07	4.000
4	4	0.292–03		0.104–03	
	16	0.304–06	4.954	0.101–06	5.001
	36	0.533–08	4.986	0.176–08	5.000
5	4	0.753–04		0.260–04	
	16	0.191–07	5.972	0.634–08	6.000
	36	0.148–09	5.997	0.492–10	5.993

TABLE 5.3
Rectangle; L^∞ errors in $u_x(\cdot,\cdot,d)$, $u_y(\cdot,\cdot,d)$; H^1 error in $u(\cdot,\cdot,d)$.

r	N_i	L^∞ error in u_x	Rate	L^∞ error in u_y	Rate	H^1 error	Rate
3	4	0.163–01		0.106–01		0.761–02	
	16	0.343–03	2.787	0.186–03	2.918	0.130–03	2.937
	36	0.321–04	2.923	0.164–04	2.995	0.115–04	2.984
4	4	0.475–02		0.288–02		0.204–02	
	16	0.220–04	3.875	0.113–04	3.994	0.820–05	3.981
	36	0.893–06	3.954	0.442–06	3.999	0.322–06	3.991
5	4	0.124–02		0.727–03		0.517–03	
	16	0.140–05	4.897	0.707–06	5.003	0.515–06	4.986
	36	0.249–07	4.965	0.123–07	4.998	0.897–08	4.995

TABLE 5.4
L-shaped region; L^∞ errors in $u_x(\cdot,\cdot,d)$, $u_y(\cdot,\cdot,d)$; H^1 error in $u(\cdot,\cdot,d)$.

r	N_i	L^∞ error in u_x	Rate	L^∞ error in u_y	Rate	H^1 error	Rate
3	4	0.172–01		0.787–02		0.646–02	
	16	0.340–03	2.831	0.205–03	2.631	0.107–03	2.959
	36	0.370–04	2.736	0.234–04	2.676	0.952–05	2.982
4	4	0.471–02		0.252–02		0.167–02	
	16	0.249–04	3.778	0.157–04	3.665	0.676–05	3.974
	36	0.127–05	3.667	0.797–06	3.674	0.266–06	3.991
5	4	0.123–02		0.752–03		0.425–03	
	16	0.182–05	4.700	0.115–05	4.678	0.425–06	4.983
	36	0.411–07	4.675	0.260–07	4.668	0.740–08	4.995

TABLE 5.5
Rectangle; nodal errors in $u(\cdot,\cdot,d)$, $u_x(\cdot,\cdot,d)$, $u_y(\cdot,\cdot,d)$.

r	N_i	Nodal error in u	Rate	Nodal error in u_x	Rate	Nodal error in u_y	Rate
3	16	0.470–05		0.155–04		0.114–04	
	32	0.294–06	4.000	0.107–05	3.851	0.717–06	3.995
	64	0.184–07	3.999	0.726–07	3.884	0.449–07	3.998
4	8	0.109–05		0.371–05		0.277–05	
	16	0.175–07	5.963	0.699–07	5.728	0.427–07	6.019
	32	0.274–09	5.995	0.120–08	5.869	0.678–09	5.977
5	4	0.428–05		0.881–05		0.103–04	
	8	0.170–07	7.975	0.527–07	7.384	0.440–07	7.876
	16	0.684–10	7.957	0.253–09	7.704	0.173–09	7.992

TABLE 5.6
L-shaped region; nodal errors in $u(\cdot,\cdot,d)$, $u_x(\cdot,\cdot,d)$, $u_y(\cdot,\cdot,d)$.

r	N_i	Nodal error in u	Rate	Nodal error in u_x	Rate	Nodal error in u_y	Rate
3	16	0.470–05		0.204–04		0.128–04	
	32	0.300–06	3.971	0.160–05	3.672	1.000–06	3.673
	64	0.189–07	3.985	0.126–06	3.676	0.786–07	3.669
4	8	0.118–05		0.500–05		0.307–05	
	16	0.190–07	5.955	0.981–07	5.672	0.603–07	5.670
	32	0.299–09	5.986	0.192–08	5.674	0.119–08	5.664
5	4	0.464–05		0.186–04		0.114–04	
	8	0.187–07	7.956	0.912–07	7.670	0.559–07	7.672
	16	0.746–10	7.970	0.447–09	7.671	0.275–09	7.668

In Tables 5.5 and 5.6, we give the maximum absolute value errors in $u(x_k, y_l, d)$, $u_x(x_k, y_l, d)$, $u_y(x_k, y_l, d)$, $k, l = 0, \ldots, N_i$. We approximate the boundary conditions using Hermite interpolants of $g_1, \ldots, g_6$. We take $\tau_i = h_i^{r-1}$ ($N_t^{(i)} = N_i^{r-1}$) to demonstrate numerically the expected second order accuracy in t and $2r-2$ order accuracy in space. Such an order of accuracy in space indicates the phenomenon of superconvergence.

We point out that for the L-shaped region, the L^∞ rates for u_x and u_y are both approximately between $r-1/2$ and r (cf. Table 5.4). Also in the case of the L-shaped region, the phenomenon of superconvergence is observed for the maximum norm error in u at the nodes, whereas the rates of convergence for the maximum norm error in u_x and u_y at the nodes are both between $2r-5/2$ and $2r-2$ (cf. Table 5.6). These results are expected for the L-shaped region and are consistent with the observed results for elliptic problems [3].

In the last example, we use the modification of our scheme described in Remark 2.9 to solve an equation of the form (1.4). In fact, for the rectangle Ω with $a_1 = a_2 = 0$, $b_1 = b_2 = 1$, we solve problem (1.5), (1.2), (2.1), (2.2) with the exact solution (5.1), α_i and β_i, $i = 1, 2, 3, 4$, of (5.2),

$$q_1(x, y, t, u, u_x, u_y) = \tan^{-1}\left(1 + \sqrt{u_x^2 + u_y^2}\right) u_x,$$

$$q_2(x, y, t, u, u_x, u_y) = \tan^{-1}\left(1 + \sqrt{u_x^2 + u_y^2}\right) u_y,$$

and

$$f(x, y, t, v, w, z) = u_t - (q_1(x, y, t, u, u_x, u_y))_x - (q_2(x, y, t, u, u_x, u_y))_y.$$

TABLE 5.7
Rectangle; L^∞, L^2 errors in $u(\cdot,\cdot,d)$.

r	N_i	L^∞ error	Rate	L^2 error	Rate
3	4	0.731–03		0.516–03	
	16	0.290–05	3.988	0.205–05	3.987
	36	0.113–06	3.999	0.803–07	3.998
4	4	0.184–03		0.131–03	
	16	0.181–06	4.996	0.129–06	4.995
	36	0.314–08	4.999	0.224–08	4.999
5	4	0.462–04		0.330–04	
	16	0.113–07	5.998	0.809–08	5.997
	36	0.887–10	5.978	0.636–10	5.975

(The elliptic operator in this problem is $A_4 u$ on page 284 of [29].) In order to apply the modified scheme, (1.5) is first transformed, as discussed in section 1 (cf. (1.6)), into an equation of the form (1.4). The boundary conditions are approximated using Gauss interpolants of $g_1, \ldots, g_4$. In Table 5.7, for $\tau_i = h_i^{(r+1)/2}$ ($N_t^{(i)} = N_i^{(r+1)/2}$), we give the L^∞ and L^2 errors and the corresponding rates. The results confirm the expected second order accuracy in t and $r+1$ order accuracy in space.

REFERENCES

[1] V. K. BANGIA, C. BENNETT, A. REYNOLDS, R. RAGHAVAN, AND G. THOMAS, *Alternating direction collocation methods for simulating reservoir performance*, in Proceedings of the 53rd SPE Fall Technical Conference and Exhibition, Houston, TX, 1978, Paper SPE 7414.

[2] B. BIALECKI, *Convergence analysis of orthogonal spline collocation for elliptic boundary value problems*, SIAM J. Numer. Anal., 35 (1998), pp. 617–631.

[3] B. BIALECKI AND M. DRYJA, *A nonoverlapping domain decomposition method for orthogonal spline collocation problems*, SIAM J. Numer. Anal., 41 (2003), pp. 1709–1728.

[4] B. BIALECKI AND R. I. FERNANDES, *Orthogonal spline collocation Laplace-modified and alternating-direction methods for parabolic problems on rectangles*, Math. Comp., 60 (1993), pp. 545–573.

[5] B. BIALECKI AND R. I. FERNANDES, *An orthogonal spline collocation alternating direction implicit Crank–Nicolson method for linear parabolic problems on rectangles*, SIAM J. Numer. Anal., 36 (1999), pp. 1414–1434.

[6] J. H. BRAMBLE, R. E. EWING, AND G. LI, *Alternating direction multistep methods for parabolic problems–iterative stabilization*, SIAM J. Numer. Anal., 26 (1989), pp. 904–919.

[7] H. CHENG, P. LIN, Q. SHENG, AND R. C. E. TAN, *Solving degenerate reaction-diffusion equations via variable step Peaceman–Rachford splitting*, SIAM J. Sci. Comput., 25 (2003), pp. 1273–1292.

[8] C. CLAVERO AND J. L. GRACIA, *A uniformly convergent alternating direction HODIE finite difference scheme for 2D time-dependent convection-diffusion problems*, IMA J. Numer. Anal., 26 (2006), pp. 155–172.

[9] C. DE BOOR, *A Practical Guide to Splines*, revised ed., Appl. Math. Sci. 27, Springer-Verlag, New York, 2001.

[10] C. DE BOOR, *Package for calculating with B-splines*, SIAM J. Numer. Anal., 14 (1977), pp. 441–472.

[11] J. E. DENDY, JR., *Alternating direction methods for nonlinear time-dependent problems*, SIAM J. Numer. Anal., 14 (1977), pp. 313–326.

[12] J. E. DENDY, JR., AND G. FAIRWEATHER, *Alternating-direction Galerkin methods for parabolic and hyperbolic problems on rectangular polygons*, SIAM J. Numer. Anal., 12 (1975), pp. 144–163.

[13] J. C. DIAZ, G. FAIRWEATHER, AND P. KEAST, *FORTRAN packages for solving certain almost block diagonal linear systems by modified alternate row and column elimination*, ACM Trans. Math. Software, 9 (1983), pp. 358–375.

[14] J. C. Diaz, G. Fairweather, and P. Keast, *Algorithm* 603*, COLROW, and ARCECO: FORTRAN packages for solving certain almost block diagonal linear systems by modified alternate row and column elimination*, ACM Trans. Math. Software, 9 (1983), pp. 376–380.
[15] J. Douglas, Jr., and T. Dupont, *Collocation Methods for Parabolic Equations in a Single Space Variable*, Lecture Notes in Math. 385, Springer-Verlag, New York, 1974.
[16] J. Douglas, Jr., and T. Dupont, *Galerkin methods for parabolic equations*, SIAM J. Numer. Anal., 7 (1970), pp. 575–626.
[17] J. Douglas, Jr., and T. Dupont, *Alternating-direction Galerkin methods on rectangles*, in Proceedings of the Symposium on Numerical Solution of Partial Differential Equations, H. B. Habbard, ed., Academic Press, New York, 1971, pp. 133–214.
[18] E. G. Dyakonov, *On the application of disintegrating difference operators*, U.S.S.R. Comput. Math. Math. Phys., 3 (1963), pp. 511–515.
[19] G. Fairweather and A. R. Mitchell, *A new computational procedure for A.D.I. methods*, SIAM J. Numer. Anal., 4 (1967), pp. 163–170.
[20] L. J. Hayes, *Galerkin alternating-direction methods for nonrectangular regions using patch approximations*, SIAM J. Numer. Anal., 18 (1981), pp. 627–643.
[21] P. Keast, G. Fairweather, and J. C. Diaz, *A computational study of finite element methods for second order linear two-point boundary value problems*, Math. Comp., 40 (1983), pp. 499–518.
[22] W. Hundsdorfer and J. Verwer, *Numerical Solution of Time-Dependent Advection-Diffusion-Reaction Equations*, Springer-Verlag, Berlin, 2003.
[23] S. Karaa, *An accurate LOD scheme for two-dimensional parabolic problems*, Appl. Math. Comput., 170 (2005), pp. 886–894.
[24] O. A. Ladyženskaja, V. A. Solonnikova, and N. N. Uralceva, *Linear and Quasi-linear Equations of Parabolic Type*, Transl. Math. Monogr. 23, AMS, Providence, RI, 1988.
[25] D. W. Peaceman and H. H. Rachford, Jr., *The numerical solution of parabolic and elliptic differential equations*, SIAM J. Appl. Math., 3 (1955), pp. 28–41.
[26] D. W. Peaceman, *Fundamentals of Numerical Reservoir Simulation*, Elsevier, Amsterdam, 1977.
[27] V. Thomée, *Galerkin Finite Element Methods for Parabolic Problems*, Springer-Verlag, Berlin, 1997.
[28] E. Zeidler, *Nonlinear Functional Analysis and Its Applications,* II*/B: Nonlinear Monotone Operators*, Springer-Verlag, New York, 1990.
[29] A. Ženišek, *Nonlinear Elliptic and Evolution Problems and Their Finite Element Approximations*, Academic Press, London, 1990.

SIAM J. SCI. COMPUT.
Vol. 28, No. 3, pp. 1078–1099
© 2006 Society for Industrial and Applied Mathematics

SIMULTANEOUS PSEUDO-TIMESTEPPING FOR AERODYNAMIC SHAPE OPTIMIZATION PROBLEMS WITH STATE CONSTRAINTS*

S. B. HAZRA† AND V. SCHULZ†

Abstract. This paper presents a numerical method for constrained aerodynamic shape optimization problems. It is based on simultaneous pseudo-timestepping in which stationary states are obtained by solving the pseudo-stationary system of equations representing the state, costate, and design equations. The main advantages of this method are that it blends in nicely with a previously existing pseudo-timestepping method for the states only, and that a preconditioner can be used for convergence acceleration which stems from the reduced SQP methods. Design examples for drag reduction with constant lift and drag reduction with constant pitching moment for an RAE2822 airfoil are included. The overall cost of computation is 4–8 times that of the forward simulation runs depending on the grid size and the number of design parameters.

Key words. shape optimization, simultaneous pseudo-timestepping, Euler equations, preconditioner, reduced SQP methods, one-shot method, airfoil

AMS subject classifications. 49J20, 65L20, 65N20

DOI. 10.1137/05062442X

1. Introduction.

With the increase in efficiency of computer technology and availability of robust numerical methods, automated aerodynamic shape optimization is currently in practice. Using present-day computational fluid dynamics (CFD) methods, it is possible to compute the flow field very accurately around a complete aircraft in a few hours' time. Also, numerical algorithms for optimization methods are making significant improvements. Therefore, it is one of the goals of researchers in the field of aerodynamic design to combine both so that multidisciplinary design can be carried out in a reasonable amount of time using the current computational facilities. Considerable progress in this direction has been made in the past few years.

In control theory based design methods, as presented in [27, 28, 29, 30, 18], a control function (parameterized) is optimized to achieve the goal or objective subject to given constraints which relate the control function and the objective or cost function. Among various methods used for this purpose, gradient based optimization methods are used in most practical applications. In this method, the gradient information of the cost function with respect to the parameters, also known as sensitivities, is used to find a direction toward achieving the goal. Once the direction is found, a step, determined by some means which is known as line search or globalization strategy, toward this direction is taken in each optimization iteration. This process is continued until convergence.

The overall process is difficult due to various reasons. The constraints are nonlinear partial differential equations, known as state equations (e.g., Euler equations or Reynolds-averaged Navier–Stokes equations together with turbulence modeling). Currently available CFD is a remedy to this difficulty. Another difficulty lies in the

*Received by the editors November 29, 2005; accepted for publication (in revised form) February 7, 2006; published electronically July 7, 2006.
http://www.siam.org/journals/sisc/28-3/62442.html
†Department of Mathematics, University of Trier, D-54286 Trier, Germany (hazra@uni-trier.de, volker.schulz@uni-trier.de). The research of the second author was supported by the German Federal Ministry of Economics and Technology through project "MEGADESIGN" under contract 20A03020F.

numerical optimization algorithm. If a gradient based iterative method, e.g., steepest descent, conjugate gradient, or quasi-Newton method, is used, then several requirements are to be fulfilled for convergence of the method. The sensitivity derivatives required for this method are computed using the continuous adjoint approach which has advantage over the finite difference approach. In this approach, another set of partial differential equations, known as adjoint or costate equations, are to be solved. In each iteration of this method one has to solve the state and costate equations sufficiently accurately. This has to be repeated several times until the convergence of the optimization algorithm. Therefore, in spite of using high fidelity CFD, the overall cost is quite high to get an optimal solution. Computational results based on these methodologies have been presented in [19, 20, 32, 33, 10, 12] on structured grids. An application of this method on unstructured grids has been presented in [1]. This approach with less accurate state and costate solutions has been performed in Iollo, Kuruvila, and Ta'asan [31].

In [15] we proposed a new method for solving such problems using simultaneous pseudo-timestepping. It is well known that there is a strong correlation between iterative methods and pseudo-timestepping [24, 7] which has been exploited for construction of a timestepping method in the spirit of reduced sequential quadratic programming (RSQP) methods. This formulation is advantageous since the steady-state flow is obtained by integrating the pseudo-unsteady Euler (or Navier–Stokes) equations in this problem class. Therefore, one can use the same timestepping scheme for the whole set of equations, and preconditioners can be used to accelerate the convergence. Certain preconditioners are equivalent to SQP methods, whose mathematical background is well studied. In [14, 16] we have implemented that method for the shape design example using Euler equations.

Practical shape design problems involve additional state constraints, e.g., shape optimization of aircraft for drag reduction with constant lift. Applications of traditional gradient methods to such problems are made in, among others, [5, 6, 12]. The computational effort required is about 40 forward simulation runs and 9 adjoint runs for each of the objective functions and of the additional constraints. The methods presented in [15, 14, 16] cannot be extended easily to this larger and more important problem class. Every aspect has to be revised, and additional ideas based on the concept of partially reduced SQP methods [35] come into play, as will be seen later. The basic solution idea is to project the unconstrained design velocity onto the tangent space of the state constraints by solving a quadratic programming (QP) problem involving the reduced Hessian and reduced gradients. Application examples for drag reduction with constant lift and drag reduction with constant pitching moment for an RAE2822 airfoil are included.

The number of iterations required for the full optimization problem is 4–8 times that of the analysis problem. This means a drastic reduction of the computational cost compared to usual gradient methods.

The preconditioner based on RSQP methods causes scaling or damping of the gradients which are used to find a direction. In the convergence histories of all the computations, one observes a linear convergence with respect to the objective; starting from an almost feasible initial guess, each iteration step in the primal, adjoint, and design space is so small that the process truly reflects a continuous behavior. Each integration step of the preconditioned pseudo-time formulation is contracting enough so that no additional globalization strategy is necessary. The preconditioning in the design space gives short enough integration steps that the iterates "surf" along

an almost feasible manifold toward optimality. We do not have theoretical proof of convergence of the method as yet. This will be taken up in the near future. Here we provide computational results for two different reduced Hessian approximations, showing evidence that a better approximation leads to faster convergence. Also, computational results are provided for different numbers of design parameters, different state constraints, and different grid sizes. The computational results show evidence of numerical convergence in all cases.

The paper is organized as follows. We explain the unconstrained simultaneous pseudo-timestepping method in the next section. Section 3 presents the problem with additional constraints and its reduction to preconditioned pseudo-unsteady optimization problems. In section 4, we present the state, costate, and design equations, their discretization, geometry parameterization, gradient computation, and grid perturbation methods. Numerical results are presented in section 5. We draw our conclusions in section 6.

2. Simultaneous pseudo-timestepping for optimization problems. The optimization problem that we are dealing with in this study is of the class of optimal control problems that also includes the subclass of shape design problems. These problems can be written in abstract form as

$$\begin{aligned} &\min\ I(w,q) \\ &\text{s.t.}\quad c(w,q)=0, \end{aligned} \tag{1}$$

where $(w,q) \in X \times P$ (X, P are appropriate Hilbert spaces), and where $I : X \times P \to \mathbb{R}$ and $c : X \times P \to Y$ are twice Frechet-differentiable (with Y an appropriate Banach space). The Jacobian, $J = \frac{\partial c}{\partial w}$, is assumed to be invertible. Here, the equation $c(w,q) = 0$ represents the steady-state flow equations (in our case, Euler equations) together with boundary conditions, w is the vector of state variables, and q is the vector of design variables. The objective $I(w,q)$ is the drag of an airfoil for the purposes of this paper. In what follows, we denote by the notation $()^\top$ the transpose of the vector or the operator. In an optimal control setting, adjoint variables are from the dual space, and the notation should reflect this. However, since ultimately we are dealing with discretized magnitudes, we do not distinguish between finite- and infinite-dimensional adjoints and always use the same notation.

The necessary optimality conditions can be formulated using the Lagrangian functional

$$L(w,q,\lambda) = I(w,q) - \lambda^\top c(w,q), \tag{2}$$

where λ is the Lagrange multiplier or the adjoint variable from the dual Hilbert space. If $\hat{z} = (\hat{w}, \hat{q})$ is a minimum, then there exists a $\hat{\lambda}$ such that

$$\nabla_z L(\hat{z}, \hat{\lambda}) = \nabla_z I(\hat{z}) - \hat{\lambda}^\top \nabla_z c(\hat{z}) = 0. \tag{3}$$

Hence, the necessary optimality conditions are

$$c(w,q) = 0 \quad \text{(state equation)}, \tag{4a}$$

$$\nabla_w L(w,q,\lambda) = 0 \quad \text{(costate equation)}, \tag{4b}$$

$$\nabla_q L(w,q,\lambda) = 0 \quad \text{(design equation)}. \tag{4c}$$

Traditional so-called black-box methods, such as the steepest descent method or the conjugate gradient method, involve the solution of the state and costate equations at

each update of the design variables. These methods act only in the design space and assume that the state and costate equations are solved sufficiently accurately. This leads to very high computational cost for these methods.

In order to determine the solution of this system by use of a simultaneous pseudo-timestepping method we look for steady-state solutions of the following pseudo-time embedded evolution equations:

$$\begin{aligned} \frac{dw}{dt} + c(w,q) &= 0, \\ \frac{d\lambda}{dt} + \nabla_w L(w,q,\lambda) &= 0, \\ \frac{dq}{dt} + \nabla_q L(w,q,\lambda) &= 0. \end{aligned} \tag{5}$$

This formulation is advantageous since the steady-state flow is obtained by integrating the pseudo-unsteady Euler (or Navier–Stokes) equations in this problem class. However, this pseudo-time embedded system is usually a stiff system of ODEs. Therefore explicit timestepping schemes (which are used in most applications of this problem class) may converge very slowly or may even diverge. In order to accelerate convergence, this system needs preconditioning. The preconditioner that we use stems from RSQP methods as discussed below.

2.1. Preconditioners. For the solution of problem (1), we recall a straightforward RSQP method. A detailed discussion of this approach can be found in [34, 35]. Here we present the outline of the method.

RSQP methods are related to projected Lagrangian methods (cf. [13]) and are most advantageous in cases where the number of degrees of freedom (here the design parameters) is small compared to the number of state variables. The variable steps in each iteration can be considered linear combinations of steps toward optimality and steps toward feasibility of the constraints. The constraints are linearized by a Taylor expansion up to first order terms, so that all steps toward optimality lie in the tangent space of c of the current approximation (w,q):

$$T_{(w,q)}c = \left\{ \begin{pmatrix} \Delta w \\ \Delta q \end{pmatrix} \middle| \, c(w,q) + J(w,q)\,\Delta w + \frac{\partial c}{\partial q}(w,q)\,\Delta q = 0 \right\}.$$

The optimization problem is projected to this tangent space and approximated by a quadratic problem with the projected Hessian of the Lagrangian given by (2).

In this method, the computationally expensive operation of computing the exact projected Hessian is typically avoided by using appropriate update formulas. It can be proved that under mild conditions the reduced SQP method with the update formulas for the reduced Hessian shows superlinear local convergence properties (see [34]).

In the above method, it is necessary to invert the Jacobian J of the constraints. In many cases, that is not a viable approach. The following considerations allow one to replace J with an approximate operator A which is invertible with comparatively low cost.

To this end, one employs an inexact RSQP method as introduced in [35]. Although it is not inverted, the Jacobian is still used for the computation of the correct adjoint variables and the correct state increments in the sense of defect correcting iterations. The algorithm of this method reads as follows.

ALGORITHM 1. The RSQP method with an approximate Jacobian.

(0) Set $k := 0$, $\lambda_0 = 0$; start at some initial guess w_0, q_0.
(1) Compute the increment of the adjoint variables from the linear system with the adjoint operator $A^\top$:
$A^\top(w_k, q_k)\,\Delta\lambda_k := \nabla_w I(w_k, q_k) - J^\top(w_k, q_k)\,\lambda_k$;
compute the reduced gradient
$\gamma_k := \nabla_q I(w_k, q_k) - (\frac{\partial c}{\partial q}(w_k, q_k))^\top(\lambda_k + \Delta\lambda_k)$;
determine some approximation B_k of the projected Hessian of the Lagrangian.
(2) Solve $B_k \Delta q_k = -\gamma_k$.
(3) Compute step on w from the linear system
$A(w_k, q_k)\,\Delta w_k := -\frac{\partial c}{\partial q}(w_k, q_k)\,\Delta q_k - c(w_k, q_k)$.
(4) Set $w_{k+1} := w_k + \Delta w_k$, $q_{k+1} := q_k + \Delta q_k$, and $\lambda_{k+1} = \lambda_k + \Delta\lambda_k$.
(5) $k := k + 1$; go to (1) until convergence.

A step of this method can also be interpreted as an approximate Newton step for the necessary conditions of finding the extremum of problem (1), since the updates of the variables are computed according to the linear system

$$\begin{pmatrix} 0 & 0 & A^\top \\ 0 & B & \left(\frac{\partial c}{\partial q}\right)^\top \\ A & \frac{\partial c}{\partial q} & 0 \end{pmatrix} \begin{pmatrix} \Delta w \\ \Delta q \\ \Delta\lambda \end{pmatrix} = \begin{pmatrix} -\nabla_w L \\ -\nabla_q L \\ -c \end{pmatrix}. \tag{6}$$

This is the basic formulation of the inexact RSQP methods that we use in the subsequent sections.

2.2. Preconditioned system for pseudo-timestepping in optimization problems. The necessary conditions (4a)–(4c) for optimization can be considered as an overall nonlinear equation,

$$F(w, \lambda, q) = \begin{pmatrix} \nabla_w L(w, \lambda, q) \\ \nabla_q L(w, \lambda, q) \\ c(w, \lambda, q) \end{pmatrix} = 0,$$

which must be solved. In this way, it would be possible to perform an optimization strategy in a consistent way with the timestepping method.

In our implementation, we use the inverse of the matrix in (6) as a preconditioner for the timestepping process. The pseudo-time embedded system of ODEs that we consider is

$$\begin{pmatrix} \dot{w} \\ \dot{q} \\ \dot{\lambda} \end{pmatrix} = K \begin{pmatrix} -\nabla_w L \\ -\nabla_q L \\ -c \end{pmatrix}, \tag{7}$$

where

$$K = \begin{bmatrix} 0 & 0 & A^\top \\ 0 & B & \left(\frac{\partial c}{\partial q}\right)^\top \\ A & \frac{\partial c}{\partial q} & 0 \end{bmatrix}^{-1} = \begin{bmatrix} A^{-1}\frac{\partial c}{\partial q}B^{-1}\left(\frac{\partial c}{\partial q}\right)^\top A^{-\top} & -A^{-1}\frac{\partial c}{\partial q}B^{-1} & A^{-1} \\ -B^{-1}\left(\frac{\partial c}{\partial q}\right)^\top A^{-\top} & B^{-1} & 0 \\ A^{-\top} & 0 & 0 \end{bmatrix}.$$

This seems natural since (6) can be considered as an explicit Euler discretization for the corresponding timestepping that we envision. Also, due to its block structure, it is computationally inexpensive. The preconditioner employed is similar to the preconditioners for KKT systems discussed in [3, 2] in the context of Krylov subspace methods and in [4] in the context of Lagrange–Newton–Krylov–Schur methods.

Within the inexact RSQP preconditioner, one has to look for an appropriate approximation of the reduced Hessian, B. In particular, when dealing with partial differential equations constituting the state equations, the reduced Hessian can often be expressed as a pseudo-differential operator, the symbol of which can be computed and exploited for preconditioning purposes as in [15]. Here we use an update formula involving second order information similar to BFGS updates, as explained in section 5.

3. Scalar state constraints. Adding a scalar state constraint to problem (1) results in

$$\begin{aligned} &\min\ I(w,q) \\ &\text{s.t.}\quad c(w,q)=0, \\ &\qquad h(w,q)=0,\ \in\mathbb{R}. \end{aligned} \tag{8}$$

Typically, in practical applications, the additional constraints are of the form

$$h(w,q)\geq 0$$

and represent the validity region of the model or the design construction. For the sake of simplicity in presentation we discuss only equality constraints, which is the case in our application. Inequality constraints can be handled by an active set strategy (see, for example, [8]), which is trivial for a scalar inequality. More difficult constraints, e.g., constraints in contact problems, which are not scalar valued, are beyond the scope of this paper.

One approximate Newton step for necessary conditions of problem (8) corresponding to an RSQP method is derived from the following system of equations:

$$\begin{pmatrix} 0 & 0 & \left(\dfrac{\partial h}{\partial w}\right)^{\top} & A^{\top} \\ 0 & B & \left(\dfrac{\partial h}{\partial q}\right)^{\top} & \left(\dfrac{\partial c}{\partial q}\right)^{\top} \\ \dfrac{\partial h}{\partial w} & \dfrac{\partial h}{\partial q} & 0 & 0 \\ A & \dfrac{\partial c}{\partial q} & 0 & 0 \end{pmatrix} \begin{pmatrix} \Delta w \\ \Delta q \\ \Delta\mu \\ \Delta\lambda \end{pmatrix} = \begin{pmatrix} -\nabla_w L \\ -\nabla_q L \\ -h \\ -c \end{pmatrix}, \tag{9}$$

where μ is the Lagrange multiplier corresponding to the additional state constraint $h(w,q)$. In the following we use Gaussian elimination for partial reduction of this system.

LEMMA 3.1.

$$\begin{pmatrix} B & g_h \\ g_h^{\top} & 0 \end{pmatrix} \begin{pmatrix} \Delta q \\ \Delta\mu \end{pmatrix} = \begin{pmatrix} -\nabla_q L + \left(\dfrac{\partial c}{\partial q}\right)^{\top} A^{-\top}\nabla_w L \\ -h + \dfrac{\partial h}{\partial w}A^{-1}c \end{pmatrix}, \tag{10}$$

[illegible] $g_h = (\frac{\partial h}{\partial q} - \frac{\partial h}{\partial w} A^{-1} \frac{\partial c}{\partial q})^\top$ [illegible] $h(w,q)$ [illegible]. We use simple elimination of the variables Δw and $\Delta\lambda$ from (9). The last equation gives

$$\Delta w = A^{-1}\left(-c(w,q) - \frac{\partial c}{\partial q}\Delta q\right).$$

Substitution of this for Δw in the third equation results in

$$\left(\frac{\partial h}{\partial q} - \frac{\partial h}{\partial w}A^{-1}\frac{\partial c}{\partial q}\right)\Delta q = -h + \frac{\partial h}{\partial w}A^{-1}c.$$

If we define

$$g_h := \left(\frac{\partial h}{\partial q} - \frac{\partial h}{\partial w}A^{-1}\frac{\partial c}{\partial q}\right)^\top = \begin{bmatrix} -A^{-1}\dfrac{\partial c}{\partial q} \\ I \end{bmatrix}^\top \begin{bmatrix} \nabla_w h \\ \nabla_q h \end{bmatrix}$$

as the reduced gradient of $h(w,q)$, one obtains

$$g_h^\top \Delta q = -h + \frac{\partial h}{\partial w}A^{-1}c. \tag{11}$$

From the first equation one gets

$$\Delta\lambda = A^{-\top}\left(-\nabla_w L - \left(\frac{\partial h}{\partial w}\right)^\top \Delta\mu\right).$$

Substitution in the second equation results in

$$B\Delta q + \left(\left(\frac{\partial h}{\partial q}\right)^\top - \left(\frac{\partial c}{\partial q}\right)^\top A^{-\top}\left(\frac{\partial h}{\partial w}\right)^\top\right)\Delta\mu = -\nabla_q L + \left(\frac{\partial c}{\partial q}\right)^\top A^{-\top}\nabla_w L.$$

This can be written in terms of the reduced gradient as

$$B\Delta q + g_h\Delta\mu = -\nabla_q L + \left(\frac{\partial c}{\partial q}\right)^\top A^{-\top}\nabla_w L. \tag{12}$$

Writing (11) and (12) in matrix vector notation results in the new system (10). ∎

We use this reduction strategy in the following to solve problem (8) using simultaneous pseudo-timestepping. The preconditioned pseudo-time embedded nonstationary system to be solved reads as

$$\begin{pmatrix} 0 & 0 & \left(\dfrac{\partial h}{\partial w}\right)^\top & A^\top \\ 0 & B & \left(\dfrac{\partial h}{\partial q}\right)^\top & \left(\dfrac{\partial c}{\partial q}\right)^\top \\ \dfrac{\partial h}{\partial w} & \dfrac{\partial h}{\partial q} & 0 & 0 \\ A & \dfrac{\partial c}{\partial q} & 0 & 0 \end{pmatrix} \begin{pmatrix} \dot w \\ \dot q \\ \dot\mu \\ \dot\lambda \end{pmatrix} = \begin{pmatrix} -\nabla_w L \\ -\nabla_q L \\ -h \\ -c \end{pmatrix}. \tag{13}$$

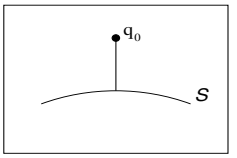


FIG. 1. *Projection.*

This problem is reduced to the QP problem (Lemma 3.1)

$$\begin{aligned} &\min \ \frac{1}{2}\dot{q}^\top B\dot{q} + g^\top \dot{q} \\ &\text{s.t.} \quad g_h^\top \dot{q} = -h(w,q) + \frac{\partial h}{\partial w} A^{-1} c, \end{aligned} \tag{14}$$

where $g = (\nabla_q L - (\frac{\partial c}{\partial q})^\top A^{-\top} \nabla_w L)$ is the reduced gradient of the objective function. For the construction of the reduced gradients g and g_h, one has to solve one adjoint problem (approximately) for g and one for g_h.

LEMMA 3.2. [illegible] (14) [illegible] (10) [illegible] $\dot{q}$ [illegible] $\dot{\mu}$ [illegible] (14)) [illegible]. Writing the Lagrangian of the system and a straightforward calculation of the first order necessary conditions will lead to the result. ∎

Therefore, this reduction can be interpreted as a projection of the design velocity from (7) toward the linearized lift constraint, thus resembling dynamic projection strategies onto invariants as in [36].

3.1. Back projection. Due to the nonlinearity of the problem there is some deviation from the additional state constraint. Therefore we use the following correction strategy in each optimization step to avoid this deviation. We minimize the distance between the point q_0 and the manifold $\mathcal{S}$ of constant lift $\mathcal{S} = \{q \,|\, h(w(q)) = l_0\}$ (Figure 1). That means we solve ideally the problem

$$\begin{aligned} &\min \ \tfrac{1}{2}||q - q_0||^2 \\ &\text{s.t.} \quad h(w(q)) - l_0 = 0. \end{aligned} \tag{15}$$

We use one step of a generalized Gauss–Newton method to solve this problem. Since the stiffness matrix of the flow equations is approximated by A when forming the reduced gradient g_h, we compute the step Δq from

$$\begin{aligned} &\min \ \frac{1}{2}||\Delta q||^2 \\ &\text{s.t.} \quad g_h^\top \Delta q = -h(w,q) + \frac{\partial h}{\partial w} A^{-1} c. \end{aligned} \tag{16}$$

The necessary optimality conditions will lead to the system of equations

$$\begin{pmatrix} I & g_h \\ g_h^\top & 0 \end{pmatrix} \begin{pmatrix} \Delta q \\ \nu \end{pmatrix} = \begin{pmatrix} 0 \\ -\left(h(w,q) - \dfrac{\partial h}{\partial w} A^{-1} c\right) \end{pmatrix},$$

where ν is the Lagrange multiplier for the system (16). Solving this system gives

$$\Delta q = \frac{-(h(w,q) - \frac{\partial h}{\partial w} A^{-1} c)}{g_h^\top g_h} g_h.$$

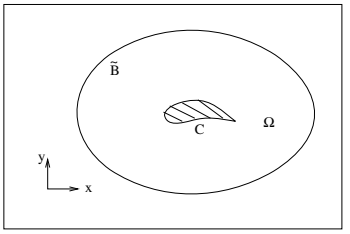


FIG. 2. *Physical domain of the problem.*

Hence, the corrected step is given by

$$q^{k+1} = q_0^{k+1} - \Delta q. \tag{17}$$

The overall algorithm reads as follows.

ALGORITHM 2. The simultaneous pseudo-timestepping for the preconditioned system.

(0) Set $k := 0$; start at some initial guess w_0, λ_0, and q_0.
(1) Compute λ^{k+1} marching one step in time for the adjoint equations.
(2) Compute sensitivities using state and adjoint solutions (using (28)).
(3) Determine some approximation B_k of the projected Hessian of the Lagrangian (for example, using (31)).
(4) Solve the quadratic subproblem (14) to get $\dot{q}$.
(5) March in time one step for the design equation as follows: $q_0^{k+1} = q^k + \Delta t \cdot \dot{q}$
(6) Use the correction step (17) for the new step.
(7) Compute w^{k+1} marching one step in time for the state equations.
(8) Set $k := k + 1$; go to (1) until convergence.

In the current implementation, $A^{-1}c$ in the right-hand side of the constraint in (14) is approximated by (scaled) $\dot{w}$ value from the previous iteration as it is updated only at step (7) of the above algorithm. The scaling factor depends on the accuracy of the state solution achieved in one time step as well as discretization of the parameter space. The size of the vector $\dot{w}$ is caused by the "orthogonal step" $A^{-1}c$ as well as by the tangential optimization step. Therefore it is only a crude measure for $A^{-1}c$. In our applications the scaling factor lies between 0.01 and 0.04. If we were to work with a sufficiently converged state solution, then c would be almost equal to 0. Since we do not work with a sufficiently converged solution, this approximation is necessary. Δt in step (5) of the algorithm is the minimum time step length of the forward solver. It is a "one-shot" method since we perform one time step for each design update.

4. Detailed equations of the aerodynamic shape optimization problem. In this section we explain briefly the state, costate, and design equations represented in equations (4a)–(4c) for the shape optimization problem.

State equations. Since we are interested in the steady flow, a proper approach for numerical modeling is to integrate the unsteady Euler equations in time until a steady state is reached. These equations in Cartesian coordinates (x, y) for two-dimensional flow can be written in integral form for the region Ω with boundaries $\partial\Omega$ $(= \tilde{B} \cup C)$ (see Figure 2) as

$$\frac{\partial}{\partial t} \int_\Omega w \, d\Omega + \int_{\partial\Omega} F \cdot \vec{n} \, ds = 0, \tag{18}$$

where $\vec{n}$ denotes the unit outward normal to $\partial\Omega$ and

$$w := \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{pmatrix}, \qquad F := [f, g], \qquad f := \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uH \end{pmatrix}, \quad \text{and} \quad g := \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ \rho vH \end{pmatrix},$$

where ρ, u, v, p, E, and H denote density, x- and y-components of velocity, pressure, total energy, and enthalpy, respectively. For a perfect gas the pressure and total enthalpy are given by

$$p = (\gamma - 1)\rho \left\{ E - \frac{1}{2}(u^2 + v^2) \right\}, \quad H = E + \frac{p}{\rho},$$

respectively, with γ denoting the ratio of specific heats. The boundary conditions used to solve these equations are the zero normal velocity on the solid wall, and the far field boundary is treated by considering the incoming and outgoing characteristics based on the one-dimensional Riemann invariants.

The cost function that we choose in the present optimization problem is drag reduction. Hence, the cost function reads as

$$I(w, q) := C_D = \frac{1}{C_{ref}} \int_C C_p \left(n_x \cos\alpha + n_y \sin\alpha \right) ds, \tag{19}$$

where C_{ref} denotes the chord length, α denotes the angle of attack, n_x and n_y denote the components of the unit outward normal, and the surface pressure coefficient is defined by

$$C_p := \frac{2(p - p_\infty)}{\gamma M_\infty^2 p_\infty}. \tag{20}$$

Here M denotes the Mach number, and the quantities with subscript $_\infty$ denote the values at free stream. The additional state constraint is constant lift; that is,

$$h(w, q) := C_L - C_{L0} = \frac{1}{C_{ref}} \int_C C_p \left(n_y \cos\alpha - n_x \sin\alpha \right) ds \; - \; C_{L0}, \tag{21}$$

where C_{L0} is a given constant value. Similarly, the additional state constraint of constant pitching moment is given by

$$h_1(w, q) := C_M - C_{M0} = \frac{1}{C_{ref}^2} \int_C C_p \left(n_y(x - x_m) - n_x(y - y_m) \right) ds \; - \; C_{M0}, \tag{22}$$

where (x_m, y_m) is the pitching moment's reference point and C_{M0} is a given constant value. The airfoil is decomposed into thickness and camber distributions. During the optimization only camberline is modified. This allows the thickness of the airfoil to remain constant (this geometric constraint is necessary; otherwise the optimization will result in a flat plate).

Costate equations. The costate or adjoint Euler equations are given by (see, for example, [11])

$$\frac{\partial}{\partial t} \int_\Omega \lambda \, d\Omega + \int_{\partial\Omega} \bar{F} \cdot \vec{n} \, ds = 0, \tag{23}$$

where the vector λ contains the components of the adjoint variable and $\bar{F}$ is the matrix of adjoint flux density, defined as

$$\lambda := \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_4 \end{pmatrix}, \qquad \bar{F} := \left[\left(\frac{\partial f}{\partial w} \right)^T \lambda \, , \left(\frac{\partial g}{\partial w} \right)^T \lambda \right].$$

The boundary conditions for the adjoint Euler equations on the solid body are of Neumann type, and for the above mentioned cost function they are given by

$$n_x \lambda_2 + n_y \lambda_3 = \text{RHS} \quad \text{on solid wall.} \tag{24}$$

For the above mentioned cost function the right-hand side (RHS) of (24) is

$$\text{RHS} = -\frac{2}{\gamma M_\infty^2 p_\infty C_{ref}} (n_x \cos\alpha + n_y \sin\alpha) \quad \text{on solid wall.} \tag{25}$$

For the lift constraint it is given by

$$\text{RHS} = -\frac{2}{\gamma M_\infty^2 p_\infty C_{ref}} (n_y \cos\alpha - n_x \sin\alpha) \quad \text{on solid wall,} \tag{26}$$

and for the pitching moment constraint it is given by

$$\text{RHS} = -\frac{2}{\gamma M_\infty^2 p_\infty C_{ref}^2} (n_y (x - x_m) - n_x (y - y_m)) \quad \text{on solid wall.} \tag{27}$$

The far field boundary conditions are based on incoming and outgoing characteristics, and free-stream conditions apply here as well. It is important to note that the adjoint Euler equations are linear in λ and the wall boundary conditions depend on the cost function.

Design equation. For the design equation (4c), we need an expression for the derivative of the Lagrangian with respect to the geometry of the airfoil. All the computations are carried out in a generalized coordinate system. Therefore, a transformation is used to transform the physical (x, y)-domain into the computational (ξ, η)-domain. In the computational domain, the components of the gradient $\frac{\partial L}{\partial q}$ can be determined by integrating the adjoint solutions multiplied by the metric sensitivities as follows:

$$\begin{aligned} \left(\frac{\partial L(q + \epsilon \tilde{q})}{\partial q} \right) \Big|_{\epsilon=0} &= -\int_C p(-\lambda_2 \tilde{q}_\xi^y + \lambda_3 \tilde{q}_\xi^x) ds \\ &\quad - \int_\Omega \left(\lambda_\xi^T \left(\tilde{q}_\eta^y f - \tilde{q}_\eta^x g \right) + \lambda_\eta^T \left(-\tilde{q}_\xi^y f + \tilde{q}_\xi^x g \right) \right) d\Omega + T1, \end{aligned} \tag{28}$$

where $\tilde{q}$ is the variation in the geometry of the airfoil and $\tilde{q}^x$, $\tilde{q}^y$ are its x- and y-components. For the sensitivities of the cost function the term $T1$ is given by

$$T1 = \frac{1}{C_{ref}} \int_C C_p \left((\tilde{q}^\perp)^x \cos\alpha + (\tilde{q}^\perp)^y \sin\alpha \right) ds, \tag{29}$$

and for the additional constraint of lift, it is given by

$$T1 = \frac{1}{C_{ref}} \int_C C_p \left((\tilde{q}^\perp)^y \cos\alpha - (\tilde{q}^\perp)^x \sin\alpha \right) ds, \tag{30}$$

where $\left((\tilde{q}^\perp)^x, (\tilde{q}^\perp)^y \right)$ are the components of the unit normal to $\tilde{q}$.

Discretization. The state and costate equations are discretized using a cell centered finite volume scheme. The semidiscrete equations are augmented with first and third order artificial dissipation and then solved using a 5-stage Runge–Kutta type timestepping scheme, as explained in detail in [22, 23].

Surface parameterization. The airfoil is decomposed into thickness and camber distributions. During the optimization only camber distributions are modified. This allows the thickness of the airfoil to remain constant (otherwise the drag reduction problem will result in a flat geometry). In this representation the y-coordinates of the surface are written in parametric form

$$y(\bar{r}) = \sum_{i=1}^{n} \tau_i \psi_i(\bar{r}),$$

where $\bar{r}$ is the coordinate along the airfoil chord, ψ_i are the base functions, and τ_i are the design parameters (here $q = (\tau_1, \tau_2, \ldots, \tau_n)^\top$). In the current study Hicks–Henne functions [17] are selected as base functions which are given by

$$\psi_i(r) = \sin\left(\pi r^{\frac{\ln(0.5)}{\ln(a)}}\right)^b,$$

where a and b control the center and thickness of the perturbations and r is the normalized coordinate along the chord.

Gradient computation. As an efficient method of calculating the gradient $(\delta I)_{m=1,\ldots,n} := \nabla_q I$, we use, as in [11], the so-called grid moving technique based on Reuther's approach (see [21]) for evaluating the integrals (28), which are dependent on the adjoint field vector λ and the metric sensitivities generated by the perturbation of the geometry (by the n design variables). The idea is to allow the geometry perturbation in the whole flow field (the whole grid) while keeping the far field boundary fixed. For this, one introduces a distance function $R(\xi, \eta) = 1 - \frac{\eta}{\eta_B(\xi)}$, where $\eta_B(\xi)$ denotes the values of η at the far field corresponding to the points $(\xi, 0)$ at the wall. Then it holds that

$$R(\xi, 0) = 1 \quad \text{at the wall} \quad \text{and} \quad R(\xi, \eta_B(\xi)) = 0 \quad \text{at the far field.}$$

The grid for the perturbed (new) geometry is defined by

$$x_{new} - x_{old} := R \cdot (x_{new_s} - x_{old_s}),$$

$$y_{new} - y_{old} := R \cdot (y_{new_s} - y_{old_s}),$$

where the index s refers to the values at the surface and x_{new}, y_{new} correspond to x_{new_s}, y_{new_s}, respectively. Thus, the perturbations are

$$\delta x = R \delta x_s, \qquad \delta y = R \delta y_s,$$

where $\tilde{q}^x = \delta x_s$, $\tilde{q}^y = \delta y_s$ in (28), and the metric sensitivities are

$$\delta x_\xi = R \cdot (\delta x_s)_\xi + R_\xi \delta x_s, \quad \delta y_\xi = R \cdot (\delta y_s)_\xi + R_\xi \delta y_s,$$

$$\delta x_\eta = R \cdot (\delta x_s)_\eta + R_\eta \delta x_s, \quad \delta y_\eta = R \cdot (\delta y_s)_\eta + R_\eta \delta y_s.$$

The second term of each of these equalities is very small and, therefore, could be omitted from the above expressions. The effect of this simplification has been studied and justified in [11]. In that case the metric sensitivities can be expressed as

$$\delta x_\xi \approx R \cdot (\delta x_s)_\xi, \quad \delta y_\xi \approx R \cdot (\delta y_s)_\xi,$$

$$\delta x_\eta \approx R \cdot (\delta x_s)_\eta, \quad \delta y_\eta \approx R \cdot (\delta y_s)_\eta,$$

and one obtains the components of the gradient (for the cost function (19)) as integrals along the geometry C:

$$\begin{aligned}\delta I = &- \int_C \left((\delta y_s)_\eta \int_\eta \frac{\partial \lambda^T}{\partial \xi} (R(\eta) \cdot f) d\eta \right) d\xi + \int_C \left((\delta x_s)_\eta \int_\eta \frac{\partial \lambda^T}{\partial \xi} (R(\eta) \cdot g) d\eta \right) d\xi \\ &+ \int_C \left((\delta y_s)_\xi \int_\eta \frac{\partial \lambda^T}{\partial \eta} (R(\eta) \cdot f) d\eta \right) d\xi - \int_C \left((\delta x_s)_\xi \int_\eta \frac{\partial \lambda^T}{\partial \eta} (R(\eta) \cdot g) d\eta \right) d\xi \\ &- \int_C p \left(-\lambda_2 (\delta y_s)_\xi + \lambda_3 (\delta x_s)_\xi \right) d\xi + \frac{1}{S_{ref}} \int_C C_p \left((\delta y_s)_\xi \cos\alpha - (\delta x_s)_\xi \sin\alpha \right) d\xi.\end{aligned}$$

The integrals in η have to be evaluated once. Then, for each design variable only the integration in ξ along C remains to be evaluated.

Grid-perturbation strategy. As the shape of the airfoil changes during the optimization process, the location of the grid nodes has to be adjusted. This can be done by generating a new grid after each design iteration or by using a grid-perturbation strategy (as explained in [16]) after each design iteration.

The discretization, surface parameterization, and grid-perturbation strategy are performed in a standard way (cf. [25, 26, 11]).

5. Numerical results and discussion. The optimization method is applied to test cases of the RAE2822 airfoil for drag reduction with constant lift and drag reduction with constant pitching moment. The physical domain is discretized using an algebraically generated (193×33) C-grid. The camberline representation of the airfoil is parameterized by 21 Hicks–Henne parameters. The complete optimization cycle is performed under the optimization platform SynapsPointerPro [9]. We start the optimization iteration with the solution obtained after 500 time steps of the state equations (w_0) and of the costate equations (λ_0). The FLOWer code [25, 26] of the German Aerospace Center (DLR) is used for solving the forward and adjoint equations. These codes are very robust and have been tested in many applications including flow computations for the company AIRBUS-Germany. Therefore, we made no modifications to this code by taking the block matrices A and $A^\top$ corresponding to the state and costate equations in the preconditioner. They are just identity matrices in the current implementation.

The design equation is integrated in time using an explicit Euler scheme, and the state and costate equations are integrated in time using a 5-stage Runge–Kutta scheme. Therefore the time steps used for the three sets of equations are not the same. In the current implementation of FLOWer the time steps are not the same even in each discretization cell as they are determined independently according to the local stability. However, this has no effect on the final solution at steady state.

One of the main issues in using this kind of preconditioned pseudo-timestepping is the approximation of the reduced Hessian. A better approximation leads to faster

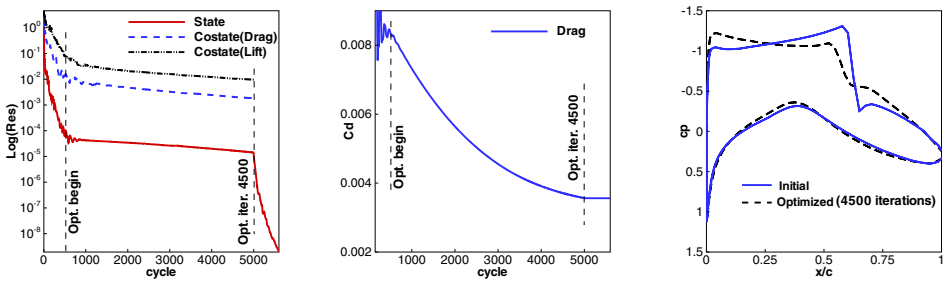


FIG. 3. *Convergence history of the optimization iterations and the comparison of surface pressure distributions.*

convergence of the optimization algorithm. We first take an approximation of the reduced Hessian as

$$B_k = \beta \delta_{ij},$$

where β ($>> 1$) is a constant which is chosen to scale the design updates and δ_{ij} is the Kronecker symbol representing the identity matrix. In the current computation the value of β is 15. The convergence history and the surface pressure distributions of 4500 optimization iterations are presented in Figure 3. As we see from the optimized surface pressure distribution, there is still a weak shock present there, which means that the solution has not yet reached the optimum after 4500 optimization iterations.

Next, we consider the reduced Hessian approximation based on the current gradient and parameter update information. We define $s_k := (q_{k+1} - q_k)$ and $z_k := (\nabla I_{k+1} - \nabla I_k)$, where k represents the iteration number. Then the reduced Hessian approximation is determined based on the sign of the product $(z_k^T s_k)$. If the sign is positive, the reduced Hessian is approximated by

$$B_k = \bar{\beta} \frac{z_k^T s_k}{z_k^T z_k} \delta_{ij}, \tag{31}$$

where $\bar{\beta}$ is a constant. Otherwise, it is approximated by $\beta \delta_{ij}$, where β is another constant. Additionally, we impose upper and lower limits on the factor so that

$$\beta_{min} < \bar{\beta} \frac{z_k^T s_k}{z_k^T z_k} < \beta_{max}.$$

This prevents the optimizer from taking too small or too large steps. The constants β_{min} and β_{max} can be chosen, e.g., depending on the accuracy achieved in one time step by the forward and adjoint solver. In the current computations the values of the constants $\bar{\beta}$, β_{max}, and β_{min} are 6, 15, and 2.5.

The optimization iteration is stopped when $||\Delta q||_\infty < 0.0005$. After the convergence of the optimization iteration, another 600 time steps are performed for the state equation on the optimized geometry to reduce its residual further so that the force coefficients and surface pressure distribution can be compared with those obtained by other methods.

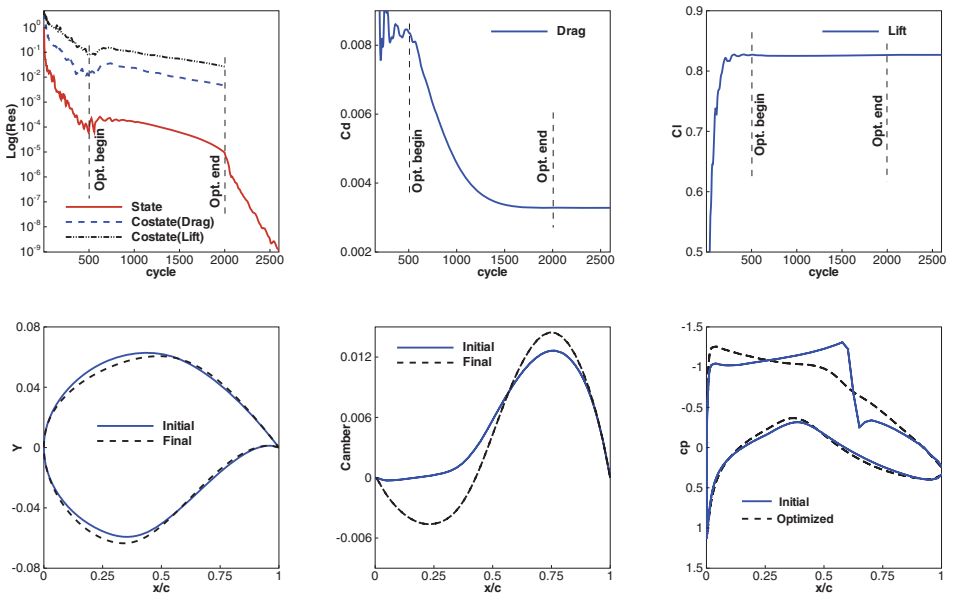


FIG. 4. *Convergence histories* (*top*) *and comparison of initial and final airfoils, camberlines, and surface pressure distributions* (*bottom*) *of the optimization of Case* 1.

TABLE 1

Comparison of number of iterations and force coefficients for baseline and optimized airfoils for different numbers of design parameters (*on* 193×33 *grid*).

Geometry	Iter	CD	ΔC_D	CL	ΔC_L
Baseline		0.849651E-02		0.826600E+00	
Opt.(Case 1, p=21)	1500	0.328198E-02	61.37%	0.826832E+00	-0.028%
Opt.(Case 2, p=40)	750	0.328765E-02	61.30%	0.826671E+00	-0.009%

[illegible] 1 [illegible] 21 [illegible]

The optimization problem requires 1500 iterations to converge. Figure 4 (top) presents the optimization convergence histories of this case. Figure 4 (bottom) presents the comparison of initial and final airfoils, camberlines, and surface pressure distributions. Since the major amount of drag in the transonic regime is caused by the shock, the optimization for drag reduction results in a shockless airfoil. A comparison of the force coefficients for baseline and optimized geometries is presented in Table 1. The drag reduction is 61.37%, and the lift is well maintained with an increase of only 0.028%.

Each forward run and each adjoint run requires approximately 1500 iterations in time to reach a steady state. The total number of time steps required in this case is $(500+1500+600)$ for a state solver, and 2 times $(500+1500)$ for adjoint solvers. This together is approximately 4.4 times that of the forward simulation run. The effort to solve the design equation is almost negligible compared to state or costate solutions. Also, in every optimization iteration the approximate gradients are to be computed twice (once for the drag and once for the lift). Since this is only a matrix-vector product with the current approximations of the adjoints, the time required for this is also negligible compared to that of the state or costate solutions. Additionally, the

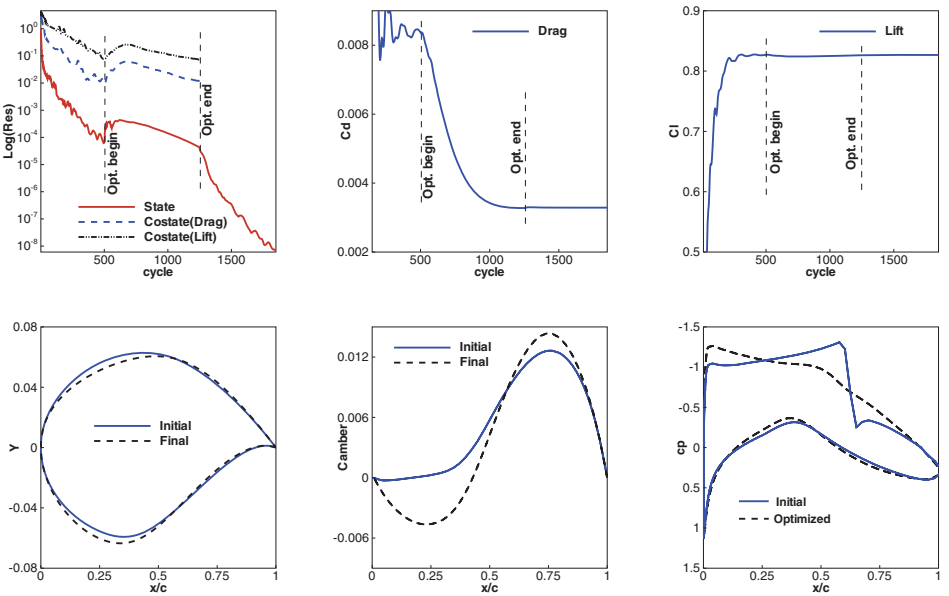


FIG. 5. *Convergence histories* (*top*) *and comparison of initial and final airfoils, camberlines, and surface pressure distributions* (*bottom*) *of the optimization of Case* 2.

simultaneous pseudo-time method needs a new grid, obtained by grid perturbation as discussed earlier, after each optimization iteration. Additional time is required to write the output after every iteration and read the same before each iteration, as the iterations start with solution values from the previous iteration. However, the total time required for this overhead is less than one complete forward run. If we add on everything, the total effort is less than 7 times that of forward simulation runs.

2 40

In this case we use finer parameterization of the design space with 40 design parameters. The optimization problem requires 750 iterations to converge. The faster convergence of the method can be explained from the physics of the problem. The optimization problem deals with drag reduction with additional state constraint. The major drag in the transonic regime is caused by the shock jump. Therefore, the optimizer tries to change the design parameters in such a way that the drag disappears. In a finer parameterization a small change in parameters has more effect on shock than in a coarser parameterization. To achieve the same effect in coarser parameterization one has to have large changes in design parameters, which causes constraint violation. That is why in this particular problem class finer parameterization leads to faster convergence.

Since we use adjoint gradients, the gradient computation is independent of the number of design variables. Therefore, it is advantageous to use finer design space. Figure 5 (top) presents the optimization convergence histories of this case. Figure 5 (bottom) presents the comparison of initial and final airfoils, camberlines, and surface pressure distributions. A comparison of the force coefficients for baseline and optimized geometries is presented in Table 1. The drag reduction is 61.30%, and the lift is increased by only 0.009%. The optimized force coefficients are almost the same

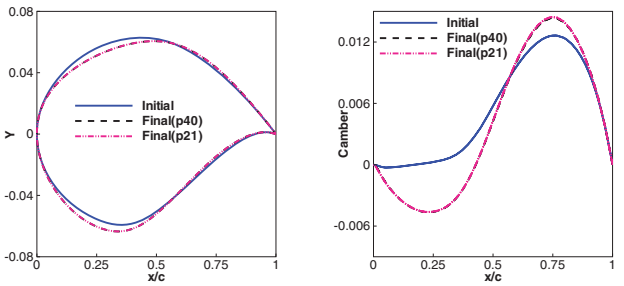


FIG. 6. *Comparison of initial and final airfoils and camberlines of Case* 1 *and Case* 2 *computations.*

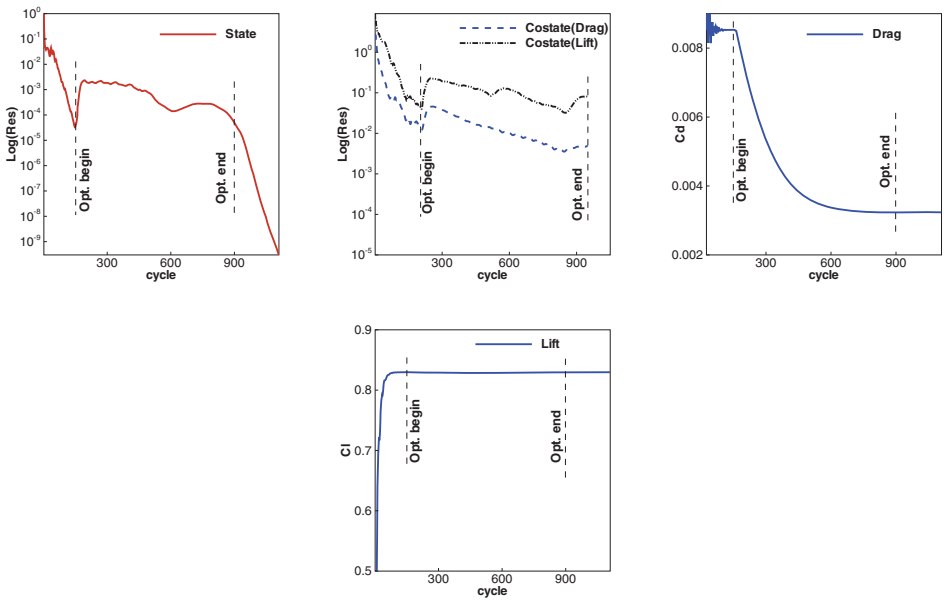


FIG. 7. *Convergence histories of the optimization iteration of Case* 3.

in both cases. Figure 6 presents the comparison of airfoils and camberlines obtained in Cases 1 and 2. Again, there is no noticeable difference. However, the number of optimization iterations is half that of Case 1, and the total effort in this case is less than 5 times that of forward simulation runs.

3 40 321×57

In this case the computational domain is discretized by a finer grid of size 321×57. The forward and adjoint solutions are computed using the multigrid solvers in FLOWer. The optimization is started with initial state solution (w_0) obtained after 150 time steps and costate solution (λ_0) obtained after 200 time steps. The optimization requires 750 iterations to converge. After the convergence of the optimization, another 200 time steps are carried out for the state solution to achieve sufficiently accurate force coefficients. Figure 7 presents the optimization convergence histories

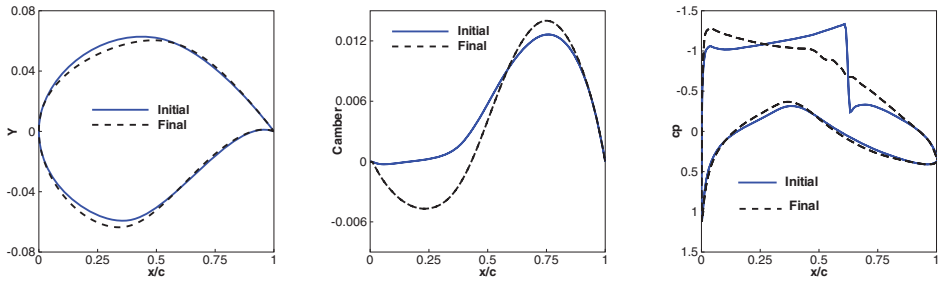


FIG. 8. *Comparison of initial and final airfoils, camberlines, and surface pressure distributions of Case* 3 *computations.*

TABLE 2
Comparison of force coefficients for baseline and optimized airfoils in Case 3 *computation* (*on* 321×57 *grid*).

Geometry	Iter	CD	ΔC_D	CL	ΔC_L
Baseline		0.852910E-02		0.829500E+00	
Opt.(Case 3, p=40)	750	0.323553E-02	62.06%	0.829657E+00	-0.02%

of this case. Figure 8 presents the comparison of initial and final airfoils, camberlines, and surface pressure distributions. A comparison of the force coefficients for baseline and optimized geometries is presented in Table 2. The drag reduction is 62.06% which is a bit more than the last two cases, and the lift is again well maintained by an increase of only 0.02%. In this case the steady-state forward solution is reached by 450 multigrid iterations and thus, if we add up everything, the total effort of the optimization problem is less than 8 forward simulation runs.

4 21

Computations for this case are carried out on a 193×33 grid as in Case 1 (and Case 2) with a different state constraint, namely, pitching moment. The design space is parameterized with 21 design parameters. The optimization problem requires 1400 iterations to converge in this case. Figure 9 (top) presents the optimization convergence histories of this case. Figure 9 (bottom) presents the comparison of initial and final airfoils, camberlines, and surface pressure distributions. Since the major amount of drag in this case also is caused by the shock, the optimization for drag reduction results in a shockless airfoil. A comparison of the force coefficients for baseline and optimized geometries is presented in Table 3. The drag reduction is 60.50% and the pitching moment is well maintained by a decrease of only 0.02%. In this case the total computational effort is approximately 6 times that of forward simulation runs.

5 40

Computations for this case are carried out as in the last case but with 40 design parameters. This requires 625 iterations for the convergence of the optimization problem. In this case also the convergence is faster than that of Case 3 due to finer parameterization of the design space. Figure 10 (top) presents the optimization convergence histories of this case. Figure 10 (bottom) presents the comparison of initial and final airfoils, camberlines, and surface pressure distributions. A comparison of the force coefficients for baseline and optimized geometries is presented in Table 3.

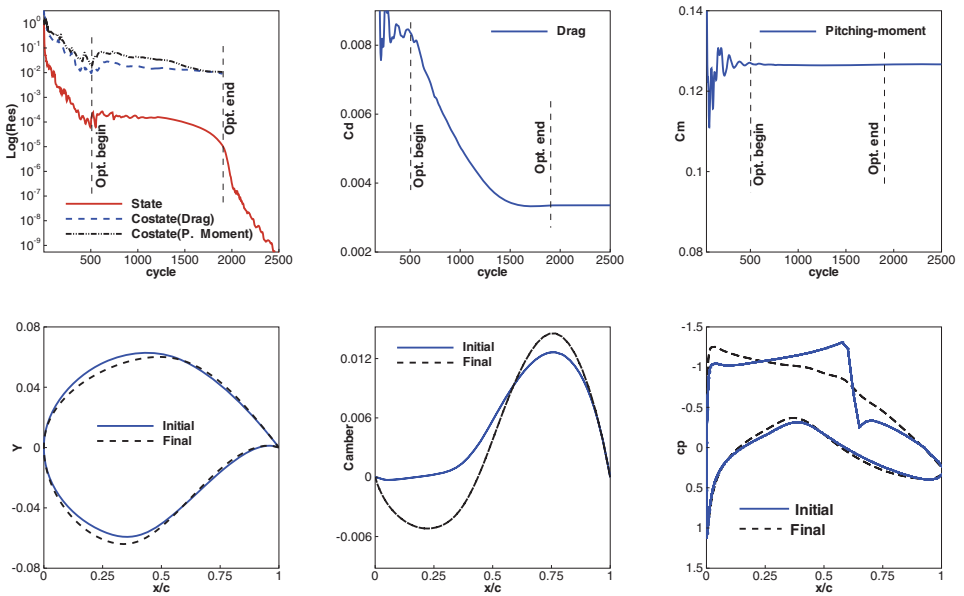


FIG. 9. *Convergence histories (top) and comparison of initial and final airfoils, camberlines, and surface pressure distributions (bottom) of the optimization of Case* 4.

TABLE 3

Comparison of number of iterations and force coefficients for baseline and optimized airfoils for different numbers of design parameters (on 193×33 *grid).*

Geometry	Iter	CD	ΔC_D	CM	ΔC_M
Baseline		0.849651E-02		0.126700E+00	
Opt.(Case 4, p=21)	1400	0.335566E-02	60.50%	0.126669E+00	0.02%
Opt.(Case 5, p=40)	625	0.340304E-02	59.95%	0.126731E+00	-0.02%

The drag reduction is 59.95%, and the pitching moment is well maintained with an increase of only 0.02%. The final force coefficients are almost the same as in Case 4. Figure 11 presents the comparison of initial and final airfoils and camberlines obtained in Cases 4 and 5. There is also no noticeable difference in the optimized airfoils and camberlines obtained in these two cases. Again, as in Case 2, the total computational effort in Case 5 is approximately 4 times that of forward simulation runs.

In the convergence histories of all the figures, one observes a linear convergence with respect to the objective; starting from an almost feasible initial guess (i.e., after 500 state iterations and 500 costate iterations independently (150 and 200 iterations, respectively, for Case 3)), each iteration step in the primal, adjoint, and design space is so small that the process truly reflects a continuous behavior. Each integration step of the preconditioned pseudo-time formulation is contracting enough so that no additional globalization strategy is necessary. The preconditioning in the design space gives short enough integration steps that the iterates "surf" along an almost feasible manifold toward optimality.

We have not made an explicit comparison of our results with those of other optimization methods. However, it is possible to obtain an impression of this comparison if one imagines what optimization progress could be achieved in 3 optimization iter-

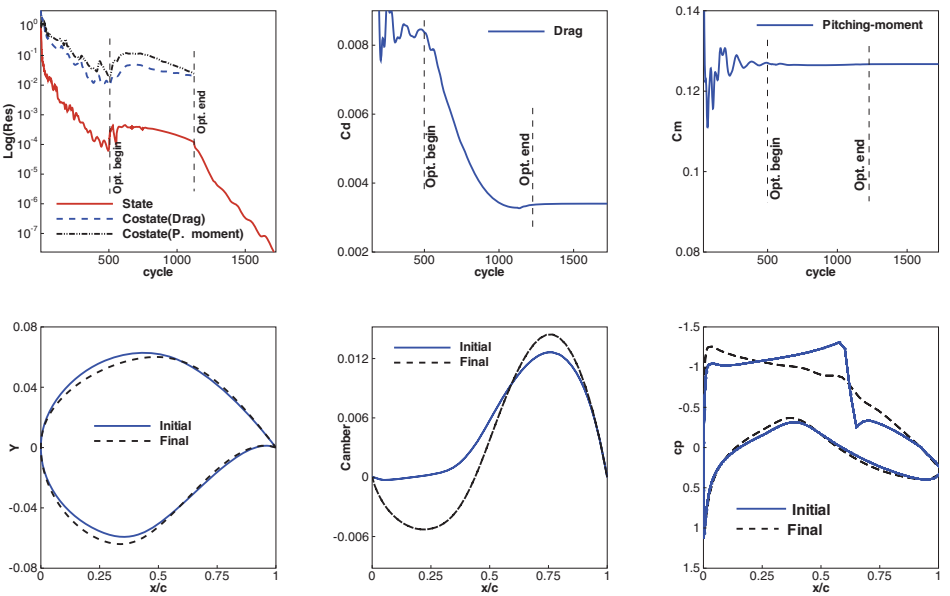


FIG. 10. *Convergence histories* (*top*) *and comparison of initial and final airfoils, camberlines, and surface pressure distributions* (*bottom*) *of the optimization of Case* 5.

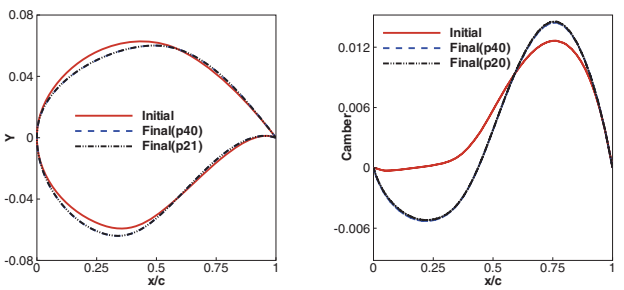


FIG. 11. *Comparison of initial and final airfoils and camberlines of Case* 4 *and Case* 5 *computations.*

ations (that is, 3 forward and 2×3 adjoint simulation runs) of any gradient based methods, since this takes more (9 forward simulation runs) computational effort than our one-shot approach. The CPU time required for the computations on a 193×33 grid is 8–16 minutes and on a 321×57 grid about 30 minutes on a Linux machine with an AMD Opteron processor, cpu 2405 MHz, and 16 MB RAM.

6. Conclusions. A new method has been proposed for aerodynamic shape optimization with state constraints which is based on simultaneous pseudo-timestepping. The preconditioned pseudo-stationary state, costate, and design equations are integrated simultaneously in time until a steady state is reached. The preconditioner used in this study is motivated by a continuous reinterpretation of RSQP methods. Reduced Hessian approximation based on quasi-Newton updates leads to faster convergence than that obtained by using a constant diagonal matrix. The overall cost of

computation is approximately 4–8 times that of forward simulation runs. Application to three-dimensional problems is our next goal.

Acknowledgments. We would like to thank N. Gauger and J. Brezillon for discussions on the FLOWer code and the gradient based optimization methods for FLOWer that currently exist. We also would like to thank the anonymous referees for their comments and suggestions which have substantially improved the exposition.

REFERENCES

[1] W. K. Anderson and V. Venkatakrishnan, *Aerodynamic design optimization on unstructured grids with a continuous adjoint formulation*, Comput. Fluids, 28 (1999), pp. 443–480.

[2] A. Battermann and M. Heinkenschloss, *Preconditioners for Karush-Kuhn-Tucker systems arising in the optimal control of distributed systems*, in Control and Estimation of Distributed Parameter Systems (Vorau, 1996), W. Desch, F. Kappel, and K. Kunisch, eds., Birkhäuser-Verlag, Basel, 1998, pp. 15–32.

[3] A. Battermann and E. W. Sachs, *Block preconditioners for KKT systems in PDE-governed optimal control problems*, in Fast Solution of Discretized Optimization Problems, K. H. Hoffmann, R. H. W. Hoppe, and V. Schulz, eds., Birkhäuser-Verlag, Basel, 2001, pp. 1–18.

[4] G. Biros and O. Ghattas, *Parallel Lagrange–Newton–Krylov–Schur methods for PDE-constrained optimization. Part* I: *The Krylov–Schur solver*, SIAM J. Sci. Comput., 27 (2005), pp. 687–713.

[5] J. Brezillon, *Application of the adjoint technique with the optimization framework Synaps Pointer Pro*, in MEGAFLOW—Numerical Flow Simulation for Aircraft Design, Notes Numer. Fluid Mech. Multidiscip. Des. 89, N. Kroll and J. Fassbender, eds., Springer-Verlag, Berlin, 2002, pp. 195–209.

[6] J. Brezillon and N. R. Gauger, *2D and 3D aerodynamic shape optimization using the adjoint approach*, Aerosp. Sci. Technol., 8 (2004), pp. 715–727.

[7] T. S. Coffey, C. T. Kelley, and D. E. Keyes, *Pseudotransient continuation and differential-algebraic equations*, SIAM J. Sci. Comput., 25 (2003), pp. 553–569.

[8] R. Fletcher, *Practical Methods of Optimization*, John Wiley & Sons, New York, 1987.

[9] O. Fromman, *SynapsPointerPro* v2.50, Synaps Ingenieure Gesellschaft mbH, Bremen, Germany, 2002.

[10] N. R. Gauger, *Aerodynamic shape optimization using the adjoint Euler equations*, in Proceedings of the GAMM Workshop on Discrete Modeling and Discrete Algorithms in Continuum Mechanics, Logos Verlag, Berlin, 2001, pp. 87–96.

[11] N. R. Gauger, *Das Adjungiertenverfahren in der aerodynamischen Formoptimierung*, DLR-Report DLR-FB-2003-05, DLR, Braunschweig, Germany, 2003.

[12] N. R. Gauger and J. Brezillon, *Aerodynamic shape optimization using adjoint method*, J. Aero. Soc. of India, 54 (2002), pp. 247–254.

[13] P. E. Gill, W. Murray, and M. H. Wright, *Practical Optimization,* Academic Press, London, 1981.

[14] S. B. Hazra, *An efficient method for aerodynamic shape optimization*, in Proceedings of the 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Albany, NY, 2004, AIAA Paper 2004-4628.

[15] S. B. Hazra and V. Schulz, *Simultaneous pseudo-timestepping for PDE-model based optimization problems*, BIT, 44 (2004), pp. 457–472.

[16] S. B. Hazra, V. Schulz, J. Brezillon, and N. R. Gauger, *Aerodynamic shape optimization using simultaneous pseudo-timestepping*, J. Comput. Phys., 204 (2005), pp. 46–64.

[17] R. M. Hicks and P. A. Henne, *Wing design by numerical optimization*, J. Aircraft, 15 (1978), pp. 407–412.

[18] A. Jameson, *Aerodynamic design via control theory*, J. Sci. Comput., 3 (1988), pp. 23–260.

[19] A. Jameson, *Automatic design of transonic airfoils to reduce the shock induced pressure drag*, in Proceedings of the 31st Israel Annual Conference on Aviation and Aeronautics, 1990, pp. 5–17.

[20] A. Jameson, *Optimum aerodynamic design using CFD and control theory*, in Proceedings of the 12th AIAA Computational Fluid Dynamics Conference, San Diego, 1995, AIAA Paper 95-1729.

[21] A. Jameson and J. Reuther, *Control theory based airfoil design using the Euler equations*, in Proceedings of the AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Panama City Beach, FL, 1994, AIAA Paper 94-4272.

[22] A. Jameson, W. Schmidt, and E. Turkel, *Numerical solution of the Euler equations by finite volume methods using Runge-Kutta time-stepping schemes*, in Proceedings of the 14th AIAA Fluid and Plasma Dynamic Conference, Palo Alto, CA, 1981, AIAA Paper 81-1259.

[23] N. Kroll and R. K. Jain, *Solution of Two-Dimensional Euler Equations—Experience with a Finite Volume Code*, Technical report DFVLR-IB-129-84/19, DLR, Braunschweig, Germany, 1984.

[24] C. T. Kelley and D. E. Keyes, *Convergence analysis of pseudo-transient continuation*, SIAM J. Numer. Anal., 35 (1998), pp. 508–523.

[25] N. Kroll, C. C. Rossow, K. Becker, and F. Thiele, *The MEGAFLOW—A Numerical Flow Simulation System*, presented at the 21st ICAS Symposium, Melbourne, Australia, 1998, Paper 98-2.7.4.

[26] N. Kroll, C. C. Rossow, K. Becker, and F. Thiele, *The MEGAFLOW project*, Aerosp. Sci. Technol., 4 (2000), pp. 223–237.

[27] J.-L. Lions, *Optimal Control of Systems Governed by Partial Differential Equations*, Springer-Verlag, New York, 1971.

[28] O. Pironneau, *On optimum design in fluid mechanics*, J. Fluid Mech., 64 (1974), pp. 97–110.

[29] O. Pironneau, *On optimum profiles in Stokes flow*, J. Fluid Mech., 59 (1973), pp. 117–128.

[30] O. Pironneau, *Optimal Shape Design for Elliptic Systems*, Springer-Verlag, New York, 1982.

[31] A. Iollo, G. Kuruvila, and S. Ta'asan, *Pseudo-Time Method for Optimal Shape Design Using the Euler Equations*, Technical report 95-59, ICASE, NASA Langley Research Center, Hampton, VA, 1995.

[32] J. Reuther and A. Jameson, *Aerodynamic shape optimization of wing and wing-body configurations using control theory*, in Proceedings of the 33rd AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, 1995, AIAA Paper 95-0123.

[33] J. Reuther, A. Jameson, J. Farmer, L. Martinelli, and D. Saunders, *Aerodynamic shape optimization of complex aircraft configurations via an adjoint formulation*, in Proceedings of the 34rd AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, 1996, AIAA Paper 96-0094.

[34] V. H. Schulz, *Reduced SQP Methods for Large-Scale Optimal Control Problems in DAE with Application to Path Planning Problems for Satellite Mounted Robots*, Ph.D. thesis, Universität Heidelberg, Heidelberg, Germany, 1996.

[35] V. H. Schulz, *Solving discretized optimization problems by partially reduced SQP methods*, Comput. Vis. Sci., 1 (1998), pp. 83–96.

[36] V. H. Schulz, H. G. Bock, and M. C. Steinbach, *Exploiting invariants in the numerical solution of multipoint boundary value problems for DAEs*, SIAM J. Sci. Comput., 19 (1998), pp. 440–467.

SIAM J. SCI. COMPUT.
Vol. 28, No. 3, pp. 1100–1119
© 2006 Society for Industrial and Applied Mathematics

STRUCTURED TOTAL LEAST SQUARES FOR COLOR IMAGE RESTORATION*

HAOYING FU†, MICHAEL K. NG‡, AND JESSE L. BARLOW†

Abstract. The problem of 3×3 color mixing image restoration is considered. The blurring matrices, as well as the observed image, are contaminated by noise; therefore the total least squares (TLS) method is employed to restore the original image. Since the blurring matrices are also structured, we apply the structured total least squares (STLS) method [J. B. Rosen, H. Park, and J. Glick, *SIAM J. Matrix Anal. Appl.*, 17 (1996), pp. 110–126]. The blurring matrices are generally ill conditioned; thus Tikhonov's regularization is used to stabilize the solution. Since Neumann boundary conditions are used in the restoration process, the discrete cosine transform (DCT) based preconditioner is effective for the linear systems encountered in our STLS algorithm.

Key words. structured total least squares, color image, image restoration, Toeplitz-like matrices, preconditioners

AMS subject classifications. 65F10, 65F15

DOI. 10.1137/040605436

1. Introduction.

The color image restoration problem has attracted much research effort recently [2, 8, 9, 12, 15, 17, 22]. A color image may be regarded as a set of three images in the three color components, red, green, and blue. A more general term, multichannel images, is often used to denote color images where each color component is considered as a channel.

In most applications [2], the observed image is distorted by blur and additive noise. Let $f_r(x,y)$, $f_g(x,y)$, and $f_b(x,y)$ be the original red, green, and blue scenes. Let $u_r(x,y)$, $u_g(x,y)$, and $u_b(x,y)$ be the observed red, green, and blue scenes. Let $h_{ij}(x,y)$ (for $i,j \in \{r,g,b\}$) be the within-channel and cross-channel point spread functions, normalized such that each function integrates to 1. Let $n_r(x,y)$, $n_g(x,y)$, and $n_b(x,y)$ be the noise present in the observed image. The image formation process can be modeled as follows [8, 9]:

$$\begin{aligned} u_r &= w_{rr}h_{rr} \star f_r + w_{gr}h_{gr} \star f_g + w_{br}h_{br} \star f_b + n_r, \\ u_g &= w_{rg}h_{rg} \star f_r + w_{gg}h_{gg} \star f_g + w_{bg}h_{bg} \star f_b + n_g, \\ u_b &= w_{rb}h_{rb} \star f_r + w_{gb}h_{gb} \star f_g + w_{bb}h_{bb} \star f_b + n_b. \end{aligned} \tag{1.1}$$

Here w_{ij} (for $i,j \in \{r,g,b\}$) are nonnegative weights that measure the contribution of the original channels to each observed channel, and $\star$ denotes two-dimensional (2D) convolution. We note that at least one of w_{ri}, w_{gi}, and w_{bi} ($i \in \{r,g,b\}$) is nonzero.

Let $\mathbf{f}_i$, $\mathbf{u}_i$, and $\mathbf{n}_i$ (for $i \in \{r,g,b\}$) be the row by row ordering of the discretized original scene, observed scene, and additive noise, respectively. Let H_{ij} (for $i,j \in \{r,g,b\}$) be the blurring matrices of appropriate size built according to the discretized

*Received by the editors March 21, 2004; accepted for publication (in revised form) March 3, 2006; published electronically July 7, 2006.
http://www.siam.org/journals/sisc/28-3/60543.html

†Department of Computer Science and Engineering, Pennsylvania State University, University Park, PA 16802 (hfu@cse.psu.edu, barlow@cse.psu.edu). Part of this work was done while the first author was visiting the University of Hong Kong Mathematics Department.

‡Department of Mathematics, Hong Kong Baptist University, Kowloon Tong, Hong Kong (mng@math.hkbu.edu.hk). The research of this author was supported in part by Hong Kong Research Grants Council grants 7046/03P, 7035/04P, and 7035/05P and HKBU FRGs.

point spread functions $\mathbf{h}_{ij}$. Then the discretized image formation process can be put into a matrix-vector form:

$$\begin{aligned} \mathbf{u}_r &= w_{rr}H_{rr}\mathbf{f}_r + w_{gr}H_{gr}\mathbf{f}_g + w_{br}H_{br}\mathbf{f}_b + \mathbf{n}_r, \\ \mathbf{u}_g &= w_{rg}H_{rg}\mathbf{f}_r + w_{gg}H_{gg}\mathbf{f}_g + w_{bg}H_{bg}\mathbf{f}_b + \mathbf{n}_g, \\ \mathbf{u}_b &= w_{rb}H_{rb}\mathbf{f}_r + w_{gb}H_{gb}\mathbf{f}_g + w_{bb}H_{bb}\mathbf{f}_b + \mathbf{n}_b. \end{aligned} \tag{1.2}$$

Let

$$\mathbf{f} = \begin{bmatrix} \mathbf{f}_r \\ \mathbf{f}_g \\ \mathbf{f}_b \end{bmatrix}, \quad \mathbf{u} = \begin{bmatrix} \mathbf{u}_r \\ \mathbf{u}_g \\ \mathbf{u}_b \end{bmatrix}, \quad \text{and} \quad \mathbf{n} = \begin{bmatrix} \mathbf{n}_r \\ \mathbf{n}_g \\ \mathbf{n}_b \end{bmatrix},$$

let

$$\mathbf{w} = [w_{rr} \quad w_{gr} \quad w_{br} \quad w_{rg} \quad w_{gg} \quad w_{bg} \quad w_{rb} \quad w_{gb} \quad w_{bb}]^T,$$

and define

$$H(\mathbf{w}) = \begin{bmatrix} w_{rr}H_{rr} & w_{gr}H_{gr} & w_{br}H_{br} \\ w_{rg}H_{rg} & w_{gg}H_{gg} & w_{bg}H_{bg} \\ w_{rb}H_{rb} & w_{gb}H_{gb} & w_{bb}H_{bb} \end{bmatrix}.$$

Equation (1.2) can be written as

$$\mathbf{u} = H(\mathbf{w})\mathbf{f} + \mathbf{n}. \tag{1.3}$$

Very often people use the observed image to recover the original image of the same domain. The observed image is not completely determined by the original image in the region of interest (ROI) because of the convolution. To determine the image, we make assumptions about the pixel values of the original image outside the ROI. The zero boundary condition assumes that the scene outside the ROI is totally dark. When the real scene outside the ROI does not satisfy this assumption well, ringing effects will appear at the boundaries of the restored image. Zero boundary conditions cause the blurring matrices H_{ij} to be block Toeplitz with Toeplitz blocks (BTTB). Neumann boundary conditions assume that the scene outside the ROI is a reflection of the scene inside. This reduces the ringing effects when the image is close to stationary at the boundaries. Using Neumann boundary conditions causes the blurring matrices H_{ij} to be block Toeplitz plus Hankel with Toeplitz plus Hankel blocks (BTHTHB), which in general gives better restoration results than those using zero boundary conditions [16].

Image restoration problems tend to be very ill conditioned. Directly solving (1.2) will yield a solution that is extremely sensitive to noise; therefore regularization methods are needed to stabilize the solution.

There is considerable work in the literature on least squares restoration of color images. Based on the assumption that the within-channel and cross-channel point spread functions are separable, Hunt and Kubler [12] derived a minimum mean squares error (MSE) multichannel restoration filter. Filters that utilize the cross-channel correlation without using the separability assumption can be found in [8] and [9]. More results can be found in [15, 17, 22].

The least squares solution assumes that exact information for the point spread functions, $\mathbf{h}_{ij}$, and the weights, w_{ij}, is available. For single channel image restoration problems, it has been demonstrated [14, 18, 19] that total least squares (TLS)

methods, in particular structured total least squares (STLS) [21] methods, yield better results than the ordinary least squares method when error exists in the blurring matrices. In this paper, we consider applying STLS for the multichannel image restoration problem. Our algorithm extends the basic algorithm presented in [19] and incorporates an efficient preconditioning technique.

The rest of this paper is organized as follows. In section 2, we give a description of the regularized STLS problem. In section 3, we derive our algorithm for solving the regularized STLS problem. In section 4, we propose an efficient way to precondition the linear systems encountered in section 3. Experimental results are shown in section 5, and section 6 concludes this paper.

2. Tikhonov's regularization of the STLS problem. In general the matrix H in (1.3) is ill conditioned. The method of regularization can be used to stabilize the solution. In the classical Tikhonov regularization [11], stability is attained by introducing a penalty term that measures the "irregularity" of the solution. One of the widely used regularization techniques for color image restoration is the one proposed by Galatsanos et al. in [9].

An important concept in the regularization technique of Galatsanos et al. is the "weighted three-dimensional (3D) Laplacian." Let

$$\boldsymbol{\beta} = [\beta_{rg} \quad \beta_{rb} \quad \beta_{gb}]^T = \left[\frac{\|\mathbf{f}_r\|_2}{\|\mathbf{f}_g\|_2} \quad \frac{\|\mathbf{f}_r\|_2}{\|\mathbf{f}_b\|_2} \quad \frac{\|\mathbf{f}_g\|_2}{\|\mathbf{f}_b\|_2} \right]^T, \tag{2.1}$$

and let L be the 2D Laplacian matrix of appropriate size. The "weighted 3D Laplacian" matrix is defined as follows:

$$R(\boldsymbol{\beta}) = \begin{bmatrix} 2I + L & -\beta_{rg} I & -\beta_{rb} I \\ -\frac{1}{\beta_{rg}} I & 2I + L & -\beta_{gb} I \\ -\frac{1}{\beta_{rb}} I & -\frac{1}{\beta_{gb}} I & 2I + L \end{bmatrix}.$$

Note that

$$R(\boldsymbol{\beta}) = D \hat{L} D^{-1},$$

where

$$D = \begin{bmatrix} \|\mathbf{f}_r\|_2 I & 0 & 0 \\ 0 & \|\mathbf{f}_g\|_2 I & 0 \\ 0 & 0 & \|\mathbf{f}_b\|_2 I \end{bmatrix}, \quad \hat{L} = \begin{bmatrix} 2I + L & -I & -I \\ -I & 2I + L & -I \\ -I & -I & 2I + L \end{bmatrix}.$$

Therefore, the operation $R(\boldsymbol{\beta})\mathbf{f}$ can be interpreted as applying the standard 3D Laplacian (with circulant boundary conditions imposed on the third dimension) on a scaled version of $\mathbf{f}$ with the results scaled back afterwards.

Let

$$\Omega = \begin{bmatrix} \alpha_r I & 0 & 0 \\ 0 & \alpha_g I & 0 \\ 0 & 0 & \alpha_b I \end{bmatrix},$$

where α_r, α_g, and α_b are positive constants, and I is the identity matrix of appropriate size.

Denote the elementwise square root of the matrix Ω by $\sqrt{\Omega}$ (note that Ω is a diagonal matrix with positive diagonal elements). The regularization technique of Galatsanos et al. takes the following form:

$$\min_{\mathbf{f}} \|\mathbf{u} - H\mathbf{f}\|_2^2 + \|\sqrt{\Omega}R(\beta)\mathbf{f}\|_2^2. \tag{2.2}$$

In this equation, Ω is used to control the degree of regularization. The constants α_r, α_g, and α_b are called the regularization parameters, and R is called the regularization operator. Notice that R depends on the vector $\boldsymbol{\beta}$, which further depends on the solution. In the literature, it is common practice to use an estimate of $\boldsymbol{\beta}$ rather than the exact $\boldsymbol{\beta}$. However, our algorithm will solve for $\boldsymbol{\beta}$ as part of the problem.

From a statistical point of view, if H is known exactly, the solution to (2.2) can be interpreted as the "maximum a posteriori estimator" of the original image. In this interpretation, the regularization term represents prior information on the original image, the regularization operator reflects the type of prior information, and the parameters α_r, α_g, and α_b indicate our level of confidence in the prior information.

As mentioned earlier, the least squares method assumes that exact information of the blurring matrix is available. In practice, the blurring matrix may also be contaminated by noise. The TLS method allows for the possibility of error in the elements of the blurring matrix H, so that the modified blurring matrix is given by $\tilde{H} = H + E$, where E is the error matrix to be determined. For image restoration problems, because H exhibits certain structures, it is more appropriate to apply STLS [21]. The STLS method requires that $\tilde{H}$ has the same structure as H.

In this paper, our focus is on cases where our knowledge of H_{ij}, for $i, j \in \{r, g, b\}$, is precise but that of $\mathbf{w}$ is inexact. However, the basic algorithms of this paper can be easily modified to handle the case in which our knowledge of H_{ij} is also inexact; see, for instance, [7, 19].

Let

$$\tilde{\mathbf{w}} = [\tilde{w}_{rr} \quad \tilde{w}_{gr} \quad \tilde{w}_{br} \quad \tilde{w}_{rg} \quad \tilde{w}_{gg} \quad \tilde{w}_{bg} \quad \tilde{w}_{rb} \quad \tilde{w}_{gb} \quad \tilde{w}_{bb} \quad]^T$$

be the modified $\mathbf{w}$. From our assumption, it follows that the modified blurring matrix is parameterized by $\tilde{\mathbf{w}}$. Therefore, the regularized STLS problem for color image restoration takes the following form:

$$\min_{\mathbf{f},\tilde{\mathbf{w}}} \|H(\tilde{\mathbf{w}})\mathbf{f} - \mathbf{u}\|_2^2 + \|\sqrt{\Omega}R(\boldsymbol{\beta})\mathbf{f}\|_2^2 + \alpha_h\|\mathbf{w} - \tilde{\mathbf{w}}\|_2^2. \tag{2.3}$$

The last term in this formula can be interpreted as a penalty term for the "irregularity" of the error in $\mathbf{w}$, and α_h can be interpreted as the regularization parameter used to control the degree of regularization on that error.

For simplicity, let

$$\begin{aligned}\mathbf{z} &= \tilde{\mathbf{w}} - \mathbf{w} \\ &= [z_{rr} \quad z_{gr} \quad z_{br} \quad z_{rg} \quad z_{gg} \quad z_{bg} \quad z_{rb} \quad z_{gb} \quad z_{bb}]^T;\end{aligned}$$

then (2.3) becomes

$$\min_{\mathbf{f},\mathbf{z}} \|H(\mathbf{w} + \mathbf{z})\mathbf{f} - \mathbf{u}\|_2^2 + \|\sqrt{\Omega}R(\boldsymbol{\beta})\mathbf{f}\|_2^2 + \alpha_h\|\mathbf{z}\|_2^2. \tag{2.4}$$

Experimental results showed that this formula tends to yield a darker version of the original image and a $\tilde{\mathbf{w}}$ whose elements are systematically larger than those of $\mathbf{w}$

(see Figure 5.8). This way it keeps the residual norm almost unchanged but reduces the "irregularity" of the solution.

To overcome this problem, we add the following constraints to (2.3):

$$\begin{aligned} z_{rr} + z_{gr} + z_{br} &= 0, \\ z_{rg} + z_{gg} + z_{bg} &= 0, \\ z_{rb} + z_{gb} + z_{bb} &= 0. \end{aligned} \tag{2.5}$$

That is, we require the sum of the modified weights for each observed channel to be the same as that of the original ones. Equation (2.5) is an expression of the conservation of intensity values in the color image.

This looks as if an unconstrained optimization problem has been turned into a constrained one. However, since

$$\begin{aligned} z_{br} &= -z_{rr} - z_{gr}, \\ z_{bg} &= -z_{rg} - z_{gg}, \\ z_{bb} &= -z_{rb} - z_{gb}, \end{aligned}$$

now the modified blurring matrix, $\tilde{H}$, is parameterized by only six variables, z_{rr}, z_{gr}, z_{rg}, z_{gg}, z_{rb}, and z_{gb}. Let

$$\hat{\mathbf{z}} = [z_{rr} \quad z_{gr} \quad z_{rg} \quad z_{gg} \quad z_{rb} \quad z_{gb}]^T;$$

then $\hat{\mathbf{z}}$ contains all the information in $\mathbf{z}$.

Note that if the constraints stated in (2.5) are satisfied, then $\mathbf{z} = \Gamma\hat{\mathbf{z}}$ with

$$\Gamma = \operatorname{diag}(\hat{\Gamma}, \hat{\Gamma}, \hat{\Gamma}), \quad \hat{\Gamma} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -1 & -1 \end{bmatrix}.$$

Therefore,

$$\|\mathbf{z}\|_2^2 = \hat{\mathbf{z}}^T \Gamma^T \Gamma \hat{\mathbf{z}}.$$

It is easy to check that the matrix S defined as follows is symmetric positive definite and satisfies $S^2 = \Gamma^T\Gamma$:

$$S = \operatorname{diag}(\hat{S}, \hat{S}, \hat{S}), \quad \text{where } \hat{S} = \begin{bmatrix} \frac{\sqrt{3}+1}{2} & \frac{\sqrt{3}-1}{2} \\ \frac{\sqrt{3}-1}{2} & \frac{\sqrt{3}+1}{2} \end{bmatrix}. \tag{2.6}$$

Therefore, $\|\mathbf{z}\|_2 = \|S\hat{\mathbf{z}}\|_2$.

It follows that our problem can be stated as

$$\min_{\mathbf{f},\hat{\mathbf{z}}} \|H(\mathbf{w} + \Gamma\hat{\mathbf{z}})\mathbf{f} - \mathbf{u}\|_2^2 + \|\sqrt{\Omega}R(\boldsymbol{\beta})\mathbf{f}\|_2^2 + \alpha_h\|S\hat{\mathbf{z}}\|_2^2. \tag{2.7}$$

An important issue in the regularization is the choice of the regularization parameters, α_r, α_g, α_b, and α_h. The optimal value for these parameters depends on the noise level of the observed image and estimated point spread functions. For the least squares solution, Ng and Bose proposed an efficient way to find the optimal $[\alpha_r \quad \alpha_g \quad \alpha_b]$ in [15]. Based on the statistical interpretation of the regularization term, $[\alpha_r \quad \alpha_g \quad \alpha_b]$ reflects our level of confidence in the prior information, which is

largely determined by the noise level on the right-hand side. Considering the TLS solution is also the least squares solution of a problem that has the same right-hand side but a different blurring matrix, we expect the optimal values for $[\alpha_r \quad \alpha_g \quad \alpha_b]$ in the TLS solution to be close to those in the least squares solution. Therefore, we simply use the optimal $[\alpha_r \quad \alpha_g \quad \alpha_b]$ computed by the least squares method for our TLS solution. In order to find the optimal α_h, we use an approach that is similar to the approach employed by Chan and Wong in [4]. That is, we start with a small value for α_h and increase it until a satisfactory solution is found by visually checking the recovered image. One can also use a more systematic approach such as the L-curve method [11], which tries to find a good balance between yielding small TLS residual and "regularity" of the recovered $\hat{\mathbf{z}}$ in (2.7).

3. Solving the regularized STLS problem. In [21], Rosen, Park, and Glick presented an iterative algorithm for structured total least norm in 1-, 2-, and infinity-norms. Pruessner and O'Leary extended the algorithm of Rosen, Park, and Glick to include regularization in all of these norms and demonstrated its use on single channel image deblurring in [19]. For the 2-norm case, the basic idea of the technique of Rosen, Park, and Glick is to approximate the cost function by a quadratic function and minimize that quadratic function at each iteration. Here we show that this technique can be extended to solve (2.7).

For convenience, let

$$\Psi(\mathbf{f}, \hat{\mathbf{z}}) = \|H(\mathbf{w} + \Gamma\hat{\mathbf{z}})\mathbf{f} - \mathbf{u}\|_2^2 + \|\sqrt{\Omega}R(\boldsymbol{\beta})\mathbf{f}\|_2^2 + \alpha_h\|S\hat{\mathbf{z}}\|_2^2;$$

then (2.7) becomes

$$\min_{\mathbf{f},\hat{\mathbf{z}}} \Psi(\mathbf{f}, \hat{\mathbf{z}}). \tag{3.1}$$

If the elements of $\boldsymbol{\beta}$ are constants, then this problem can be solved in a way similar to how Pruessner and O'Leary solved the single channel image deblurring problem in [19]. Taking into account that elements of $\boldsymbol{\beta}$ are not constants, we propose solving (2.7) as follows. At each iteration, we fix $\boldsymbol{\beta}$, approximate the cost function by a quadratic function, and minimize that function. At the end of each iteration, we use the new guess for $\mathbf{f}$ to update $\boldsymbol{\beta}$.

If $\boldsymbol{\beta}$ is fixed, then

$$\begin{aligned}\Psi(\mathbf{f} + \Delta\mathbf{f}, \hat{\mathbf{z}} + \Delta\hat{\mathbf{z}}) = {} & \|H(\mathbf{w} + \Gamma\hat{\mathbf{z}} + \Gamma\Delta\hat{\mathbf{z}})(\mathbf{f} + \Delta\mathbf{f}) - \mathbf{u}\|_2^2 \\ & + \|\sqrt{\Omega}R(\boldsymbol{\beta})(\mathbf{f} + \Delta\mathbf{f})\|_2^2 + \alpha_h\|S(\hat{\mathbf{z}} + \Delta\hat{\mathbf{z}})\|_2^2.\end{aligned} \tag{3.2}$$

Ignoring higher order terms, we get

$$\begin{aligned}\Psi(\mathbf{f} + \Delta\mathbf{f}, \hat{\mathbf{z}} + \Delta\hat{\mathbf{z}}) \approx {} & \|H(\mathbf{w} + \Gamma\hat{\mathbf{z}})(\mathbf{f} + \Delta\mathbf{f}) + H(\Gamma\Delta\hat{\mathbf{z}})\mathbf{f} - \mathbf{u}\|_2^2 \\ & + \|\sqrt{\Omega}R(\boldsymbol{\beta})(\mathbf{f} + \Delta\mathbf{f})\|_2^2 + \alpha_h\|S(\hat{\mathbf{z}} + \Delta\hat{\mathbf{z}})\|_2^2.\end{aligned} \tag{3.3}$$

Let

$$\begin{aligned}\hat{\Psi}(\Delta\mathbf{f}, \Delta\hat{\mathbf{z}}) = {} & \|H(\mathbf{w} + \Gamma\hat{\mathbf{z}})(\mathbf{f} + \Delta\mathbf{f}) + H(\Gamma\Delta\hat{\mathbf{z}})\mathbf{f} - \mathbf{u}\|_2^2 \\ & + \|\sqrt{\Omega}R(\boldsymbol{\beta})(\mathbf{f} + \Delta\mathbf{f})\|_2^2 + \alpha_h\|S(\hat{\mathbf{z}} + \Delta\hat{\mathbf{z}})\|_2^2.\end{aligned} \tag{3.4}$$

When $\boldsymbol{\beta}$ is fixed, minimizing $\hat{\Psi}$ is a Gauss–Newton step for minimizing Ψ.

Define the matrix Z as follows:

$$Z = \begin{bmatrix} Z_1 & 0 & 0 \\ 0 & Z_2 & 0 \\ 0 & 0 & Z_3 \end{bmatrix}, \tag{3.5}$$

where

$$\begin{aligned} Z_1 &= [H_{rr}\mathbf{f}_r - H_{br}\mathbf{f}_b \quad H_{gr}\mathbf{f}_g - H_{br}\mathbf{f}_b], \\ Z_2 &= [H_{rg}\mathbf{f}_r - H_{bg}\mathbf{f}_b \quad H_{gg}\mathbf{f}_g - H_{bg}\mathbf{f}_b], \\ Z_3 &= [H_{rb}\mathbf{f}_r - H_{bb}\mathbf{f}_b \quad H_{gb}\mathbf{f}_g - H_{bb}\mathbf{f}_b]. \end{aligned}$$

Then Z satisfies

$$H(\Gamma\Delta\hat{\mathbf{z}})\mathbf{f} = Z\Delta\hat{\mathbf{z}}. \tag{3.6}$$

Therefore, $\hat{\Psi}$ becomes

$$\begin{aligned} \hat{\Psi}(\Delta\mathbf{f}, \Delta\hat{\mathbf{z}}) &= \|H(\mathbf{w} + \Gamma\hat{\mathbf{z}})(\mathbf{f} + \Delta\mathbf{f}) + Z\Delta\hat{\mathbf{z}} - \mathbf{u}\|_2^2 \\ &\quad + \|\sqrt{\Omega}R(\boldsymbol{\beta})(\mathbf{f} + \Delta\mathbf{f})\|_2^2 + \alpha_h\|S(\hat{\mathbf{z}} + \Delta\hat{\mathbf{z}})\|_2^2. \end{aligned} \tag{3.7}$$

It follows that minimizing $\hat{\Psi}$ is equivalent to solving the following linear least squares system:

$$\min_{\Delta\mathbf{f}, \Delta\hat{\mathbf{z}}} \left\| \begin{bmatrix} H(\mathbf{w} + \Gamma\hat{\mathbf{z}}) & Z \\ \sqrt{\Omega}R(\boldsymbol{\beta}) & 0 \\ 0 & \sqrt{\alpha_h}S \end{bmatrix} \begin{bmatrix} \Delta\mathbf{f} \\ \Delta\hat{\mathbf{z}} \end{bmatrix} - \begin{bmatrix} \mathbf{u} - H(\mathbf{w} + \Gamma\hat{\mathbf{z}})\mathbf{f} \\ -\sqrt{\Omega}R(\boldsymbol{\beta})\mathbf{f} \\ -\sqrt{\alpha_h}S\hat{\mathbf{z}} \end{bmatrix} \right\|_2. \tag{3.8}$$

The STLS algorithm for color image restoration is summarized as follows.

ALGORITHM 3.1.
1. Set $\mathbf{f}$ to be the least squares solution, $\hat{\mathbf{z}} = 0$, construct Z according to (3.5)
2. Repeat
   (a) Solve (3.8) for $\Delta\mathbf{f}$, $\Delta\hat{\mathbf{z}}$.
   (b) Set $\mathbf{f} = \mathbf{f} + \Delta\mathbf{f}$, $\hat{\mathbf{z}} = \hat{\mathbf{z}} + \Delta\hat{\mathbf{z}}$,
   (c) construct Z according to (3.5),
   (d) update $\boldsymbol{\beta}$ according to (2.1).
   until $\|\Delta\mathbf{f}\|$, $\|\Delta\hat{\mathbf{z}}\| \leq \epsilon$.

We have no proof that this algorithm converges; however, experimental results indicate that this algorithm converges very quickly, provided that the regularization parameters are not too small. It is recommended that the least squares solution be used as the initial guess.

4. Solving the linear least squares systems efficiently. The main computational task of one iteration of Algorithm 3.1 is to solve the linear least squares system (3.8). The conjugte gradient least squares (CGLS) method [1] is often used to solve large sparse linear least squares systems in the form of

$$\min_{\mathbf{x}} \|\mathbf{b} - A\mathbf{x}\|_2.$$

The convergence of the CGLS method depends on the condition number of the coefficient matrix, A. When the condition number of A is large, the CGLS method may converge slowly; in that case, preconditioners [10, sect. 10.3] can be used to accelerate the convergence. The idea of preconditioning is to apply the regular CGLS to the transformed system

$$\min_{\tilde{\mathbf{x}}} \|\mathbf{b} - \tilde{A}\tilde{\mathbf{x}}\|_2,$$

where $\tilde{A} = AG^{-1}$ and $\tilde{\mathbf{x}} = G\mathbf{x}$. We need G to be such that $M = G^TG$ has a spectrum similar to A^TA and that the linear system $M\mathbf{x} = \mathbf{b}$ is easy to solve. We call G a preconditioner for A, or M a preconditioner for A^TA.

In [7], Fu and Barlow proposed a preconditioner for a linear system that is similar to (3.8). Unfortunately, the system here requires a different preconditioning strategy, because even though the submatrices of H are BTHTHB, the matrix H itself is not BTHTHB.

4.1. Symmetric blurring matrices. For (3.8), since

$$A = \begin{bmatrix} H(\mathbf{w}+\mathbf{z}) & Z \\ \sqrt{\Omega}R(\boldsymbol{\beta}) & 0 \\ 0 & \sqrt{\alpha_h}S \end{bmatrix}, \tag{4.1}$$

we have

$$A^TA = \begin{bmatrix} H(\mathbf{w}+\mathbf{z})^TH(\mathbf{w}+\mathbf{z}) + R(\boldsymbol{\beta})^T\Omega R(\boldsymbol{\beta}) & H(\mathbf{w}+\mathbf{z})^TZ \\ Z^TH(\mathbf{w}+\mathbf{z}) & Z^TZ + \alpha_h S^TS \end{bmatrix}.$$

Let $\hat{M}$ be the upper left block of A^TA, that is,

$$\hat{M} = H(\mathbf{w}+\mathbf{z})^TH(\mathbf{w}+\mathbf{z}) + R(\boldsymbol{\beta})^T\Omega R(\boldsymbol{\beta}).$$

We remark that the inverse of $\hat{M}$ can be easily computed if all the cross- and within-channel point spread functions are symmetric. By "symmetric" we mean that $h(x,y) = h(-x,y) = h(x,-y) = h(-x,-y)$. In fact, Ng and Bose efficiently computed the inverse of a matrix that is quite similar to $\hat{M}$ in [15].

Ng, Chan, and Tang [16] showed that if the point spread function h is symmetric, then the corresponding blurring matrix H with Neumann boundary conditions can be diagonalized by the 2D discrete cosine transform (DCT) [20] matrix C. Therefore,

$$H = \begin{bmatrix} C^T\Lambda_{rr}C & C^T\Lambda_{gr}C & C^T\Lambda_{br}C \\ C^T\Lambda_{rg}C & C^T\Lambda_{gg}C & C^T\Lambda_{bg}C \\ C^T\Lambda_{rb}C & C^T\Lambda_{gb}C & C^T\Lambda_{bb}C \end{bmatrix},$$

where Λ_{ij} (for $i,j \in \{r,g,b\}$) are real diagonal matrices.

The 2D Laplacian matrix with Neumann boundary conditions can also be diagonalized by the 2D DCT matrix. Let

$$L = C^T\Lambda_L C;$$

then

$$R = \begin{bmatrix} C^T(\Lambda_L + 2I)C & C^T(-\beta_{rg}I)C & C^T(-\beta_{rb}I)C \\ C^T(-\frac{1}{\beta_{rg}}I)C & C^T(\Lambda_L + 2I)C & C^T(-\beta_{gb}I)C \\ C^T(-\frac{1}{\beta_{br}}I)C & C^T(-\frac{1}{\beta_{gb}}I)C & C^T(\Lambda_L + 2I)C \end{bmatrix}.$$

It follows that the matrix $\hat{M}$ can be put into the following form:

$$\hat{M} = \begin{bmatrix} C^T\Lambda_{11}C & C^T\Lambda_{12}C & C^T\Lambda_{13}C \\ C^T\Lambda_{12}C & C^T\Lambda_{22}C & C^T\Lambda_{23}C \\ C^T\Lambda_{13}C & C^T\Lambda_{23}C & C^T\Lambda_{33}C \end{bmatrix} = (I \otimes C^T) \begin{bmatrix} \Lambda_{11} & \Lambda_{12} & \Lambda_{13} \\ \Lambda_{12} & \Lambda_{22} & \Lambda_{23} \\ \Lambda_{13} & \Lambda_{23} & \Lambda_{33} \end{bmatrix} (I \otimes C).$$

Here Λ_{ij} are real diagonal matrices. The matrix with diagonal blocks can be permuted into a block diagonal matrix so that $\hat{M}$ is expressed as

$$\hat{M} = \tilde{C}^T \operatorname{diag}(B_{1,1}, \ldots, B_{1,n}, B_{2,1}, \ldots, B_{2,n}, \ldots, B_{n,1}, \ldots, B_{n,n})\tilde{C}, \tag{4.2}$$

where

$$\tilde{C} = P(I \otimes C)$$

for the appropriate permutation matrix P. Here each $B_{k,\ell}$ $(1 \le k, \ell \le n)$ is a 3×3 symmetric matrix, and we consider an $n \times n$ image in each color channel. Next we show that $\hat{M}$ is symmetric and positive definite.

THEOREM 4.1. [illegible] $(\mathbf{w}+\mathbf{z})_{ij} \ge 0$ [illegible] $i, j \in \{r, g, b\}$ [illegible] $(\mathbf{w}+\mathbf{z})_{ri}$ $(\mathbf{w}+\mathbf{z})_{gi}$ [illegible] $(\mathbf{w}+\mathbf{z})_{bi}$ $i \in \{r, g, b\}$ [illegible] $\hat{M}$ [illegible]. It is clear that $\hat{M}$ is symmetric. According to (4.2), it suffices to show that $B_{k,\ell}$ are positive definite for $1 \le k, \ell \le n$. In fact, the matrices $B_{k,\ell}$ are given by

$$B_{k,\ell} = Q_{k,\ell}(\mathbf{w}+\mathbf{z})^T Q_{k,\ell}(\mathbf{w}+\mathbf{z}) + (U + [\Lambda_L]_{k\ell} I)^T \Phi^2 (U + [\Lambda_L]_{k\ell} I),$$

where

$$Q_{k,\ell}(\mathbf{w}+\mathbf{z}) = \begin{bmatrix} (\mathbf{w}+\mathbf{z})_{rr}[\Lambda_{rr}]_{k\ell} & (\mathbf{w}+\mathbf{z})_{gr}[\Lambda_{gr}]_{k\ell} & (\mathbf{w}+\mathbf{z})_{br}[\Lambda_{br}]_{k\ell} \\ (\mathbf{w}+\mathbf{z})_{rg}[\Lambda_{rg}]_{k\ell} & (\mathbf{w}+\mathbf{z})_{gg}[\Lambda_{gg}]_{k\ell} & (\mathbf{w}+\mathbf{z})_{bg}[\Lambda_{bg}]_{k\ell} \\ (\mathbf{w}+\mathbf{z})_{rb}[\Lambda_{rb}]_{k\ell} & (\mathbf{w}+\mathbf{z})_{gb}[\Lambda_{gb}]_{k\ell} & (\mathbf{w}+\mathbf{z})_{bb}[\Lambda_{bb}]_{k\ell} \end{bmatrix},$$

$$U = \begin{bmatrix} 2 & -\beta_{rg} & -\beta_{rb} \\ -\frac{1}{\beta_{rg}} & 2 & -\beta_{gb} \\ -\frac{1}{\beta_{rb}} & -\frac{1}{\beta_{gb}} & 2 \end{bmatrix}, \quad \text{and} \quad \Phi = \begin{bmatrix} \sqrt{\alpha_r} & 0 & 0 \\ 0 & \sqrt{\alpha_g} & 0 \\ 0 & 0 & \sqrt{\alpha_b} \end{bmatrix}.$$

Let

$$F_{k,\ell} = \begin{bmatrix} Q_{k,\ell}(\mathbf{w}+\mathbf{z}) \\ \Phi(U + [\Lambda_L]_{k,\ell} I) \end{bmatrix}. \tag{4.3}$$

It is clear that $B_{k,\ell}$ is positive definite if and only if $F_{k,\ell}$ has full column rank. We note that the eigenvalues of the discrete Laplacian with the Neumann boundary condition are given by

$$[\Lambda_L]_{k\ell} = 4\sin^2\left(\frac{(k-1)\pi}{2n}\right) + 4\sin^2\left(\frac{(\ell-1)\pi}{2n}\right), \quad 1 \le k, \ell \le n.$$

Therefore, we have, $[\Lambda_L]_{1,1} = 0$. It implies that $F_{k,\ell}$ has full column rank for k and ℓ, except when $k = \ell = 1$.

Next we show that $F_{1,1}$ also has full column rank. It is easy to check that the eigenvalues of U are 0, 3, and 3 and that their corresponding eigenvectors are

$$[\beta_{rb}, \beta_{gb}, 1]^T, \quad [-\beta_{rb}, 0, 1]^T, \quad \text{and } [-\beta_{rg}, 1, 0]^T,$$

respectively. Therefore $\Phi(U + [\Lambda_L]_{k,\ell} I)\mathbf{x} = \mathbf{0}$ if and only if $k = \ell = 1$ and $\mathbf{x}$ is proportional to the nonnegative vector $[\beta_{rb}, \beta_{gb}, 1]^T$. Since discrete point spread functions are normalized such that their sums are equal to 1, it is easy to show that $[\Lambda_{ij}]_{1,1} = 1$. By the assumption, it is clear that $Q_{1,1}(\mathbf{w} + \mathbf{z})$ is a positive matrix, and thus $Q_{1,1}(\mathbf{w} + \mathbf{z})\mathbf{x} = 0$. Thus there is no vector $\mathbf{x}$ such that $F_{1,1}\mathbf{x} = \mathbf{0}$. It follows that $F_{1,1}$ must have full column rank. The result follows. □

It follows that

$$\hat{M}^{-1} = (I \otimes C^T) P^T \operatorname{diag}(B_{1,1}^{-1}, \ldots, B_{1,n}^{-1}, B_{2,1}^{-1}, \ldots, B_{2,n}^{-1}, \ldots, B_{n,1}^{-1}, \ldots, B_{n,n}^{-1}) P (I \otimes C).$$

Therefore, the inverse of $\hat{M}$ can be computed by performing several 2D DCTs and computing the inverses of n^2 3×3 matrices. The preconditioner $\hat{M}$ can also be factored into $\hat{M} = \hat{G}^T \hat{G}$, where

$$G = \operatorname{diag}(G_{1,1}, \ldots, G_{1,n}, G_{2,1}, \ldots, G_{2,n}, \ldots, G_{n,1}, \ldots, G_{n,n}) P (I \otimes C),$$

and $B_{k,\ell} = G_{k,\ell}^T G_{k,\ell}, 1 \le k, \ell \le n$, are the result of either the Cholesky factorization of $B_{k,\ell}$ or the QR factorization of $F_{k,\ell}$ in (4.3).

When all the cross- and within-channel point spread functions are symmetric, we propose to precondition $A^T A$ by the matrix M defined as follows:

$$M = \begin{bmatrix} \hat{M} & 0 \\ 0 & Z^T Z + \alpha_h S^T S \end{bmatrix}.$$

Since the eigenvalues of S are 1 and $\sqrt{3}$ and $\alpha_h > 0$, the matrix $Z^T Z + \alpha_h S^T S$ is positive definite, and the preconditioner M is also positive definite. It is clear that $A^T A$, when preconditioned by M, is different from the identity matrix only at the last six rows and last six columns, which means it is the identity matrix plus a rank-12 change. Thus preconditioned CGLS is guaranteed to converge to the exact solution within 13 iterations.

4.2. Nonsymmetric blurring matrices. If not all the cross- and within-channel point spread functions are symmetric, we suggest constructing the preconditioner according to the symmetric part of the point spread functions. In [13], Kwan and Ng proved the following theorem and lemma.

THEOREM 4.2. $\mathbf{h}$ H_n $\mathbf{h}$ $c(H_n)$ H_n $\mathbf{h}_s$

$$\mathbf{h}_s(i,j) = (\mathbf{h}(i,j) + \mathbf{h}(i,-j) + \mathbf{h}(-i,j) + \mathbf{h}(-i,-j))/4$$

LEMMA 4.3. $\mathbf{h}$ H_n $\mathbf{h}$ $\mathbf{h}$ $\mathbf{h}_s$

$$\delta = \sum_{i,j} |\mathbf{h}(i,j) - \mathbf{h}_s(i,j)|.$$

$$\|H_n - c(H_n)\|_1, \|H_n - c(H_n)\|_\infty \leq 4\delta,$$

$c(H_n)$ H_n

Here we consider the preconditioner for A^TA to be given by

$$M_c = \begin{bmatrix} c(H(\mathbf{w}+\mathbf{z}))^T c(H(\mathbf{w}+\mathbf{z})) + R(\boldsymbol{\beta})^T \Omega R(\boldsymbol{\beta}) & 0 \\ 0 & Z^TZ + \alpha_h S^T S \end{bmatrix}, \tag{4.4}$$

where

$$c(H(\mathbf{w}+\mathbf{z})) = \begin{bmatrix} (\mathbf{w}+\mathbf{z})_{rr} c(H_{rr}) & (\mathbf{w}+\mathbf{z})_{gr} c(H_{gr}) & (\mathbf{w}+\mathbf{z})_{br} c(H_{br}) \\ (\mathbf{w}+\mathbf{z})_{rg} c(H_{rg}) & (\mathbf{w}+\mathbf{z})_{gg} c(H_{gg}) & (\mathbf{w}+\mathbf{z})_{bg} c(H_{bg}) \\ (\mathbf{w}+\mathbf{z})_{rb} c(H_{rb}) & (\mathbf{w}+\mathbf{z})_{gb} c(H_{gb}) & (\mathbf{w}+\mathbf{z})_{bb} c(H_{bb}) \end{bmatrix}.$$

Using the above lemma, we have the following result. First, recall that

$$\|\mathbf{w}+\mathbf{z}\|_\infty = \max_{i,j \in \{r,g,b\}} (\mathbf{w}+\mathbf{z})_{ij}. \tag{4.5}$$

LEMMA 4.4. A (4.1) M_c (4.4)

$$\max_{i,j \in \{r,g,b\}} \|H_{ij} - c(H_{ij})\|_{1,\infty} \leq 4\delta,$$

$$\|H(\mathbf{w}+\mathbf{z})^T H(\mathbf{w}+\mathbf{z}) - c(H(\mathbf{w}+\mathbf{z}))^T c(H(\mathbf{w}+\mathbf{z}))\|_2 \leq 8\delta \|\mathbf{w}+\mathbf{z}\|_\infty.$$

. We note that

$$\begin{aligned} \|H_{ij}^T H_{k\ell} - c(H_{ij})^T c(H_{k\ell})\| &\leq \|H_{ij}^T [H_{k\ell} - c(H_{k\ell})]\| + \|[H_{ij}^T - c(H_{ij})^T] c(H_{k\ell})\| \\ &\leq \|H_{ij}^T\| \|[H_{k\ell} - c(H_{k\ell})]\| + \|[H_{ij}^T - c(H_{ij})^T]\| \|c(H_{k\ell})\| \end{aligned}$$

with $i,j,k,\ell \in \{r,g,b\}$. Since the discrete point spread function is normalized such that its sum is equal to 1,

$$\|H_{ij}^T\|_1 = \|H_{ij}^T\|_\infty = \|c(H_{ij})\|_1 = \|c(H_{ij})\|_\infty = 1.$$

Using the above results and the statement (4.5), we can estimate

$$\|H(\mathbf{w}+\mathbf{z})^T H(\mathbf{w}+\mathbf{z}) - c(H(\mathbf{w}+\mathbf{z}))^T c(H(\mathbf{w}+\mathbf{z}))\|_{1,\infty} \leq 8\delta \|\mathbf{w}+\mathbf{z}\|_\infty.$$

Hence the result follows. □

According to Lemma 4.4, we see that if the point spread function is close to symmetric, then M_c will be a good approximation for A^TA. It follows that if we construct the matrix $\hat{M}$ according to the symmetric part of the point spread functions, then the matrix M_c defined as in (4.4) will be a good preconditioner for A^TA.

To show that the spectra of the preconditioned matrices are clustered around 1, we consider a sequence of blurring matrices and investigate the spectra of the sequence of the preconditioned blurring matrices. Therefore, we assume that the given blurring matrix H_{ij} comes from a fixed singly infinite matrix H_{ij}^∞. We associate with H_{ij}^∞ the function ϕ_{ij} where the entries of H_{ij} and H_{ij}^∞ are the Fourier coefficients of ϕ_{ij}. The

function ϕ_{ij} is called the [illegible] of the sequence of blurring matrices; see [3] for details. With the generating function ϕ_{ij} for the blurring matrix H_{ij}, it can be shown that the eigenvalues of the matrix $c(H_{ij})$ are given by

$$\phi_{ij}\left(\frac{(k-1)\pi}{2n}, \frac{(\ell-1)\pi}{2n}\right), \quad 1 \le k, \ell \le n, \quad i, j \in \{r, g, b\},$$

and the largest eigenvalue of $c(H_{ij})$ is given by $\phi_{ij}(0,0)$ when $\ell = k = 1$; see, for instance, [3, 13]. Since discrete point spread functions are normalized such that their sums are equal to 1, it is easy to show that $\phi_{ij}(0,0) = 1$. Next we show that the smallest eigenvalue of M_c is uniformly bounded below away from 0 by a positive constant independent of the size of M_c.

THEOREM 4.5. [illegible] $(\mathbf{w}+\mathbf{z})_{ij} \ge 0$ [illegible] $i, j \in \{r, g, b\}$ [illegible] $(\mathbf{w}+\mathbf{z})_{ri}$ $(\mathbf{w}+\mathbf{z})_{gi}$ [illegible] $(\mathbf{w}+\mathbf{z})_{bi}$ $(i \in \{r, g, b\})$ [illegible] γ_{min} [illegible] M_c [illegible] M_c [illegible] γ_{min}

[illegible]. We define for simplicity that

$$\xi(x, y) = \lambda_{\min}(F_{x,y}^T F_{x,y}),$$

where

$$F_{x,y} = \begin{bmatrix} Q_{x,y}(\mathbf{w}+\mathbf{z}) \\ \Phi(U + \psi(x,y)I) \end{bmatrix},$$

$$Q_{x,y}(\mathbf{w}+\mathbf{z}) = \begin{bmatrix} (\mathbf{w}+\mathbf{z})_{rr}\phi_{rr}(x,y) & (\mathbf{w}+\mathbf{z})_{gr}\phi_{gr}(x,y) & (\mathbf{w}+\mathbf{z})_{br}\phi_{br}(x,y) \\ (\mathbf{w}+\mathbf{z})_{rg}\phi_{rg}(x,y) & (\mathbf{w}+\mathbf{z})_{gg}\phi_{gg}(x,y) & (\mathbf{w}+\mathbf{z})_{bg}\phi_{bg}(x,y) \\ (\mathbf{w}+\mathbf{z})_{rb}\phi_{rb}(x,y) & (\mathbf{w}+\mathbf{z})_{gb}\phi_{gb}(x,y) & (\mathbf{w}+\mathbf{z})_{bb}\phi_{bb}(x,y) \end{bmatrix},$$

$$\psi(x, y) = 4\sin^2 x + 4\sin^2 y.$$

With this notation, we have shown in Theorem 4.1 that $F_{x,y}$ has full column rank, and hence the smallest eigenvalue of the matrix $F_{x,y}^T F_{x,y}$ is positive. Therefore, $\xi(x,y) > 0$ for all $(x,y) \in [0, \pi/2]^2$. Since $\xi(x,y)$ is a continuous function on $[0, \pi/2]^2$, there exists a constant γ_{min}, independent of the size of the matrix, such that $\xi(x,y) \ge \gamma_{min} > 0$ for all $(x,y) \in [0, \pi/2]^2$. Hence the result follows. □

By using Lemma 4.4 and Theorem 4.5, we can prove the following theorem.

THEOREM 4.6. [illegible] $(\mathbf{w}+\mathbf{z})_{ij} \ge 0$ [illegible] $i, j \in \{r, g, b\}$ [illegible] $(\mathbf{w}+\mathbf{z})_{ri}$ $(\mathbf{w}+\mathbf{z})_{gi}$ [illegible] $(\mathbf{w}+\mathbf{z})_{bi}$ $i \in \{r, g, b\}$ [illegible]

$$\max_{i,j \in \{r,g,b\}} \|H_{ij} - c(H_{ij})\|_{1,\infty} \le 4\delta.$$

[illegible]

$$M_c^{-1} A^T A = I + E_1 + E_2,$$

[illegible] E_1 [illegible] 12 [illegible] $\|E_2\|_2 \le 8\delta \|\mathbf{w}+\mathbf{z}\|_\infty / \gamma_{min}$ [illegible] γ_{min} [illegible] 4.5

[illegible]. We note that

$$\begin{aligned} & M_c^{-1} A^T A \\ &= I + M_c^{-1}(A^T A - M_c) \\ &= I + M_c^{-1} \begin{bmatrix} H(\mathbf{w}+\mathbf{z})^T H(\mathbf{w}+\mathbf{z}) - c(H(\mathbf{w}+\mathbf{z}))^T c(H(\mathbf{w}+\mathbf{z})) & H(\mathbf{w}+\mathbf{z})^T Z \\ Z^T H(\mathbf{w}+\mathbf{z}) & 0 \end{bmatrix}. \end{aligned}$$

The result follows by noting that the matrix

$$M_c^{-1}\begin{bmatrix} 0 & H(\mathbf{w}+\mathbf{z})^T Z \\ Z^T H(\mathbf{w}+\mathbf{z}) & 0 \end{bmatrix}$$

is a matrix of rank 12 and

$$\begin{aligned} &\left\| M_c^{-1}\begin{bmatrix} (H(\mathbf{w}+\mathbf{z})^T H(\mathbf{w}+\mathbf{z}) - c(H(\mathbf{w}+\mathbf{z}))^T c(H(\mathbf{w}+\mathbf{z}))) & 0 \\ 0 & 0 \end{bmatrix}\right\|_2 \\ &\le \|M_c^{-1}\|_2 \left\|\begin{bmatrix} H(\mathbf{w}+\mathbf{z})^T H(\mathbf{w}+\mathbf{z}) - c(H(\mathbf{w}+\mathbf{z}))^T c(H(\mathbf{w}+\mathbf{z})) & 0 \\ 0 & 0 \end{bmatrix}\right\|_2 \\ &\le 8\delta \|M_c^{-1}\|_2 \|\mathbf{w}+\mathbf{z}\|_\infty \le 8\delta \|\mathbf{w}+\mathbf{z}\|_\infty / \gamma_{min}. \qquad \square \end{aligned}$$

According to Theorem 4.6, if δ is sufficiently small (i.e., if the point spread function is quite symmetric), then the spectra of the preconditioned matrices $M_c^{-1}A^TA$ are clustered around 1, and therefore the preconditioned conjugate gradient method will converge very quickly. In the next section, this result is confirmed by our experimental results.

5. Experimental results. The image shown in Figure 5.1 is used for computer simulation.

5.1. The first test. The discretized point spread functions are as follows:

$$\mathbf{h}_{rr} = \mathbf{h}_{gr} = \mathbf{h}_{br} = \begin{bmatrix} \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{bmatrix},$$

$$\mathbf{h}_{rg} = \mathbf{h}_{gg} = \mathbf{h}_{bg} = \begin{bmatrix} 0 & 0.125 & 0 \\ 0.125 & 0.5 & 0.125 \\ 0 & 0.125 & 0 \end{bmatrix},$$

$$\mathbf{h}_{rb} = \mathbf{h}_{gb} = \mathbf{h}_{bb} = \begin{bmatrix} 0.0625 & 0.125 & 0.0625 \\ 0.125 & 0.25 & 0.125 \\ 0.0625 & 0.125 & 0.0625 \end{bmatrix}.$$

FIG. 5.1. *The original image.*

FIG. 5.2. *The observed image.*

The weights used to generate the observed channels are as follows:

$$\mathbf{w} = [0.5 \quad 0.3 \quad 0.2 \quad 0.25 \quad 0.5 \quad 0.25 \quad 0.2 \quad 0.3 \quad 0.5]^T.$$

A noise returned by the MATLAB command "0.01*(rand-0.5)" [6] is added to the observed channels. Figure 5.2 shows the observed image.

The estimated weights are set to

$$\mathbf{w} = [0.45 \quad 0.33 \quad 0.22 \quad 0.275 \quad 0.45 \quad 0.275 \quad 0.22 \quad 0.33 \quad 0.55]^T.$$

The relative error on the estimated weights is 10%.

We use the peak signal to noise ratio (PSNR) to measure the quality of the recovered images. The PSNR of a discretized color image is defined as

$$PSNR(\mathbf{f}) = 20log_{10}\sqrt{\frac{1}{\|\mathbf{f} - \mathbf{f}_{true}\|_2^2/(3mn)}},$$

where $\mathbf{f}_{true}$ is the true image, and m and n are the size of the image.

Two least squares solutions are computed. One is obtained by using the $\boldsymbol{\beta}$ calculated from the observed image; the other is obtained by using the $\boldsymbol{\beta}$ calculated from the original image. Figures 5.3 and 5.4 show these two images. For both solutions, $[\alpha_r \quad \alpha_g \quad \alpha_b] = [0.0001 \quad 0.0001 \quad 0.0001]$ is used.

For the TLS solution, $\alpha_h = 9$ yields satisfactory results. The recovered weights are as follows:

$$\mathbf{w} = [0.4943 \quad 0.3137 \quad 0.1920 \quad 0.2555 \quad 0.4688 \quad 0.2757 \quad 0.1902 \quad 0.3292 \quad 0.4807]^T.$$

The relative error of the recovered weights is 5.4%, which is much smaller than that of the original estimates. Figure 5.5 shows the image recovered by the STLS method. The PSNRs of the recovered images indicate that the TLS solution is much better.

Figure 5.6 shows the error norm (measured as $\|[\Delta\mathbf{f}^T \Delta\hat{\mathbf{z}}^T]\|$) at each iteration of Algorithm 3.1. As can be seen, Algorithm 3.1 converged to an error level of 10^{-2} in

FIG. 5.3. *The image recovered by the least squares method. The observed image is used to calculate* $\boldsymbol{\beta}$. *PSNR* $= 27.79db$.

FIG. 5.4. *The image recovered by the least squares method. The original image is used to calculate* $\boldsymbol{\beta}$. *PSNR* $= 28.01db$.

12 iterations. More iterations will further reduce the error norm but will have very little effect on the recovered image.

Two variants of Algorithm 3.1 are also used to compute TLS solutions. One uses a fixed $\boldsymbol{\beta}$, which is calculated from the observed image. The image recovered by this variation is shown in Figure 5.7. The difference between the PSNRs of this image and the one shown in Figure 5.5 is 0.82db, which is not a small value. The difference can be larger if the $\boldsymbol{\beta}$ values based on the observed image are quite different from those based on the recovered image, which will occur if there is significant cross-channel blur and the original image has significantly different intensity levels in the three channels (for example, when the original image is dominated by the red channel). The other variant does not apply the constraints stated in (2.5). That is, the solution

FIG. 5.5. *The image recovered by Algorithm* 3.1. *PSNR* = 34.22*db*.

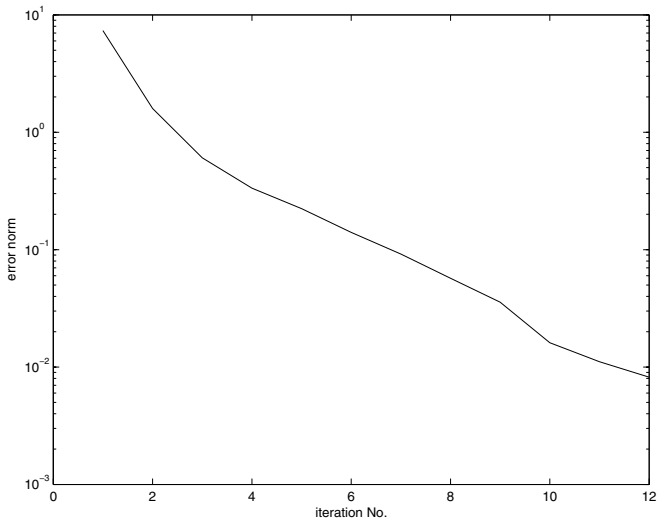


FIG. 5.6. *Error norm at each iteration of Algorithm* 3.1.

of (2.4) is computed. The image recovered by this variant, which has a much lower PSNR, is shown in Figure 5.8. Table 5.1 shows the relative error of each channel when different restoration methods are used, which also indicates that the solution returned by Algorithm 3.1 is the best.

5.2. The second test. We remark that in practice, it is quite possible that no information for the weights is available at all. In that situation, one can use an estimate of 1/3 for all the cross- and within-channel weights. Using the same observed image in Figure 5.2, the recovered weights are as follows:

$$\mathbf{w} = [0.4809 \quad 0.3054 \quad 0.2137 \quad 0.3018 \quad 0.3488 \quad 0.3494 \quad 0.2125 \quad 0.3465 \quad 0.4411]^T.$$

FIG. 5.7. *The image recovered by a variant of Algorithm* 3.1 *(a fixed* $\boldsymbol{\beta}$ *is used). PSNR* $= 33.40db$.

FIG. 5.8. *The image recovered by a variant of Algorithm* 3.1 *(without the constraints stated in* (2.5)). *PSNR* $= 29.47db$.

TABLE 5.1

Relative error of each channel when different restoration methods are used. TLS: total least squares restoration; Algorithm 3.1 *is used. TLS-*1*: total least squares restoration; a fixed* $\boldsymbol{\beta}$ *is used. TLS-*2*: total least squares restoration; the solution of* (2.4) *is computed. LS-*1*: least squares restoration; the observed image is used to calculate* $\boldsymbol{\beta}$*. LS-*2*: least squares restoration; the original image is used to calculate* $\boldsymbol{\beta}$*.*

	TLS	TLS-1	TLS-2	LS-1	LS-2
Red channel	0.0219	0.0224	0.0222	0.0857	0.0826
Green channel	0.0166	0.0251	0.0918	0.0522	0.0492
Blue channel	0.0238	0.0239	0.0674	0.0691	0.0684

The relative error on the estimated weights is 19.19%. The PSNRs of the images recovered by the least squares method are 24.55db and 21.00db when the β are calculated from the original image and the observed image, respectively. The PSNR of the

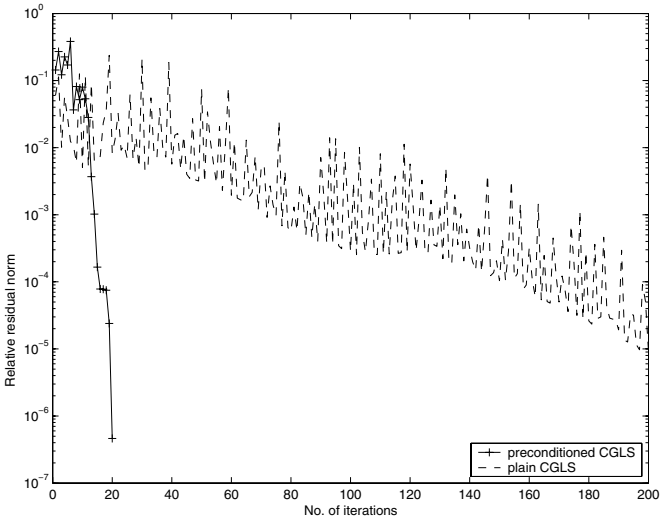


FIG. 5.9. *Relative residual norm at each CGLS iteration.*

image recovered by our proposed method is 29.74db, where the β is calculated from the observed image. Without a doubt, a less accurate estimate of the weights will worsen both the least squares and TLS solutions, but it also makes the TLS solution more likely to recover a better estimate of the weights and therefore outperform the least squares solution.

5.3. The third test. Experiments also showed that the preconditioning technique proposed in section 4 is very effective if the point spread functions are symmetric or close to symmetric. As predicted by theoretical analysis, preconditioned CGLS converges to the exact solution in 13 iterations when the point spread functions are symmetric. When the point spread functions are not symmetric, there is no guarantee that preconditioned CGLS will converge to the exact solution in 13 iterations, but the convergence rate is considerably accelerated if the functions are close to symmetric. This is shown in Figure 5.9. In this example, a noise returned by the MATLAB command "0.01*rand" is added to the elements of the original symmetric point spread functions. The point spread functions are then normalized such that they integrate to 1. As can be seen, preconditioned CGLS converges much faster.

Next we test the performance of the preconditioning techniques for nonsymmetric point spread functions. In the test, we set $H_{rr} = H_{rg} = H_{rb}$, $H_{gr} = H_{gg} = H_{gb}$, and $H_{br} = H_{bg} = H_{bb}$. Their corresponding point spread functions are examples of blurring that occurs in wavefront coding where a cubic phase filter is used to improve depth of field restoration in light efficient wide aperture optical systems [5]. An example of the point spread function for H_{rr} can be found in Figure 5.10. We find in Figure 5.11 that the performance of the preconditioner is poorer than in the previous case, but the convergence rate is considerably accelerated and is much better than the convergence rate of the method without using the preconditioner.

6. Conclusion. In conclusion, we have proposed a regularized STLS algorithm for color image restoration. In our problem setup, we assume that the blurring matrix, as well as the right-hand side, is contaminated by noise. The regularized STLS problem is stated as an unconstrained optimization problem. Our algorithm solves

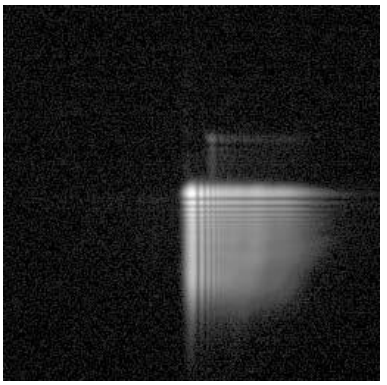


FIG. 5.10. *Image of the nonsymmetric point spread function.*

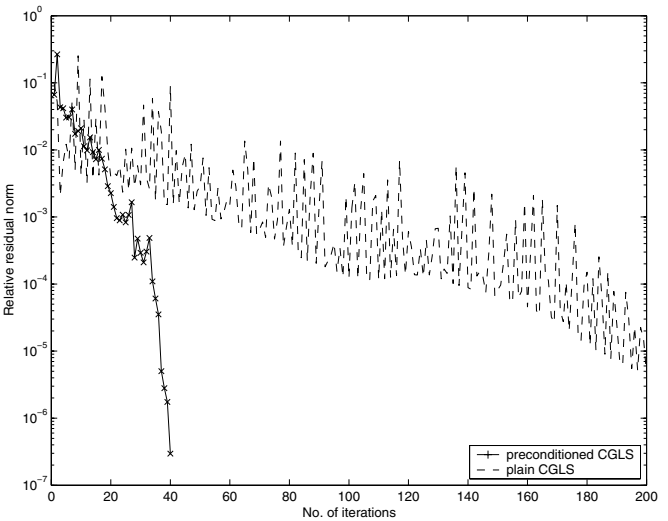


FIG. 5.11. *Relative residual norm at each CGLS iteration.*

the problem iteratively. At each iteration, the objective function is simplified by fixing certain parameters and ignoring high order terms. The simplified version of the objective function can be minimized by solving a linear least squares system. This system is solved by the preconditioned CGLS method. The main contribution of this paper includes solving for the weights of the weighted 3D Laplacian as part of the problem and proposing an efficient preconditioner for the linear systems encountered in the STLS algorithm. Experimental results showed that the STLS method recovers the error on the blurring matrix well, and it yields an image that is superior to the least squares solution.

REFERENCES

[1] A. BJÖRCK AND T. ELFVING, *Accelerated projection methods for computing pseudo-inverse solutions of linear equations*, BIT, 19 (1979), pp. 145–163.

[2] K. BOO AND N. K. BOSE, *Multispectral image restoration with multisensors*, IEEE Trans. Geosci. Remote Sensing, 35 (1997), pp. 1160–1170.

[3] R. H. Chan and M. K. Ng, *Conjugate gradient methods for Toeplitz systems*, SIAM Rev., 38 (1996), pp. 427–482.
[4] T. F. Chan and C. Wong, *Total variation blind deconvolution*, IEEE Trans. Image Process., 7 (1998), pp. 370–375.
[5] E. R. Dowski and W. T. Cathey, *Extended depth of field through wavefront coding*, Applied Optics, 34 (1995), pp. 1859–1866.
[6] D. M. Etter, D. C. Kuncicky, and D. W. Hull, *Introduction to Matlab* 6, Prentice–Hall, Upper Saddle River, NJ, 2002.
[7] H. Fu and J. Barlow, *A regularized structured total least squares algorithm for high resolution image reconstruction*, Linear Algebra Appl., 391 (2004), pp. 75–98.
[8] N. P. Galatsanos and R. T. Chin, *Digital restoration of multichannel images*, IEEE Trans. Acoust. Speech Signal Process., 37 (1989), pp. 415–421.
[9] N. P. Galatsanos, A. K. Katsaggelos, R. T. Chin, and A. D. Hillery, *Least squares restoration of multichannel images*, IEEE Trans. Signal Process., 39 (1991), pp. 2222–2236.
[10] G. H. Golub and C. F. Van Loan, *Matrix Computations*, The Johns Hopkins University Press, Baltimore, MD, 1996.
[11] P. C. Hansen and D. P. O'Leary, *The use of the L-curve in the regularization of discrete ill-posed problems*, SIAM J. Sci. Comput., 14 (1993), pp. 1487–1503.
[12] B. R. Hunt and O. Kubler, *Karhunen-Loeve multispectral image restoration, Part* I: *Theory*, IEEE Trans. Acoust. Speech, Signal Process., 32 (1984), pp. 592–600.
[13] W. Kwan and M. K. Ng, *Iterative Regularization for Ill-posed Imaging Problems using Neumann Boundary Conditions*, in Structured Matrices: Recent Developments in Theory and Computation, Nova Science Publishers, Hauppauge, NY, 2001, pp. 177–190.
[14] V. Z. Mesarović, N. P. Galatsanos, and A. K. Katsaggelos, *Regularized constrained total least squares image restoration*, IEEE Trans. Image Process., 4 (1995), pp. 1096–1108.
[15] M. K. Ng and N. K. Bose, *Fast color image restoration with multisensors*, International Journal of Imaging Systems and Technology, 12 (2002), pp. 189–197.
[16] M. K. Ng, R. H. Chan, and W.-C. Tang, *A fast algorithm for deblurring models with Neumann boundary conditions*, SIAM J. Sci. Comput., 21 (1999), pp. 851–866.
[17] M. K. Ng and W. C. Kwan, *Comments on least squares restoration of multichannel images*, IEEE Trans. Signal Process., 49 (2001), p. 2885.
[18] M. K. Ng, R. J. Plemmons, and F. Pimentel, *A new approach to constrained total least squares image restoration*, Linear Algebra Appl., 316 (2000), pp. 237–258.
[19] A. Pruessner and D. P. O'Leary, *Blind deconvolution using a regularized structured total least norm algorithm*, SIAM J. Matrix Anal. Appl., 24 (2003), pp. 1018–1037.
[20] K. Rao and P. Yip, *Discrete Cosine Transform: Algorithms, Advantages, Applications*, Academic Press, Boston, MA, 1990.
[21] J. B. Rosen, H. Park, and J. Glick, *Total least norm formulation and solution for structured problems*, SIAM J. Matrix Anal. Appl., 17 (1996), pp. 110–126.
[22] A. Tekalp and G. Pavlovic, *Multichannel image modeling and kalman filtering for multispectral image restoration*, Signal Process., 19 (1990), pp. 221–232.

SIAM J. SCI. COMPUT.
Vol. 28, No. 3, pp. 1120–1132
© 2006 Society for Industrial and Applied Mathematics

A FINITE-DIFFERENCE APPROXIMATION OF A TWO-LAYER SYSTEM FOR GROWING SANDPILES*

MAURIZIO FALCONE† AND STEFANO FINZI VITA†

Abstract. We study from a numerical point of view a system of partial differential equations recently proposed in [K. P. Hadeler and C. Kuttler, *Granular Matter*, 2 (1999), pp. 9–18] to model the dynamics of growing sandpiles generated by a vertical source on a flat bounded table. In such a system, an eikonal equation for the standing layer of the pile is coupled to an advection equation for the rolling layer. We analyze a suitable explicit finite difference scheme in one dimension and discuss its properties. A comparison is made with respect to the variational approach of [L. Prigozhin, *European J. Appl. Math.*, 7 (1996), pp. 225–235], particularly in terms of steady-state solutions. Since the effective equilibrium configuration reached by the growing process is not completely clear for this model, we show experiments in one and two dimensions, which characterize the steady-state solutions for different source terms.

Key words. growing piles, granular matter, hyperbolic systems, finite difference schemes

AMS subject classifications. 65M06, 65M12, 70F45

DOI. 10.1137/050629410

1. Introduction.

In recent years several papers have been devoted to the mathematical study of granular matter, and different models have been proposed in the framework of kinetic [3, 4], cellular automata [2, 9, 17], or partial differential equations theories [6, 7, 1, 10, 14, 11, 13].

A typical prototype for granular matter is sand, and one of the simplest phenomena to describe is the following. Let us consider an initially empty table (a bounded open domain Ω of $\mathbb{R}^2$). The sand is poured on a subset of Ω according to the value of a nonnegative function $f(x,t)$ (the source). The pile grows in height with a slope which is always lower than a characteristic value (strongly depending on the physical properties of the granular matter). This means that the sand is added to the pile at every point where f is strictly positive while the slope of the pile profile u remains below the critical value α. When the slope becomes steeper than α the sand rolls down, enlarging the heap base. Physical experiments on real sandpiles show that, for f independent of t in the steady avalanching regime, the profile reaches a steady state in a finite time. The height of the pile stops growing when at some point its base reaches the boundary of the table. After that time the sand will fall down from the table at that point (the tip moves slightly in the opposite direction), so that the pile profile remains unchanged.

We will focus our attention on the numerical simulations based on differential models and in particular on a recent model proposed by Hadeler and Kuttler [13]. Their model is an improvement on the one proposed by de Gennes [10] as a simplification of the one-dimensional model studied by Bouchaud, Cates, Ravi Prakash, and Edwards (i.e., BCRE; see [6, 7]). All the above models use two interacting variables: the standing layer u and the rolling layer v. The interactions between u and v

*Received by the editors April 18, 2005; accepted for publication (in revised form) January 26, 2006; published electronically July 17, 2006. This work was partially supported by MIUR Project 2003 "Metologie Numeriche Avanzate per il Calcolo Scientifico."
http://www.siam.org/journals/sisc/28-3/62941.html
†Dipartimento di Matematica, Università di Roma "La Sapienza," P.le Aldo Moro, 5-00185 Roma, Italy (falcone@mat.uniroma1.it, finzi@mat.uniroma1.it).

are described in [13] by a system of nonlinear partial differential equations for which it is quite delicate to prove existence and uniqueness results. However, a partial characterization of its stationary solutions has been given recently by Cannarsa and Cardaliaguet [8] (see Theorem 2.1). We will construct an approximation scheme for that system and analyze its properties in the one-dimensional case, showing that the scheme mimics several characteristics of the continuous model (Theorem 3.1). Moreover, we propose a natural extension of the scheme to the two-dimensional model and test it in several experiments, comparing the solutions to those generated by a different model (always in the PDE framework). In particular, we will consider the general model for growing sandpiles on an infinite flat table introduced by Aronsson, Evans, and Wu [1] as the limit of fast/slow diffusion evolution equations for the p-Laplace operator when p tends to $+\infty$. Such a variational model, when restricted to a bounded domain, turns out to be analytically equivalent to the degenerate parabolic problem independently studied by Prigozhin [14] for the bounded table case. It is interesting to note that the two models considered in [13] and in [14] have theoretically the same set of admissible equilibria. The precise asymptotic behavior of solutions is well known for the second one and depends only on the geometry of the source support. In contrast, no rigorous mathematical characterization is known for the steady-state solution of the two-layer model. We are able to give a numerical description of this equilibrium configuration, showing that it has in general a larger support and also depends on the source intensity.

It is worth mentioning that a different approximation scheme (a finite difference semidiscretization in time) has been considered in [16] for the two-layer model.

The paper is organized as follows. Section 2 is devoted to the presentation of the three continuous models mentioned above and to a discussion of their asymptotic behavior. In section 3 we present our scheme and prove that it is consistent with the physics of the growing pile problem. Finally, in section 4 we present several numerical experiments and compare our results to those known for the other two models.

2. Three continuous models. In [1] the sequence of solutions $\{u_p\}_p$ of the parabolic equations for the p-Laplace operator in $\mathbb{R}^n \times (0,T)$ is proved to converge almost everywhere, for p tending to $+\infty$, to a solution of the evolutionary variational inequality

$$u(t) \in K: \ (u_t - f, \phi - u) \geq 0 \quad \forall \phi \in K, \ \forall t \in (0,T), \quad u(0) \equiv 0, \tag{1}$$

where

$$K := \{\phi \in W^{1,\infty}(\mathbb{R}^n): \ |\nabla \phi| \leq \alpha\}.$$

To simplify the presentation we hereafter set $\alpha = 1$, since this is not restrictive for the analysis of the models.

In a series of papers (see, e.g., [14]), Prigozhin independently obtained the same variational inequality for the case of a bounded domain Ω (that is, with $K \subset W_0^{1,\infty}(\Omega)$ and the same bound for $\nabla \phi$) as an equivalent formulation of the degenerate parabolic problem

$$u_t = \nabla \cdot (v \nabla u) + f \qquad \text{in } \Omega \times (0,T), \tag{2}$$

$$|\nabla u| \leq 1, \quad |\nabla u| < 1 \Rightarrow v = 0 \qquad \text{in } \Omega \times (0,T), \tag{3}$$

$$u(\cdot,t) = 0 \quad \text{on } \partial\Omega, \qquad u(\cdot,0) = 0 \quad \text{in } \Omega, \tag{4}$$

where $v(x,t) \geq 0$ is an auxiliary function, still unknown, whose positivity indicates that some matter, poured by the source, is rolling down along the surface of the pile in that point. Under suitable hypotheses, it was proved that in the bounded table case problem (1) has a unique solution u, which is a nondecreasing function of time and is also bounded from above by the maximal admissible profile, given by the distance function d from the boundary of Ω:

$$d(x) := \inf_{z \in \partial\Omega} |x - z|.$$

(The discontinuity set $\mathcal{R}$ of ∇d is usually called the [illegible] set or the [illegible] of Ω.) Then, if $f = f(x)$, the solution $u(\cdot,t)$ of (1) will evolve towards an equilibrium state $\overline{u} \in K$; from (2)–(4) we see that $\overline{u}$ solves, for a suitable v, the system

$$-\nabla \cdot (v \nabla u) = f, \quad (1 - |\nabla u|)v = 0 \quad \text{in } \Omega, \quad u|_{\partial\Omega} = 0. \tag{5}$$

As we already mentioned in the introduction, a different continuous model has been proposed by Bouchaud et al. in [6]. They introduced a system of two partial differential equations to govern the growth of a sandpile, seen as the superposition of two distinct layers, the standing and the rolling ones. More recently, following the fundamental works of de Gennes [10] on granular matter, Hadeler and Kuttler [13] have revisited that model, coupling an eikonal-type equation for the standing layer u (which forms heaps and slopes) to an advection-type equation for the small rolling layer v running down the slope. We will use their formulation, which corresponds to the system

$$v_t = \nabla \cdot (v \nabla u) - (1 - |\nabla u|)v + f \qquad \text{in } \Omega \times (0,T), \tag{6}$$

$$u_t = (1 - |\nabla u|)v \qquad \text{in } \Omega \times (0,T), \tag{7}$$

$$u(\cdot,t) = 0 \quad \text{on } \partial\Omega, \qquad u(\cdot,0) = v(\cdot,0) = 0 \quad \text{in } \Omega. \tag{8}$$

We see that the natural Dirichlet boundary condition for the open table is imposed to the standing layer u. In contrast, no condition is given for v at $\partial\Omega$. The reason can be seen by expanding (6):

$$v_t - \nabla u \cdot \nabla v = (\Delta u - (1 - |\nabla u|))v + f\ . \tag{9}$$

For a fixed u, (9) can be interpreted as a linear advection equation in v. Then, the sign of the vector field ∇u on $\partial\Omega$ implies that no boundary condition is needed in this case.

The model based on (6)–(8) is essentially different from the above-mentioned variational model, since now positive rolling layers are allowed during the evolution even before the corresponding standing layer becomes critical (see [15] for a detailed comparison of the two models). This feature can be useful in the modeling of some granular materials. Nevertheless, it is interesting to remark that this different dynamics exhibits the same equilibria of (6)–(8), given by solutions of system (5). In the framework of viscosity solutions theory, such equilibria have been recently characterized by Cannarsa and Cardaliaguet [8], as follows.

THEOREM 2.1. [illegible] $\Omega \subset \mathbb{R}^2$ [illegible] C^2 [illegible] $f \geq 0$ [illegible] Ω [illegible] (5) [illegible] (u,v) [illegible] $u = d$, Ω $v = 0$, $\mathcal{R}$

$$v(x) = \int_0^{\tau(x)} f(x + t\nabla d(x)) \frac{1 - (d(x) + t)k(x)}{1 - d(x)k(x)}\, dt \qquad \forall x \in \Omega \backslash \overline{\mathcal{R}}. \tag{10}$$

[illegible] (u', v') [illegible] $v' = v$ [illegible] Ω [illegible] $u' = d$ [illegible] $\{x \in \Omega : v > 0\}$

In the previous formula, for any point $x \in \Omega$, $k(x)$ denotes the curvature of $\partial\Omega$ at the projection of x onto it, while $\tau(x)$ indicates the so-called normal distance to $\overline{\mathcal{R}}$, defined as

$$\tau(x) = \min\{t \geq 0 : x + t\nabla d(x) \in \overline{\mathcal{R}}\}.$$

By the above result there exists a unique equilibrium solution v for any given source f, and an integral representation formula is given for it outside $\mathcal{R}$ (where $v = 0$). In contrast, for u there is no uniqueness, although every solution must coincide with d in the set where $v > 0$. Then, if we denote by D_f the set of positivity of a (constant in time) source term f in Ω, two distinct cases can occur:

(a) $\mathcal{R} \subset D_f$: the unique possible asymptotic equilibrium for both models is essentially given for the standing layer by the distance function d.

(b) $\mathcal{R}$ is not contained in D_f: the two models behave differently. The steady state for the first model can be proven to be, for the standing layer, the minimum (with respect to f) of the possible equilibria, that is, the union of all maximal circular cones centered in points of D_f. Namely,

$$u^*(x) = \max_{y \in D_f} (d(y) - |x - y|)^+. \tag{11}$$

In the two-layer system, as already remarked, the different dynamics of the rolling layer forces the solution to evolve towards a larger equilibrium configuration $\overline{u}$, which also depends on the source intensity (as clearly shown by the numerical experiments discussed in the last section) and for which a rigorous mathematical characterization is not known.

3. An approximation scheme for the two-layer model. Let us consider the one-dimensional version of (6)–(8). Let $\Omega = (0, 1)$; we want to solve numerically the system

$$v_t = (vu_x)_x - (1 - |u_x|)v + f = vu_{xx} + v_x u_x - (1 - |u_x|)v + f, \tag{12}$$

$$u_t = (1 - |u_x|)v, \tag{13}$$

$$u(0, t) = u(1, t) = 0, \qquad u(x, 0) = v(x, 0) = 0, \quad x \in \Omega\ . \tag{14}$$

Let $x_i = ih$, for $i = 0, 1, \ldots, N$ and $h = \Delta x = 1/N$, denote the nodes of a grid in Ω. We set by definition

$$D^+ u_i = \frac{u_{i+1} - u_i}{h}, \qquad D^- u_i = \frac{u_i - u_{i-1}}{h}. \tag{15}$$

Then we discretize the derivative of u at a node x_i by the difference with maximal absolute value, namely,

$$Du_i \equiv \text{maxmod}(D^- u_i, D^+ u_i), \tag{16}$$

whereas the corresponding derivative of v is approximated by the upwind (with respect to Du_i) finite difference, i.e.,

$$\overline{D} v_i \equiv \begin{cases} D^+ v_i & \text{if } Du_i > 0,\ D^+ u_i > 0, \\ D^- v_i & \text{if } Du_i < 0,\ D^- u_i < 0, \\ 0 & \text{otherwise.} \end{cases}$$

We discretize time by a fixed step Δt and use the standard notation, where z_i^n denotes the approximate value of a function $z(x,t)$ for $t = t_n \equiv n\Delta t$ and $x = x_i$. Using the forward Euler method for the time derivatives, we obtain our fully explicit scheme for the solution of (12)–(14):

$$v_i^{n+1} = v_i^n + \Delta t \left[v_i^n D^2 u_i^n + \overline{D} v_i^n D u_i^n - (1 - |D u_i^n|) v_i^n + f_i \right], \tag{17}$$

$$u_i^{n+1} = u_i^n + \Delta t (1 - |D u_i^n|) v_i^n, \tag{18}$$

$$u_1^n = u_N^n = 0 \quad \forall n, \quad u_i^0 = v_i^0 = 0 \quad \forall i, \tag{19}$$

where

$$D^2 u_i^n \equiv \frac{u_{i-1}^n - 2u_i^n + u_{i+1}^n}{h^2} \tag{20}$$

denotes the standard central finite difference scheme for the term u_{xx}, and (19) gives the natural discrete versions of boundary and initial conditions (14).

Let us briefly present some properties of the above scheme, which preserves the physical characteristics of the continuous model. Further details and complete proofs will be presented in a forthcoming paper [12].

THEOREM 3.1. [illegible] $f \geq 0$, [illegible] Ω [illegible]

$$\frac{\Delta t}{h} \leq \min\left(\frac{1}{2}, \frac{1}{C\|f\|_\infty}\right), \tag{21}$$

[illegible] u^n [illegible] v^n [illegible] (17)–(19) [illegible]:

(i) [illegible] u $0 \leq u^n \leq u^{n+1}$;

(ii) [illegible] v $v^n \geq 0$;

(iii) [illegible] u $|Du^n| \leq 1$

[illegible]. Let us remark that properties (i)–(iii) are trivially satisfied for $n = 0$, due to (19). Then, we prove the results by induction.

(i) This is a direct consequence of our assumptions and of (18).

(ii) From (17) we have

$$v_i^{n+1} = v_i^n + \Delta t G_i^n, \tag{22}$$

where

$$G_i^n \equiv v_i^n D^2 u_i^n + \overline{D} v_i^n D u_i^n - (1 - |D u_i^n|) v_i^n + f_i. \tag{23}$$

Since it can be proved that

$$G_i^n \geq -\frac{2}{h} v_i^n, \tag{24}$$

we can conclude from (21) that

$$v_i^{n+1} \geq \left(1 - \frac{2\Delta t}{h}\right) v_i^n \geq 0. \tag{25}$$

The proof of (24) is rather long since it must take into account all the possible cases with respect to the signs of first and second discrete derivatives. Let us just give here the proof for one of the cases, which leads to an upper bound for $\frac{\Delta t}{h}$. Assume

$D^2u_i < 0$, $Du_i = D^-u_i \geq D^+u_i > 0$, and $\overline{D}v_i = D^+v_i < 0$ (the index n has been dropped, since in this calculation it is fixed). By definition, we have

$$\begin{aligned} G_i &= v_i \frac{D^+u_i - Du_i}{h} + \frac{v_{i+1} - v_i}{h} Du_i - (1 - Du_i)v_i + f_i \\ &\geq -\frac{2}{h} v_i Du_i - (1 - Du_i)v_i = -v_i \left(1 + \left(\frac{2}{h} - 1\right) Du_i\right) \geq -\frac{2}{h} v_i. \end{aligned} \tag{26}$$

(iii) Let us start noting that, by (16), if $Du_i^n = D^+u_i^n$, then

$$|Du_i^n| \leq |Du_{i+1}^n|. \tag{27}$$

In fact, if $Du_{i+1} = D^-u_{i+1}$ and $Du_i = D^+u_i$, then $Du_{i+1} = Du_i$ (again n has been dropped). Otherwise, $Du_{i+1} = D^+u_{i+1}$, and we have $|D^+u_{i+1}| > |D^-u_{i+1}| = |Du_i|$. This also implies that if $|Du_i^n| \leq 1$ for any i, then

$$0 \leq (1 - |Du_{i+1}^n|) \leq (1 - |Du_i^n|). \tag{28}$$

We first prove property (iii) for $(n+1)$ under the additional assumption that for the index n

$$\frac{\Delta t}{h} v_i^n \leq 1 \quad \text{for any } i. \tag{29}$$

To simplify the presentation, we give the proof only for the case $0 < Du_i^n = D^+u_i^n < 1$ (the other cases can be treated in a similar way). Subtracting (18) for i from the same equation for $(i+1)$, we get

$$D^+u_i^{n+1} = D^+u_i^n + \Delta t \frac{(1 - |Du_{i+1}^n|)v_{i+1}^n - (1 - |Du_i^n|)v_i^n}{h}. \tag{30}$$

Then, (28) implies

$$D^+u_i^{n+1} \leq Du_i^n + \frac{\Delta t}{h}(1 - |Du_{i+1}^n|)v_{i+1}^n \leq Du_i^n + \frac{\Delta t}{h}(1 - |Du_i^n|)v_{i+1}^n. \tag{31}$$

From (29) we conclude

$$D^+u^{n+1} \leq Du_i^n + 1 - Du_i^n = 1. \tag{32}$$

In a similar way,

$$D^-u_i^{n+1} \leq D^-u_i^n + \frac{\Delta t}{h}(1 - |Du_i^n|)v_i^n \leq Du_i^n + \frac{\Delta t}{h}(1 - |Du_i^n|)v_i^n \leq 1, \tag{33}$$

so that we get $Du_i^{n+1} \leq 1$. Let us now prove that $Du_i^{n+1} \geq -1$. Again, by subtraction, assumption (29) implies

$$D^+u_i^{n+1} \geq -\frac{\Delta t}{h}(1 - |Du_i^n|)v_i^n \geq -\frac{\Delta t}{h} v_i^n \geq -1. \tag{34}$$

Moreover,

$$D^-u_i^{n+1} \geq D^-u_i^n - \frac{\Delta t}{h}(1 - |Du_{i-1}^n|)v_{i-1}^n \geq D^-u_i^n - (1 - |Du_{i-1}^n|), \tag{35}$$

and the lower bound follows if $(D^- u_i^n + |Du_{i-1}^n|) \geq 0$. Note that the previous inequality is always satisfied, since $Du_{i-1}^n = D^+ u_{i-1}^n$ implies $Du_{i-1}^n = D^- u_i^n$, and $Du_{i-1}^n = D^- u_{i-1}^n$ implies $|Du_{i-1}^n| \geq |D^+ u_{i-1}^n| = |Du_i^n|$.

Then, it should remain to prove only (29): here we do not give the rigorous proof, but we simply remark that it follows from (21) and from the natural bound

$$v_i^n \leq C\|f\|_\infty \quad \forall i,$$

which says that the amount of rolling sand in any point can be bounded from above by the source intensity. The stability condition (21) is therefore not very restrictive for small intensity source terms. □

By standard arguments one can show that the above scheme is consistent with the continuous system and that the local error is $O(\Delta x, \Delta t)$. We refer to [12] for a discussion of the stability of the scheme as well as for a proof of its convergence. Moreover, the stability conditions allow us to show that (u^n, v^n) converges, for n going to $+\infty$, to an equilibrium configuration $(\bar{u}, \bar{v})$ of the discrete system satisfying $(1 - |D\bar{u}|)\bar{v} = 0$.

The two-dimensional extension of the scheme is straightforward, and its main features are preserved. We introduce in the domain Ω (for simplicity, a rectangle) a uniform grid of nodes $x_{i,j}$, and, as usual, we denote by $(u_{i,j}^n, v_{i,j}^n)$ the components of the discrete solutions on node $x_{i,j}$ at time $t = t_n$. Then the scheme can be written as

$$v_{i,j}^{n+1} = v_{i,j}^n + \Delta t \left[v_{i,j}^n D^2 u_{i,j}^n + \overline{D} v_{i,j}^n \cdot Du_{i,j}^n - (1 - |Du_{i,j}^n|) v_{i,j}^n + f_{i,j} \right], \tag{36}$$

$$u_{i,j}^{n+1} = u_{i,j}^n + \Delta t (1 - |Du_{i,j}^n|) v_{i,j}^n, \tag{37}$$

$$u_{i,j}^0 = v_{i,j}^0 = 0 \quad \forall i,j, \quad u_{i,j}^n = 0 \quad \text{on } \partial\Omega \quad \forall n, \tag{38}$$

where now

$$D^2 u_{i,j} = \frac{u_{i,j-1} + u_{i,j+1} - 4u_{i,j} + u_{i-1,j} + u_{i+1,j}}{h^2}$$

is the standard five-point discretization of the Laplace operator, and the discrete gradient vectors $Du_{i,j}$ and $\overline{D} v_{i,j}$ are defined, component by component, as in the one-dimensional case. For example,

$$Du_{i,j} = (D_x u_{i,j}, D_y u_{i,j}),$$

where

$$D_x u_{i,j} = \text{maxmod}\left(\frac{u_{i,j} - u_{i-1,j}}{h}, \frac{u_{i+1,j} - u_{i,j}}{h} \right),$$

and $D_y u_{i,j}$ is defined in a similar way.

4. Numerical tests. All the tests refer to the Hadeler and Kuttler (HK) model. We have implemented the algorithm in one and two dimensions, testing different source terms, in cases (a) and (b).

In one dimension the ridge set $\mathcal{R}$ reduces to the middle point of the interval. Whenever the support of the source contains that point, we are in case (a), so that the final shape of the pile is given by the sum $(\overline{u} + \overline{v})$, where $\overline{u}$ is the distance function d and $\overline{v}$ is the rolling layer characterized by the explicit formula (10) (see Figure 1 (top)).

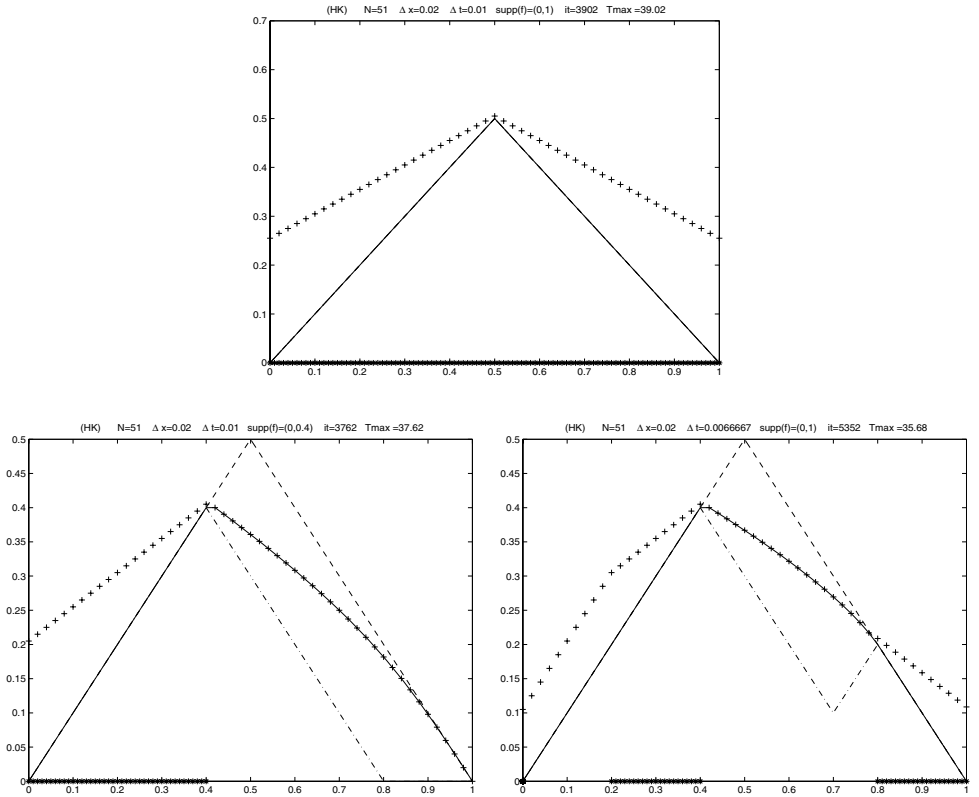


FIG. 1. *(HK-1D) $f \equiv 0.5$, plots of stationary u_h [—] and $(u_h + v_h)$ [++], compared with the distance function d [- -] and the minimal solution u^* [··-]: $D_f = [0, 1]$ (top), $D_f = [0, 0.4]$ (bottom left), $D_f = [0.2, 0.4] \cup [0.8, 1]$ (bottom right).*

In any other situation we fall into case (b), and the stable solution $\overline{u}$ in general does not coincide with the minimal solution (11) in all of Ω. More precisely, we get $u^* = \overline{u} = d$ in regions where the final rolling layer v is positive. In the complementary set the algorithm computes a larger function (see Figure 1 (bottom)). Such a function strongly depends on the source intensity, and a comparison principle holds:

$$0 \le f_1 \le f_2 \quad \text{in } D_f \quad \Rightarrow \quad u^* \le \overline{u}_1 \le \overline{u}_2 \le d \quad \text{in } \Omega.$$

This is clearly shown in Figure 2, where we plot the equilibria corresponding to different constant source terms sharing the same support D_f.

The growing process of the standing layer for different source terms is shown in Figure 3, where the pile profiles are plotted every 100 time iterations. It can be seen very well that the pile support expands before its slope becomes critical.

The stability bound (21) is confirmed by the experiments. In fact, Table 1 shows the following bound for the admissible Δt (asymptotically when $h \to 0$):

$$\Delta t \le C_f h,$$

which becomes more severe when the source intensity increases. As one can see looking at the first column ($f \equiv 1$) of Table 1, the ratio is always close to 0.5 and tends to 0.5 for N going to infinity. For bigger values of f (second and third columns) the

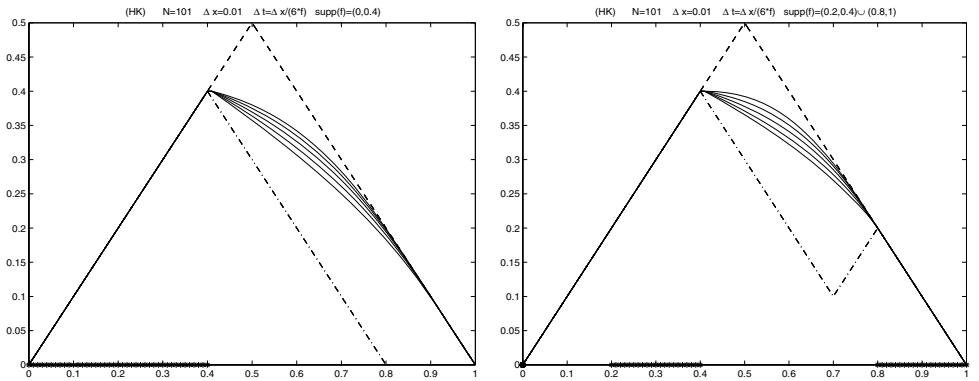


FIG. 2. *(HK-1D) $f \equiv 0.5/1/1.5/2/2.5$, plots of stationary u_h [—] compared with the distance function d [- -] and the minimal solution u^* [-·-]: $D_f = [0, 0.4]$ (left), $D_f = [0.2, 0.4] \cup [0.8, 1]$ (right).*

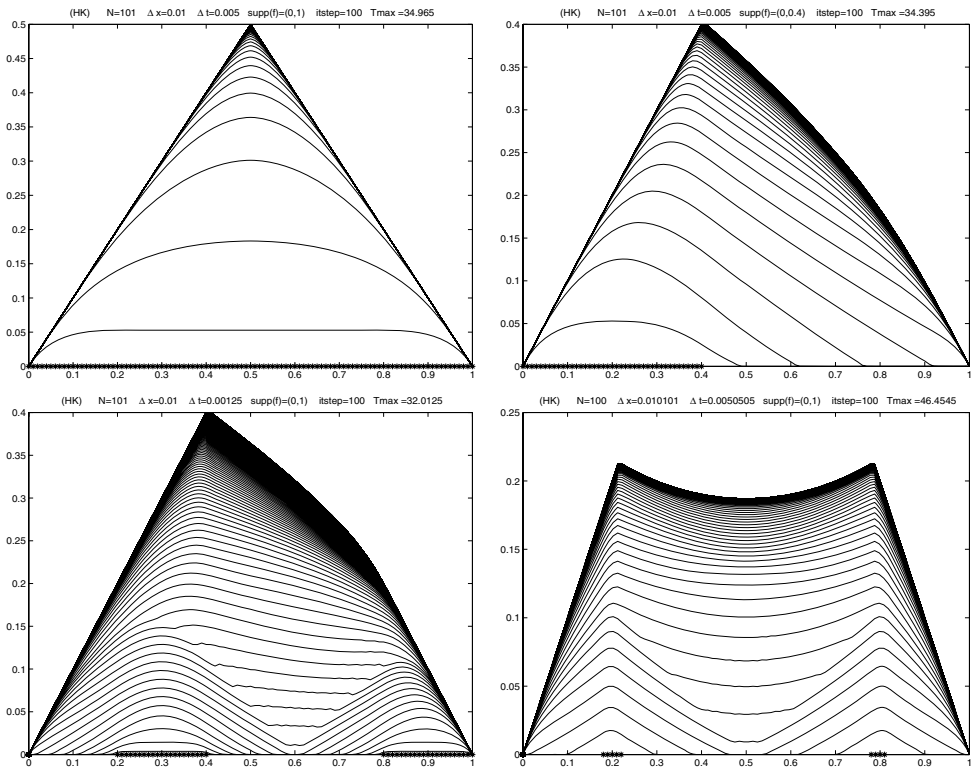


FIG. 3. *(HK-1D) $f \equiv 0.5$, plots of u_h every 100 time iterations: $D_f = [0, 1]$ (top left), $D_f = [0, 0.4]$ (top right), $D_f = [0.2, 0.4] \cup [0.8, 1]$ (bottom left), $D_f = [0.18, 0.22] \cup [0.78, 0.82]$ (bottom right).*

ratio which guarantees stability tends to a value strictly lower than 0.5. Moreover, as expected, the ratio $\Delta t/h$ always decreases along the rows of Table 1.

The two-dimensional experiments have been made in the case of a rectangular table. We used an $N \times M$ uniform grid of nodes $x_{i,j} = (ih, jh)$. Here the ridge set is no longer strictly contained in Ω, since it is the union of segments and reaches the

TABLE 1
Maximal ratio $\Delta t/h$ which yields stabilization in time of the discrete solutions when $D_f = [0,1]$ and f is constant.

N	$f \equiv 1$	$f \equiv 2$	$f \equiv 3$
201	0.6	0.52	0.46
401	0.552	0.476	0.424
601	0.528	0.45	0.41
801	0.512	0.44	0.392
1001	0.5	0.43	0.39

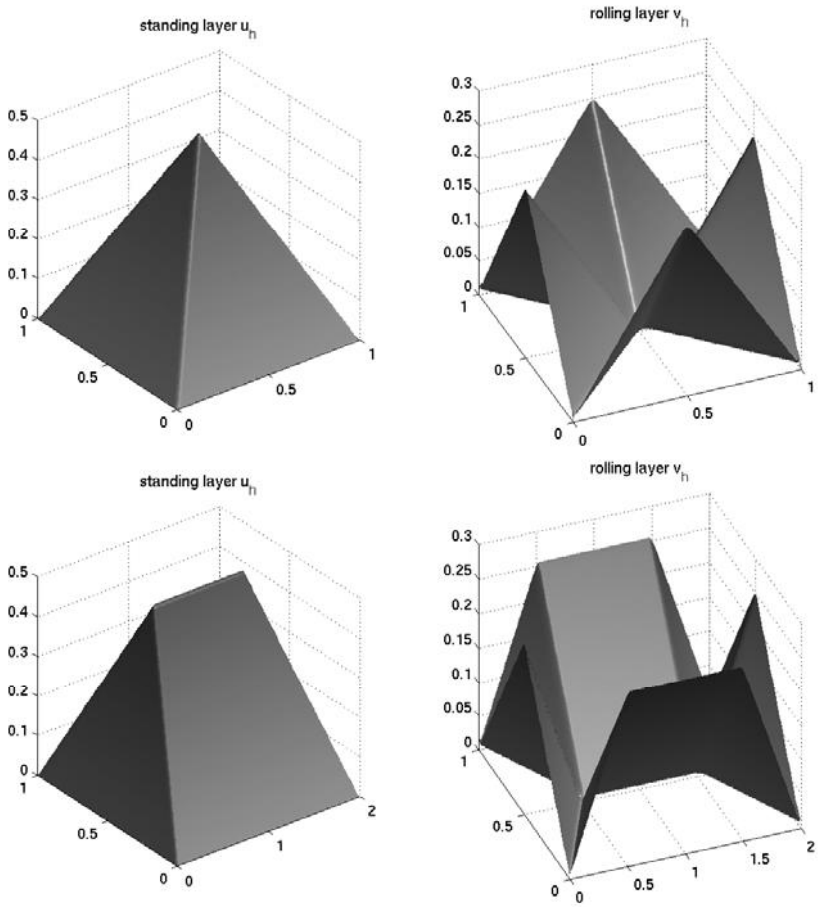


FIG. 4. *(HK-2D) $D_f = \Omega$, $f \equiv 0.5$: $\Omega = (0,1) \times (0,1)$, $N = M = 101$ (top); $\Omega = (0,1) \times (0,2)$, $N = 101$, $M = 201$ (bottom).*

boundary at the four vertices.

We are in case (a) if, for example, the source term is positive (e.g., constant) everywhere in Ω. In Figure 4 we see the stable configurations u and v for the square and the rectangular tables.

All the simulations of the growth process of sandpiles with source support in case

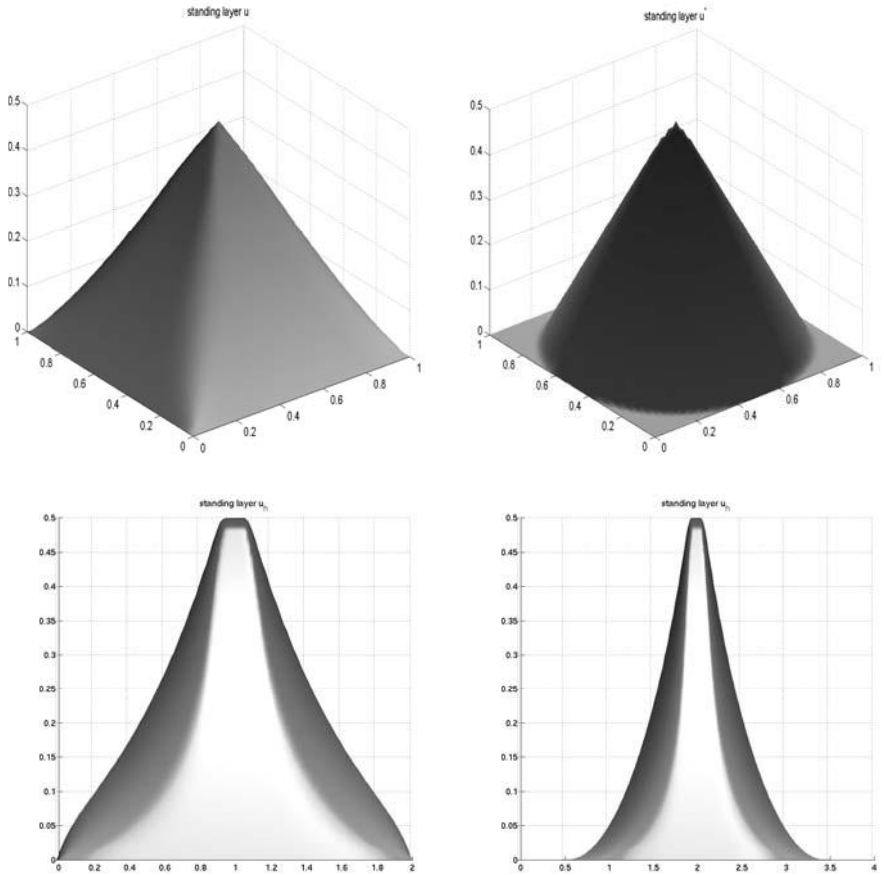


FIG. 5. *(HK-2D)* u_h *versus* u^* *for* $\Omega = (0,1) \times (0,1)$, $D_f = [0.45, 0.55]^2$, $f \equiv 0.5$, $N = M = 51$ *(top); asymptotic pile profiles when* $\Omega = (0,1) \times (0,2)$ *and* $\Omega = (0,1) \times (0,4)$ *(bottom).*

(b) show approximate equilibria that are always larger than the minimal ones. We are in case (b) if, for example, the source support is a small square at the center of the table. Then the stable equilibrium (11) has an almost conical shape, and sand is leaving the table only at the middle of the edges. The solution reached by the double-layer scheme instead fills the entire square table (Figure 5 (top)). We have to consider a much longer rectangular table in order to see also in this case a compact support solution. In Figure 5 (bottom) we see the different contact angles of the pile with the table when it touches the boundary and when it stays away from the boundary. Finally, we show a more general example, with a nonconstant source term $f(x,y) = \max\{0, \sin(2\pi x) + \cos(2\pi y) - \sin(2\pi xy)\}$ (its support is strictly contained in $\overline{\Omega}$). Note that the minimal stationary solution (11) in this case would be the union of cones of Figure 6 (top right), whereas the HK model evolves towards a relaxed equilibrium where sand partially fills the valleys among the heaps (top left). On the contrary, the corresponding rolling layers for the two equilibria are essentially the same (see Figure 6 (bottom)), in accordance with the partial uniqueness result of Theorem 2.1. As a final remark, it is interesting to note that the above experiments are in accordance with some real experiments reported in the literature; see, e.g., [1, 5, 15, 13, 17].

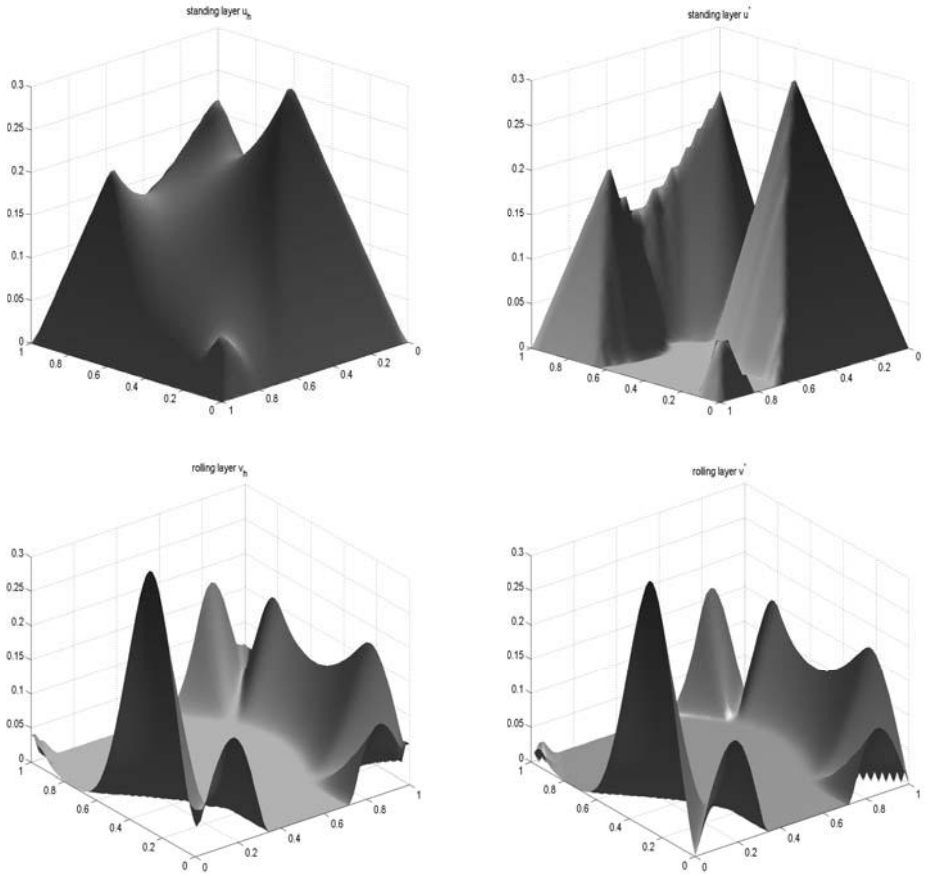


FIG. 6. *(HK-2D)* $\Omega = (0,1)\times(0,1)$, $D_f \subset \Omega$, $f(x,y) = \max\{0, \sin(2\pi x)+\cos(2\pi y)-\sin(2\pi xy)\}$, $N = M = 51$*:* u_h *versus* u^* *(top);* v_h *versus* v^* *(bottom).*

REFERENCES

[1] G. Aronsson, L. C. Evans, and Y. Wu, *Fast/slow diffusion and growing sandpiles*, J. Differential Equations, 131 (1996), pp. 304–335.

[2] P. Bak, C. Tang, and K. Wiesenfeld, *Self-organized criticality*, Phys. Rev. A, 38 (1988), pp. 364–374.

[3] D. Benedetto, E. Caglioti, and M. Pulvirenti, *A kinetic equation for granular media*, RAIRO Modél. Math. Anal. Numér., 31 (1997), pp. 615–641.

[4] D. Benedetto, E. Caglioti, and M. Pulvirenti, *Collective behavior of one-dimensional granular media*, in Modeling in Applied Sciences, A Kinetic Theory Approach, Birkhäuser Boston, Cambridge, MA, 2000, pp. 81–110.

[5] R. P. Behringer, *The dynamics of flowing sand*, Nonlinear Sci. Today, 3 (1993), pp. 1–15.

[6] J.-P. Bouchaud, M. E. Cates, J. Ravi Prakash, and S. F. Edwards, *A model for the dynamics of sandpile surfaces*, J. Phys. I France, 4 (1994), pp. 1383–1410.

[7] J.-P. Bouchaud, M. E. Cates, J. Ravi Prakash, and S. F. Edwards, *Hysteresis and metastability in a continuum sandpile model*, Phys. Rev. Lett., 74 (1995), pp. 1982–1985.

[8] P. Cannarsa and P. Cardaliaguet, *Representation of equilibrium solutions to the table problem for growing sandpiles*, J. Eur. Math. Soc., 6 (2004), pp. 435–464.

[9] D. Dhar, *Self-organized critical state of sandpile automation models*, Phys. Rev. Letters, 64 (1990), pp. 1613–1616.

[10] P. G. de Gennes, *Granular matter*, Summer School on Complex Systems (Varenna), Lecture Notes Societá Italiana di Fisica, 1996, pp. 21–51.

[11] L. C. Evans, M. Feldman, and R. F. Gariepy, *Fast/slow diffusion and collapsing sandpiles*, J. Differential Equations, 137 (1997), pp. 166–209.
[12] M. Falcone and S. Finzi Vita, *On the Convergence of a Finite-Difference Scheme for Growing Sandpiles*, in preparation.
[13] K. P. Hadeler and C. Kuttler, *Dynamical models for granular matter*, Granular Matter, 2 (1999), pp. 9–18.
[14] L. Prigozhin, *Variational model of sandpile growth*, European J. Appl. Math., 7 (1996), pp. 225–235.
[15] L. Prigozhin and B. Zaltzman, *Two continuous models for the dynamics of sandpile surfaces*, Phys. Rev. E, 63 (2001), paper 041505.
[16] L. Prigozhin and B. Zaltzman, *On the approximation of the dynamics of sandpile surfaces*, Port. Math. (N.S.), 60 (2003), pp. 127–137.
[17] H. Puhl, *On the modelling of real sand piles*, Phys. A, 182 (1992), pp. 295–308.

SIAM J. SCI. COMPUT.
Vol. 28, No. 3, pp. 1133–1152
© 2006 Society for Industrial and Applied Mathematics

A NEUMANN–DIRICHLET PRECONDITIONER FOR A FETI-DP FORMULATION OF THE TWO-DIMENSIONAL STOKES PROBLEM WITH MORTAR METHODS*

HYEA HYUN KIM† AND CHANG-OCK LEE†

Abstract. A FETI-DP (dual-primal finite element tearing and interconnecting) formulation for the two-dimensional Stokes problem with mortar methods is considered. Separate sets of unknowns are used for velocity on interfaces, and the mortar constraints are enforced on the velocity unknowns by Lagrange multipliers. Average constraints on edges are further introduced as primal constraints to solve the Stokes problem correctly and to obtain a scalable FETI-DP algorithm. A Neumann–Dirichlet preconditioner is shown to give a condition number bound, $C \max_{i=1,\ldots,N}\{(1+\log(H_i/h_i))^2\}$, where H_i and h_i are the subdomain size and the mesh size, respectively, and the constant C is independent of the mesh parameters H_i and h_i.

Key words. FETI-DP, mortar methods, Neumann–Dirichlet preconditioner, Stokes problem

AMS subject classifications. 65N30, 65N55, 76D07

DOI. 10.1137/030601119

1. Introduction.

In this paper, an iterative substructuring method with Lagrange multipliers is studied for the two-dimensional Stokes problem with nonconforming discretizations. Nonconforming discretizations are important for multiphysics simulations, contact-impact problems, the generation of meshes and partitions aligned with jumps in diffusion coefficients, *hp*-adaptive methods, and special discretizations in the neighborhood of singularities. Of the many methods for nonconforming discretizations, including Galerkin methods [7], we consider the mortar methods [1, 3, 20, 21] originally introduced by Bernardi, Maday, and Patera.

Dual-primal FETI (FETI-DP) methods were introduced in [11] as a generalization of the FETI (finite element tearing and interconnecting) method [12]. Continuity of solutions at subdomain corners is enforced by primal variables to improve the convergence as well as to make local problems nonsingular. Later Mandel and Tezaur [17] proved that these algorithms give the condition number bound, $C\,(1+\log(H/h))^2$, for both second- and fourth-order elliptic problems in two dimensions. Here, H and h denote the subdomain size and mesh size, respectively. For three-dimensional elliptic problems with heterogeneous coefficients, Klawonn, Widlund, and Dryja [14] introduced averages over individual edges and/or faces as primal constraints to obtain a method as scalable as in two dimensions. Later FETI-DP algorithms were extended to the Stokes problem in both two and three dimensions by Li [15, 16]. Edge and/or face average constraints as well as vertex constraints are selected as a set of primal constraints to enhance the convergence of the FETI-DP algorithms.

Recently, FETI-DP methods have been applied to nonconforming discretizations [8, 9, 10, 13, 18]. For elliptic problems in two dimensions, Dryja and Widlund [9, 10] proposed several preconditioners and showed the condition number bound $C(1+\log(H/h))^2$. However, the constant C depends on the ratio of mesh sizes across

*Received by the editors October 6, 2003; accepted for publication (in revised form) February 2, 2006; published electronically July 17, 2006. This work was partially supported by KRF-2002-070-C00004 and KOSEF R01-2000-00008.

http://www.siam.org/journals/sisc/28-3/60111.html

†Division of Applied Mathematics, KAIST, Daejeon 305-701, Korea (mashy@amath.kaist.ac.kr, colee@amath.kaist.ac.kr).

interfaces. Later, the current authors [13] proposed a Neumann–Dirichlet preconditioner, which gives the same condition number bound without the dependence of C upon the ratio of mesh sizes. The proposed preconditioner is similar to the previous FETI-DP preconditioners except that it solves local problems with Neumann boundary conditions on nonmortar interfaces and with a zero Dirichlet boundary condition on mortar interfaces.

The purpose of this paper is to extend the FETI-DP algorithm developed in [13] to the two-dimensional Stokes problem with mortar discretizations. The inf-sup stable $P_1(h)$-$P_0(2h)$ finite element space is considered in each subdomain. The triangulations are nonmatching across subdomain interfaces. To achieve the optimal approximation, we impose mortar matching conditions on the velocity functions. An optimal approximation of mortar methods for the Stokes problem was proved by Belgacem [2]. The inf-sup constant of the mortar finite element space is important in the analysis. If the constant is independent of mesh size and subdomain size, then the optimal order of approximation follows independent of the number of subdomains and mesh size, as in the case of elliptic problems. In [2], it was shown that the inf-sup constant is independent of mesh size for the Hood–Taylor finite element space, but this was not shown for the subdomain size. For the $P_1(h)$-$P_0(2h)$ mortar finite element space, we compute the inf-sup constant numerically by increasing the number of subdomains and decreasing mesh sizes, and we observe that the constant seems to be independent of these parameters.

We follow the FETI-DP formulation developed in [15]. The fundamental idea of the present paper is the same as one of [15], with additional technical complications caused by nonmatching grids across subdomain interfaces. Mortar matching constraints will be enforced on velocity unknowns across interfaces instead of pointwise matching constraints in conforming discretization. We introduce the primal constraints, i.e., edge average and vertex constraints, to solve the Stokes problem efficiently and correctly. We then propose a Neumann–Dirichlet preconditioner and analyze the condition number bound. The preconditioner consists of local Stokes problems with Neumann boundary conditions on nonmortar edges and a zero Dirichlet boundary condition on the remaining part of the subdomain boundary. The additional complication caused by mortar discretizations can be handled by using this preconditioner. In the analysis, the stability of the mortar projection in the $H_{00}^{1/2}$-norm is used. Our theory can be extended to Lagrange multiplier spaces with this property. Several such Lagrange multiplier spaces have been developed by Wohlmuth [20, 21].

This paper is organized as follows. Section 2 contains a brief introduction to Sobolev spaces and finite elements. In section 3, we derive a FETI-DP formulation of the Stokes problem. Section 4 is devoted to analyzing the condition number bound. Numerical results are included in section 5. Throughout this paper, C denotes a generic constant independent of mesh sizes and subdomain sizes. We will use H_i and h_i to denote the subdomain size and the typical mesh size of each subdomain Ω_i, respectively.

2. Sobolev spaces and finite elements.

2.1. A model problem. Let Ω be a bounded polygonal domain in $\mathbb{R}^2$, and let $L^2(\Omega)$ be the space of square integrable functions defined in Ω equipped with the norm

$$\|v\|_{0,\Omega}^2 := \int_\Omega v^2 \, dx.$$

The space $L^2_0(\Omega)$ contains functions in $L^2(\Omega)$ with zero average $\int_\Omega v\,dx = 0$. The space $H^1(\Omega)$ consists of functions that are square integrable up to the first weak derivatives with the norm

$$\|v\|^2_{1,\Omega} := \int \nabla v \cdot \nabla v\,dx + \int v^2\,dx.$$

The space $H^1_0(\Omega)$ is a subspace of $H^1(\Omega)$ with functions having zero trace on the boundary of Ω.

In this paper, we consider the following Stokes problem: For $\mathbf{f} \in [L^2(\Omega)]^2$, find $(\mathbf{u}, p) \in [H^1_0(\Omega)]^2 \times L^2_0(\Omega)$ satisfying

$$\begin{aligned} -\triangle \mathbf{u} + \nabla p &= \mathbf{f} \quad \text{in } \Omega, \\ -\nabla \cdot \mathbf{u} &= 0 \quad \text{in } \Omega, \\ \mathbf{u} &= \mathbf{0} \quad \text{on } \partial\Omega. \end{aligned} \tag{2.1}$$

We then partition Ω into nonoverlapping subdomains $\{\Omega_i\}_{i=1}^N$, which are geometrically conforming; a subdomain intersects its neighbors on a whole edge or at a vertex. For each subdomain, we introduce the space $H^1_D(\Omega_i)$ to be a subspace of $H^1(\Omega_i)$ with zero trace on $\partial\Omega_i \cap \partial\Omega$, the space $L^2_0(\Omega_i)$ to be a subspace of $L^2(\Omega_i)$ with zero average, and the space Π_0, which consists of functions that are constant in each subdomain and have zero average in Ω:

$$\begin{aligned} H^1_D(\Omega_i) &:= \left\{ v \in H^1(\Omega_i) \ : \ v = 0 \text{ on } \partial\Omega_i \cap \partial\Omega \right\}, \\ L^2_0(\Omega_i) &:= \left\{ q \in L^2(\Omega_i) \ : \ \int_{\Omega_i} q\,dx = 0 \right\}, \\ \Pi_0 &:= \left\{ q_0 \ : \ q_0|_{\Omega_i} \text{ is constant and } \int_\Omega q_0\,dx = 0 \right\}. \end{aligned} \tag{2.2}$$

The problem (2.1) is then written into an equivalent variational form: Find $(\mathbf{u}, p_I, p_0) \in \prod_{i=1}^N \left[H^1_D(\Omega_i)\right]^2 \times \prod_{i=1}^N L^2_0(\Omega_i) \times \Pi_0$ such that

$$\begin{aligned} \sum_{i=1}^N (\nabla \mathbf{u}, \nabla \mathbf{v})_{\Omega_i} - \sum_{i=1}^N (p_I + p_0, \nabla \cdot \mathbf{v})_{\Omega_i} &= \sum_{i=1}^N (\mathbf{f}, \mathbf{v})_{\Omega_i} \quad \forall\, \mathbf{v} \in \prod_{i=1}^N \left[H^1_D(\Omega_i)\right]^2, \\ -\sum_{i=1}^N (\nabla \cdot \mathbf{u}, q_I)_{\Omega_i} &= 0 \quad \forall\, q_I \in \prod_{i=1}^N L^2_0(\Omega_i), \\ -\sum_{i=1}^N (\nabla \cdot \mathbf{u}, q_0)_{\Omega_i} &= 0 \quad \forall\, q_0 \in \Pi_0, \\ \mathbf{u}|_{\Omega_i} - \mathbf{u}|_{\Omega_j} &= 0 \quad \forall\, \Gamma_{ij} = \partial\Omega_i \cap \partial\Omega_j. \end{aligned} \tag{2.3}$$

Here $(\cdot,\cdot)_{\Omega_i}$ denotes the inner product in $[L^2(\Omega_i)]^n$ for $n = 1, 2$. We now introduce a finite element space to approximate the above problem. Triangulations $\Omega_i^{2h_i}$ and $\Omega_i^{h_i}$ for pressure and velocity, respectively, are given in each subdomain. The finer triangulation $\Omega_i^{h_i}$ is obtained from $\Omega_i^{2h_i}$ by connecting midpoints of edges in the triangle $\tau \in \Omega_i^{2h_i}$ so that τ is divided into four triangles. We assume that these triangulations are quasi-uniform and may not match across subdomain interfaces. The finite element space $P_1(h_i) - P_0(2h_i)$ is then associated with each subdomain Ω_i;

we denote by X_i the space of conforming linear finite elements on the triangulation $\Omega_i^{h_i}$ and by Q_i the space of functions constant on each triangles in $\Omega_i^{2h_i}$ with zero average over Ω_i:

$$X_i := \left\{ \mathbf{v}_i \in \left[H_D^1(\Omega_i) \cap C(\Omega_i) \right]^2 \ : \ \mathbf{v}_i|_\tau \text{ is piecewise linear } \forall\, \tau \in \Omega_i^{h_i} \right\},$$
$$Q_i := \left\{ q_i \in L_0^2(\Omega_i) \ : \ q_i|_\tau \text{ is constant } \forall\, \tau \in \Omega_i^{2h_i} \right\}.$$

The inf-sup stability of the space $P_1(h_i) - P_0(2h_i)$ can be shown from the macro element technique in [19] or from the inf-sup stability of the space $P_2(2h_i) - P_0(2h_i)$ in [6]. The space $P_2(2h_i) - P_0(2h_i)$ consists of piecewise quadratic functions for velocity and piecewise constant functions for pressure in the same triangulation $\Omega_i^{2h_i}$. In the proof of the stability of $P_1(h_i) - P_0(2h_i)$, we may regard $P_2(2h_i) - P_0(2h_i)$ as identical to $P_1(h_i) - P_0(2h_i)$.

Our FETI-DP formulation will be described using the space X, an approximate space for velocity, which can be discontinuous across the interfaces except corners; the space Q_I for pressure, which has zero average in each subdomain; and the space W for velocity on the interfaces, which is continuous at corners and can be discontinuous on the remaining part:

$$\begin{aligned} X &:= \left\{ \mathbf{v} \in \prod_{i=1}^N X_i \ : \ \mathbf{v} \text{ is continuous at subdomain corners} \right\}, \\ Q_I &:= \prod_{i=1}^N Q_i, \\ W_i &:= X_i|_{\partial\Omega_i} \quad \text{for } i = 1, \dots, N, \\ W &:= \left\{ \mathbf{w} \in \prod_{i=1}^N W_i \ : \ \mathbf{w} \text{ is continuous at subdomain corners} \right\}. \end{aligned} \tag{2.4}$$

Throughout this paper, we will use the same notation for a finite element function and the nodal unknowns of the function. For example, $\mathbf{v}_i$ can be used to denote a finite element function or the corresponding nodal unknowns. The same holds for the notations W_i, X, W, etc.

We now introduce Sobolev spaces on the subdomain boundaries. The space $H^{1/2}(\partial\Omega_i)$ is the trace space of $H^1(\Omega_i)$ normed by

$$\|w_i\|_{1/2,\partial\Omega_i}^2 := |w_i|_{1/2,\partial\Omega_i}^2 + \frac{1}{H_i}\|w_i\|_{0,\partial\Omega_i}^2,$$

where

$$|w_i|_{1/2,\partial\Omega_i}^2 := \int_{\partial\Omega_i}\int_{\partial\Omega_i} \frac{|w_i(x) - w_i(y)|^2}{|x-y|^2}\, ds(x)\, ds(y).$$

For any $\Gamma_{ij} \subset \partial\Omega_i$, the space $H_{00}^{1/2}(\Gamma_{ij})$ is a set of functions in $L^2(\Gamma_{ij})$ of which zero extension to $\partial\Omega_i$ is contained in $H^{1/2}(\partial\Omega_i)$, and it is equipped with the norm

$$\|v\|_{H_{00}^{1/2}(\Gamma_{ij})}^2 := |v|_{H^{1/2}(\Gamma_{ij})}^2 + \int_{\Gamma_{ij}} \frac{v(x)^2}{\operatorname{dist}(x, \partial\Gamma_{ij})}\, ds.$$

From section 4.1 in [22], the following relation holds for $v \in H_{00}^{1/2}(\Gamma_{ij})$:

$$C_1 \|\widetilde{v}\|_{1/2,\partial\Omega_i} \le \|v\|_{H_{00}^{1/2}(\Gamma_{ij})} \le C_2 \|\widetilde{v}\|_{1/2,\partial\Omega_i}, \tag{2.5}$$

where $\widetilde{v}$ is the zero extension of v to $\partial\Omega_i$. The inequalities in (2.5) also hold for the product spaces $[H^{1/2}(\partial\Omega_i)]^2$ and $[H_{00}^{1/2}(\Gamma_{ij})]^2$ equipped with product norms.

2.2. Mortar methods. We consider the space X for velocity and the space $P = Q_I \times \Pi_0$ for pressure to approximate the Stokes problem (2.3); see (2.4) and (2.2) for the definitions of X, Q_I, and Π_0. We will impose the mortar matching condition on the velocity functions. On the interface $\Gamma_{ij} = \partial\Omega_i \cap \partial\Omega_j$, two different triangulations are given by the triangulations in Ω_i and Ω_j. We distinguish them by choosing one as a mortar side and the other as a nonmortar side. On each subdomain Ω_i, we define

$$m_i := \{j \ : \ \Omega_j \text{ is the mortar side of } \Gamma_{ij} \quad \forall\, \Gamma_{ij} \subset \partial\Omega_i\},$$

which is a set of subdomain indices j such that Ω_j intersects Ω_i on the interface Γ_{ij}, where Ω_j is chosen as the mortar side and Ω_i is chosen as the nonmortar side.

On each interface Γ_{ij}, we consider a trace space

$$X_{n(ij)}|_{\Gamma_{ij}}, \tag{2.6}$$

where $X_{n(ij)}$ is the finite element space on the nonmortar side of Γ_{ij}; i.e., $n(ij) = i$ if Ω_i is the nonmortar side of Γ_{ij}. The standard Lagrange multiplier space introduced in [3] is then given by

$$M_{ij} := \{\boldsymbol{\psi} \in X_{n(ij)}|_{\Gamma_{ij}} \ : \ \boldsymbol{\psi}|_\tau \text{ is constant on } \tau \text{ which intersects } \partial\Gamma_{ij}\}.$$

Here τ is the restriction of a triangle in the triangulation $\Omega_{n(ij)}^{h_{n(ij)}}$ to the interface Γ_{ij}. We then take the Lagrange multiplier space

$$M := \prod_{i=1}^{N} \prod_{j \in m_i} M_{ij}. \tag{2.7}$$

The mortar matching condition on $\mathbf{v} = (\mathbf{v}_1, \ldots, \mathbf{v}_N) \in X$ is

$$\int_{\Gamma_{ij}} (\mathbf{v}_i - \mathbf{v}_j) \cdot \boldsymbol{\lambda}_{ij}\, ds = 0 \quad \forall \boldsymbol{\lambda}_{ij} \in M_{ij},\ \forall\, i = 1, \ldots, N,\ \forall j \in m_i. \tag{2.8}$$

The mortar finite element space of velocity is then given by

$$V := \{\mathbf{v} \in X \ : \ \mathbf{v} \text{ satisfies the mortar matching condition (2.8)}\}. \tag{2.9}$$

It was shown in [2] that the best approximation property holds for the space $V \times P$, given by the Hood–Taylor finite element space for each subdomain, if it satisfies the inf-sup condition

$$\inf_{q \in P} \sup_{\mathbf{v} \in V} \frac{\int_\Omega \nabla \cdot \mathbf{v}\, q\, dx}{\|v\|_{1,\Omega}^* \|q\|_{0,\Omega}} \ge \beta,$$

where the constant β is independent of the mesh size and the subdomain size. Here $\|v\|^*_{1,\Omega}$ denotes the broken H^1-norm

$$\|v\|^*_{1,\Omega} = \left(\sum_{i=1}^{N} \|v\|^2_{1,\Omega_i} \right)^{1/2}.$$

To our best knowledge, there is no mathematical proof that shows that the constant β is independent of the subdomain size. We compute the inf-sup constant numerically and observe that the constant is independent of the subdomain size as well as the mesh size; see section 5. Therefore we can get a solution as accurate as that in conforming approximations.

3. FETI-DP formulation.

3.1. Notation. We will introduce several notations which will be used in the FETI-DP formulation. We recall X_i, the velocity finite element space in Ω_i; W_i, the trace space of X_i on $\partial\Omega_i$; and X, a subspace of the product space of X_i with functions that are continuous at subdomain corners. The space W is similarly defined as a subspace of the product space of W_i. The definitions of these spaces are given in (2.4).

The unknowns $\mathbf{v}_i \in X_i$ and $\mathbf{w}_i \in W_i$ are ordered into

$$\mathbf{v}_i = \begin{pmatrix} \mathbf{v}_I^i \\ \mathbf{v}_r^i \\ \mathbf{v}_c^i \end{pmatrix}, \quad \mathbf{w}_i = \begin{pmatrix} \mathbf{w}_r^i \\ \mathbf{w}_c^i \end{pmatrix},$$

where the subscripts I, r, and c represent the d.o.f. (degrees of freedom) corresponding to the interior, edges, and corners, respectively. We define the following spaces based on the splitting of unknowns: X_I, the space of velocity unknowns at the interior of each subdomain; W_r, the space of velocity unknowns at the interior of both mortar and nonmortar edges; W_c, the space of velocity unknowns at global corners; and W_c^i, the space of velocity unknowns at subdomain corners:

$$\begin{aligned} X_I &= \left\{ \mathbf{v}_I \ : \ \mathbf{v}_I|_{\Omega_i} = \mathbf{v}_I^i, \quad \forall i = 1, \ldots, N \right\}, \\ W_r &= \left\{ \mathbf{w}_r \ : \ \mathbf{w}_r|_{\partial\Omega_i} = \mathbf{w}_r^i, \quad \forall i = 1, \ldots, N \right\}, \\ W_c &= \left\{ \mathbf{w}_c \ : \ \mathbf{w}_c \text{ are velocity unknowns at global corners} \right\}, \\ W_c^i &= \left\{ \mathbf{w}_c^i \ : \ \mathbf{w}_c^i \text{ are velocity unknowns at corners in } \Omega_i \right\}. \end{aligned} \tag{3.1}$$

We further define the restriction map

$$L_c^i \ : \ W_c \to W_c^i$$

that gives

$$L_c^i \mathbf{w}_c = \mathbf{w}_c^i \quad \forall i = 1, \ldots, N, \text{ for } \mathbf{w}_c \in W_c.$$

We will now express the mortar matching condition (2.8) as a matrix form

$$B\mathbf{w} = \mathbf{0}, \tag{3.2}$$

where

$$\begin{aligned} B &= \begin{pmatrix} B_1 & \cdots & B_N \end{pmatrix}, \\ \mathbf{w} &= \begin{pmatrix} \mathbf{w}_1^t & \cdots & \mathbf{w}_N^t \end{pmatrix}^t, \qquad \mathbf{w}_i = \mathbf{v}_i|_{\partial\Omega_i}. \end{aligned}$$

Since $\mathbf{w}$ is continuous at global corners, the above equation can be written as a different form with unknowns $\mathbf{w}_r \in W_r$ and $\mathbf{w}_c \in W_c$:

$$B_r \mathbf{w}_r + B_c \mathbf{w}_c = \mathbf{0}, \tag{3.3}$$

where B_r and B_c are assembled by local matrices $B_{i,r}$ and $B_{i,c}$,

$$B_r = \begin{pmatrix} B_{1,r} & \cdots & B_{N,r} \end{pmatrix}, \qquad B_c = \sum_{i=1}^{N} B_{i,c} L_c^i.$$

These matrices $B_{i,r}$ and $B_{i,c}$ consist of the columns of B_i corresponding to the unknowns on edges and corners, respectively. We will use any of these two expressions (3.2) and (3.3) from place to place for the sake of convenience.

3.2. FETI-DP formulation with primal constraints. In this section, we formulate a FETI-DP operator with the mortar constraints (3.3). To solve the Stokes problem efficiently, we will consider the following primal constraints:

$$\int_{\Gamma_{ij}} (\mathbf{v}_i - \mathbf{v}_j)\, ds = \mathbf{0} \qquad \forall i = 1, \ldots, N,\ \forall j \in m_i. \tag{3.4}$$

Note that (3.4) holds by replacing $\boldsymbol{\lambda}_{ij} = \binom{1}{0}$ or $\binom{0}{1}$ in (2.8), because constant multipliers belong to the Lagrange multiplier space M_{ij}. These constraints are average matching conditions of each velocity component on subdomain interfaces. These primal constraints were introduced by Li [15, 16] to the Stokes problem with conforming discretization. The primal constraints enlarge the size of the coarse problem so that it may lead to a fast convergence of the FETI-DP iteration.

We write (3.4) as

$$R^t (B_r \mathbf{w}_r + B_c \mathbf{w}_c) = \mathbf{0}, \tag{3.5}$$

where the matrix R is a Boolean matrix that has the number of columns equal to twice the number of interfaces Γ_{ij}, and the number of rows equal to the d.o.f. of the space M. For $\boldsymbol{\lambda} \in M$, at each interior nodal point of Γ_{ij}, $\boldsymbol{\lambda}|_{\Gamma_{ij}}$ has two components corresponding to horizontal and vertical parts of velocity function, and $R^t \boldsymbol{\lambda} = \mathbf{0}$ means that sums of each $\boldsymbol{\lambda}|_{\Gamma_{ij}}$ corresponding to the horizontal and vertical parts of velocity function are zero.

Let U be the Lagrange multiplier space corresponding to the constraints (3.5); for $\boldsymbol{\mu} \in U$, $\boldsymbol{\mu}|_{\Gamma_{ij}}$ has two components that correspond to the constraints for each horizontal velocity and vertical velocity on the interface Γ_{ij}. By introducing Lagrange multipliers $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$ to enforce the constraints (3.3) and (3.5), we have the following mixed formulation of the problem (2.3): Find $(\mathbf{u}_I, p_I, \mathbf{u}_r, \mathbf{u}_c, p_0, \boldsymbol{\mu}, \boldsymbol{\lambda}) \in X_I \times Q_I \times W_r \times W_c \times \Pi_0 \times U \times M$ such that

$$\begin{pmatrix} A_{II} & G_{II} & A_{Ir} & A_{Ic} & G_{I0} & \mathbf{0} & \mathbf{0} \\ G_{II}^t & \mathbf{0} & G_{rI}^t & G_{cI}^t & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ A_{rI} & G_{rI} & A_{rr} & A_{rc} & G_{r0} & B_r^t R & B_r^t \\ A_{cI} & G_{cI} & A_{cr} & A_{cc} & G_{c0} & B_c^t R & B_c^t \\ G_{I0}^t & \mathbf{0} & G_{r0}^t & G_{c0}^t & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & R^t B_r & R^t B_c & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & B_r & B_c & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{u}_I \\ p_I \\ \mathbf{u}_r \\ \mathbf{u}_c \\ p_0 \\ \boldsymbol{\mu} \\ \boldsymbol{\lambda} \end{pmatrix} = \begin{pmatrix} \mathbf{f}_I \\ \mathbf{0} \\ \mathbf{f}_r \\ \mathbf{f}_c \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{pmatrix}. \tag{3.6}$$

Here

$$\begin{pmatrix} A_{II} & A_{Ir} & A_{Ic} \\ A_{rI} & A_{rr} & A_{rc} \\ A_{cI} & A_{cr} & A_{cc} \end{pmatrix} \quad \text{is a stiffness matrix given by} \quad \sum_{i=1}^{N} (\nabla \mathbf{u}, \nabla \mathbf{v})_{\Omega_i},$$

$$\begin{pmatrix} G_{II}^t & G_{rI}^t & G_{cI}^t \end{pmatrix} \quad \text{is a matrix given by} \quad \sum_{i=1}^{N} (-\nabla \cdot \mathbf{v}, p_I)_{\Omega_i},$$

$$\begin{pmatrix} G_{I0}^t & G_{r0}^t & G_{c0}^t \end{pmatrix} \quad \text{is a matrix given by} \quad \sum_{i=1}^{N} (-\nabla \cdot \mathbf{v}, p_0)_{\Omega_i}.$$

The velocity spaces X_I, W_r, and W_c are defined in (3.1). For the definitions of spaces Q_I, Π_0, and M, see (2.4), (2.2), and (2.7), respectively. Since $p_0|_{\Omega_i}$ is constant and $\mathbf{v}_I|_{\partial\Omega_i} = \mathbf{0}$ for $\mathbf{v}_I \in X_I$, the divergence theorem gives $G_{I0} = \mathbf{0}$.

Let

$$\mathbf{z}_r = \begin{pmatrix} \mathbf{u}_I \\ p_I \\ \mathbf{u}_r \end{pmatrix}, \quad \mathbf{z}_c = \begin{pmatrix} \mathbf{u}_c \\ p_0 \\ \boldsymbol{\mu} \end{pmatrix}.$$

We regard $\mathbf{z}_c$ as a primal variable in the FETI-DP formulation and then write (3.6) into

$$\begin{pmatrix} K_{rr} & K_{rc} & \widetilde{B}_r^t \\ K_{rc}^t & K_{cc} & \widetilde{B}_c^t \\ \widetilde{B}_r & \widetilde{B}_c & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{z}_r \\ \mathbf{z}_c \\ \boldsymbol{\lambda} \end{pmatrix} = \begin{pmatrix} \widetilde{\mathbf{f}}_r \\ \widetilde{\mathbf{f}}_c \\ \mathbf{0} \end{pmatrix}.$$

In section 4, we will show that the matrix K_{rr} is invertible; see Lemma 4.1. After eliminating $\mathbf{z}_r$, we obtain the following equation for $\mathbf{z}_c$ and $\boldsymbol{\lambda}$:

$$\begin{pmatrix} -F_{cc} & F_{cl} \\ F_{cl}^t & F_{ll} \end{pmatrix} \begin{pmatrix} \mathbf{z}_c \\ \boldsymbol{\lambda} \end{pmatrix} = \begin{pmatrix} -\mathbf{d}_c \\ \mathbf{d}_l \end{pmatrix},$$

where

$$\begin{aligned} F_{ll} &= \widetilde{B}_r K_{rr}^{-1} \widetilde{B}_r^t, \\ F_{cl} &= K_{rc}^t K_{rr}^{-1} \widetilde{B}_r^t - \widetilde{B}_c^t, \\ F_{cc} &= K_{cc} - K_{rc}^t K_{rr}^{-1} K_{rc}, \\ \mathbf{d}_l &= \widetilde{B}_r K_{rr}^{-1} \widetilde{\mathbf{f}}_r, \\ \mathbf{d}_c &= \widetilde{\mathbf{f}}_c - K_{rc}^t K_{rr}^{-1} \widetilde{f}_r. \end{aligned}$$

The matrix F_{cc}, a coarse problem in the FETI-DP formulation, is invertible; see Lemma 4.2. By eliminating $\mathbf{z}_c$, we then obtain the following equation for $\boldsymbol{\lambda}$:

$$F_{DP} \boldsymbol{\lambda} = \mathbf{d}_l - F_{cl}^t F_{cc}^{-1} \mathbf{d}_c, \tag{3.7}$$

where

$$F_{DP} = F_{ll} + F_{cl}^t F_{cc}^{-1} F_{cl}.$$

Since the primal constraints (3.4) are selected from the mortar matching condition, the solution $\boldsymbol{\lambda}$ is not uniquely determined in the space M. We define a subspace

$$\widetilde{M} = \left\{ \boldsymbol{\lambda} \in M \ : \ R^t \boldsymbol{\lambda} = \mathbf{0} \right\}, \tag{3.8}$$

where the matrix R is given in (3.5). The matrix F_{DP} is symmetric and positive definite on $\widetilde{M}$; see Remark 4.3. Hence the solution $\boldsymbol{\lambda}$ of (3.7) is uniquely determined in $\widetilde{M}$.

3.3. Preconditioner. We will define several norms of the finite element function spaces given on the interfaces and propose a preconditioner for the operator F_{DP}.

First, we define a norm for the velocity space W on the interfaces. For $\mathbf{w}_i \in W_i$, we define $S_i \mathbf{w}_i$ by

$$\begin{pmatrix} A_{II}^i & G_{II}^i & A_{Ir}^i & A_{Ic}^i \\ (G_{II}^i)^t & \mathbf{0} & (G_{rI}^i)^t & (G_{cI}^i)^t \\ A_{rI}^i & G_{rI}^i & A_{rr}^i & A_{rc}^i \\ A_{cI}^i & G_{cI}^i & A_{cr}^i & A_{cc}^i \end{pmatrix} \begin{pmatrix} \mathbf{u}_I^i \\ p_I^i \\ \mathbf{w}_r^i \\ \mathbf{w}_c^i \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \mathbf{0} \\ S_i \begin{pmatrix} \mathbf{w}_r^i \\ \mathbf{w}_c^i \end{pmatrix} \end{pmatrix},$$

where the superscript i for a matrix denotes the submatrix corresponding to the subdomain Ω_i. Since the upper left 2×2 matrix

$$\begin{pmatrix} A_{II}^i & G_{II}^i \\ (G_{II}^i)^t & \mathbf{0} \end{pmatrix}$$

represents the local Stokes problem with a Dirichlet boundary condition, it is invertible so that the Schur complement S_i is well defined. We then assemble local Schur complement matrices and define

$$S := \operatorname{diag}(S_1, \ldots, S_N).$$

It can be seen easily that S is symmetric and positive definite on W.

We now introduce finite element spaces given on the nonmortar interfaces:

$$W_{ij} := \left\{ \mathbf{w}_{ij} \in X_{n(ij)}|_{\Gamma_{ij}} \ : \ \mathbf{w}_{ij} \text{ vanishes at the end points of } \Gamma_{ij} \right\},$$
$$W_n := \prod_{i=1}^{N} \prod_{j \in m_i} W_{ij}.$$

Here $n(ij)$ denotes the nonmortar subdomain of the interface Γ_{ij}. We define by $E(\mathbf{w}_n)$ the zero extension of the function $\mathbf{w}_n \in W_n$ to the all interfaces, i.e., mortar and nonmortar interfaces. We further define the following subspaces of W and W_n that satisfy a certain set of constraints:

$$\begin{aligned} \widetilde{W} &:= \left\{ \mathbf{w} \in W \ : \ \int_{\Gamma_{ij}} (\mathbf{w}_i - \mathbf{w}_j)\, ds = 0, \quad \forall\, \Gamma_{ij} \right\}, \\ \widetilde{\widetilde{W}} &:= \left\{ \mathbf{w} \in \widetilde{W} \ : \ \int_{\partial \Omega_i} \mathbf{w}_i \cdot \mathbf{n}_i = 0, \quad \forall i \right\}, \\ \widetilde{W}_n &:= \left\{ \mathbf{w}_n \in W_n \ : \ \int_{\Gamma_{ij}} \mathbf{w}\, ds = 0, \quad \forall\, \Gamma_{ij} \right\}, \end{aligned} \tag{3.9}$$

where $\mathbf{n}_i$ denotes the outward unit normal vector on the subdomain boundary $\partial \Omega_i$.

We now introduce the Neumann–Dirichlet preconditioner $\widehat{F}_{DP}^{-1}$ given by

$$\langle \widehat{F}_{DP}\boldsymbol{\lambda}, \boldsymbol{\lambda}\rangle = \max_{\mathbf{w}_n \in \widetilde{W}_n \setminus \{0\}} \frac{\langle BE(\mathbf{w}_n), \boldsymbol{\lambda}\rangle^2}{\langle SE(\mathbf{w}_n), E(\mathbf{w}_n)\rangle}. \tag{3.10}$$

In the following, we will provide an explicit form of $\widehat{F}_{DP}^{-1}$ in detail. We define the l^2-orthogonal projections P and P_n by

$$P \,:\, M \to \widetilde{M}, \qquad P_n \,:\, W_n \to \widetilde{W}_n.$$

Since the constraints on the spaces $\widetilde{M}$ and $\widetilde{W}_n$ are given locally on each nonmortar interface, these projections are composed of diagonal blocks of projections defined on each nonmortar interface,

$$P = \mathrm{diag}_{i=1,\dots,N}\left(\mathrm{diag}_{j\in m_i}\left(P^{ij}\right)\right), \quad P_n = \mathrm{diag}_{i=1,\dots,N}\left(\mathrm{diag}_{j\in m_i}\left(P_n^{ij}\right)\right), \tag{3.11}$$

where P^{ij} and P_n^{ij} are l^2-orthogonal projections given by

$$P^{ij} \,:\, M|_{\Gamma_{ij}} \to \widetilde{M}|_{\Gamma_{ij}}, \qquad P_n^{ij} \,:\, W_n|_{\Gamma_{ij}} \to \widetilde{W}_n|_{\Gamma_{ij}}.$$

We define the restriction R_{ij} and the extension E_{ij}^i by

$$R_{ij} \,:\, W_n \to W_{ij}, \quad E_{ij}^i \,:\, W_{ij} \to W_i,$$

and then express the zero extension $E(\mathbf{w}_n) = (\mathbf{w}_1, \dots, \mathbf{w}_N)$ as

$$\mathbf{w}_i = E_i \mathbf{w}_n \quad \text{with} \quad E_i = \sum_{j\in m_i} E_{ij}^i R_{ij}. \tag{3.12}$$

Using this notation, the formula (3.10) is written as

$$\langle \widehat{F}_{DP}\boldsymbol{\lambda}, \boldsymbol{\lambda}\rangle = \max_{\mathbf{w}_n \in \widetilde{W}_n \setminus \{0\}} \frac{\langle \widehat{B}\mathbf{w}_n, \boldsymbol{\lambda}\rangle^2}{\langle \widehat{S}\mathbf{w}_n, \mathbf{w}_n\rangle}, \tag{3.13}$$

where

$$\widehat{S} = \sum_{i=1}^{N} E_i^t S_i E_i, \qquad \widehat{B} = \mathrm{diag}_{i=1}^{N}\left(\mathrm{diag}_{j\in m_i}\left(B_{ij}\right)\right). \tag{3.14}$$

Here the matrix B_{ij} is a block of B_i corresponding to the unknowns of the nonmortar interface $\Gamma_{ij} \subset \partial\Omega_i$. It is easy to check that

$$\widehat{B} \,:\, \widetilde{W}_n \to \widetilde{M}$$

is one-to-one for $\dim(\widetilde{W}_n) = \dim(\widetilde{M})$ and $\widehat{B}(\widetilde{W}_n) \subset \widetilde{M}$, and that $\widehat{S}$ is symmetric and positive definite on $\widetilde{W}_n$; see (3.8) and (3.9). Let

$$\widehat{B}_p = P^t \widehat{B} P_n, \qquad \widehat{S}_p = P_n^t \widehat{S} P_n. \tag{3.15}$$

These operators are invertible, and their inverses are denoted by $\widehat{B}_p^{-1}$ and $\widehat{S}_p^{-1}$, respectively. The maximum in (3.13) occurs when $\mathbf{w}_n \in \widetilde{W}_n$ satisfies $\widehat{S}_p \mathbf{w}_n = \widehat{B}_p^t \boldsymbol{\lambda}$, and it gives

$$\langle \widehat{F}_{DP}\boldsymbol{\lambda}, \boldsymbol{\lambda}\rangle = \langle \widehat{B}_p \widehat{S}_p^{-1} \widehat{B}_p^t \boldsymbol{\lambda}, \boldsymbol{\lambda}\rangle.$$

Since $\widehat{B}_p$ is block diagonal, $\widehat{B}_p^{-1}$ can be written as

$$\widehat{B}_p^{-1} = \mathrm{diag}_{i=1}^{N}\left(\mathrm{diag}_{j\in m_i}\left((B_p^{ij})^{-1}\right)\right), \qquad B_p^{ij} = (P^{ij})^t B_{ij} P_n^{ij}. \tag{3.16}$$

By using the expressions in (3.11), (3.12), and (3.14)–(3.16), we obtain

$$\widehat{F}_{DP}^{-1} = (\widehat{B}_p^t)^{-1}\widehat{S}_p\widehat{B}_p^{-1} = \sum_{i=1}^{N} B_{i,n}^t S_i B_{i,n},$$

where $B_{i,n}$ is given by

$$B_{i,n} = \begin{pmatrix} \mathrm{diag}_{j\in m_i}\left(P_n^{ij}(B_p^{ij})^{-1}\right) \\ 0 \end{pmatrix} R_i.$$

Here $R_i : M \to \Pi_{j\in m_i} M_{ij}$ is the restriction, and the zero submatrix corresponds to the unknowns of the mortar edges and corners that belong to Ω_i. The matrix $B_{i,n}$ provides each subdomain problem with Neumann boundary conditions on the nonmortar edges and a zero Dirichlet boundary condition on the remaining part of the subdomain boundary. Hence we call it a Neumann–Dirichlet preconditioner.

4. Condition number estimation. In this section we analyze the condition number bound of the FETI-DP operator with the Neumann–Dirichlet preconditioner. In advance, we will provide the following two lemmas, which prove that the matrices K_{rr} and F_{cc} are invertible.

LEMMA 4.1. [illegible] K_{rr} [illegible] $X_I \times Q_I \times W_r$ [illegible]. Since the upper left 2×2 matrix of K_{rr} is invertible, it suffices to show that the Schur complement S_{rr} is invertible on W_r:

$$\begin{pmatrix} A_{II} & G_{II} & A_{Ir} \\ G_{II}^t & \mathbf{0} & G_{rI}^t \\ A_{rI} & G_{rI} & A_{rr} \end{pmatrix} \begin{pmatrix} \mathbf{u}_I \\ p_I \\ \mathbf{u}_r \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \mathbf{0} \\ S_{rr}\mathbf{u}_r \end{pmatrix}. \tag{4.1}$$

For any $\mathbf{u}_r \in W_r$, we consider

$$\begin{aligned} \mathbf{u}_r^t S_{rr}\mathbf{u}_r &= \begin{pmatrix} \mathbf{u}_I \\ p_I \\ \mathbf{u}_r \end{pmatrix}^t \begin{pmatrix} A_{II} & G_{II} & A_{Ir} \\ G_{II}^t & \mathbf{0} & G_{rI}^t \\ A_{rI} & G_{rI} & A_{rr} \end{pmatrix} \begin{pmatrix} \mathbf{u}_I \\ p_I \\ \mathbf{u}_r \end{pmatrix} \\ &= \begin{pmatrix} \mathbf{u}_I \\ \mathbf{u}_r \end{pmatrix}^t \begin{pmatrix} A_{II} & A_{Ir} \\ A_{rI} & A_{rr} \end{pmatrix} \begin{pmatrix} \mathbf{u}_I \\ \mathbf{u}_r \end{pmatrix} + 2p_I^t \begin{pmatrix} G_{II}^t & G_{rI}^t \end{pmatrix} \begin{pmatrix} \mathbf{u}_I \\ \mathbf{u}_r \end{pmatrix} \\ &= \begin{pmatrix} \mathbf{u}_I \\ \mathbf{u}_r \end{pmatrix}^t \begin{pmatrix} A_{II} & A_{Ir} \\ A_{rI} & A_{rr} \end{pmatrix} \begin{pmatrix} \mathbf{u}_I \\ \mathbf{u}_r \end{pmatrix}. \end{aligned}$$

Here $\mathbf{u}_I$ and p_I are the solution of (4.1), and we have used that $G_{II}^t\mathbf{u}_I + G_{rI}^t\mathbf{u}_r = \mathbf{0}$. The last equality in the above equation gives that S_{rr} is symmetric and positive definite on W_r. ∎

LEMMA 4.2. [illegible] Ω [illegible]

$$-\sum_i (p_0, \nabla\cdot\mathbf{v}_r)_{\Omega_i} + \sum_{i,j} \boldsymbol{\mu}\cdot\int_{\Gamma_{ij}} (\mathbf{v}_r^i - \mathbf{v}_r^j)\,ds = 0 \quad \forall\ \mathbf{v}_r \in W_r,$$

$$-\sum_i (p_0, \nabla\cdot\mathbf{v}_c)_{\Omega_i} + \sum_{i,j} \boldsymbol{\mu}\cdot\int_{\Gamma_{ij}} (\mathbf{v}_c^i - \mathbf{v}_c^j)\,ds = 0 \quad \forall\ \mathbf{v}_c \in W_c$$

[illegible] $\begin{pmatrix} p_0 \\ \boldsymbol{\mu} \end{pmatrix}$ [illegible] F_{cc} [illegible]

. We define a Schur complement matrix

$$\begin{pmatrix} A_{II} & G_{II} & A_{Ir} & A_{Ic} \\ G_{II}^t & 0 & G_{rI}^t & G_{cI}^t \\ A_{rI} & G_{rI} & A_{rr} & A_{rc} \\ A_{cI} & G_{cI} & A_{cr} & A_{cc} \end{pmatrix} \begin{pmatrix} \mathbf{u}_I \\ p_I \\ \mathbf{u}_r \\ \mathbf{u}_c \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \mathbf{0} \\ \begin{pmatrix} S_{rr} & S_{rc} \\ S_{cr} & S_{cc} \end{pmatrix} \begin{pmatrix} \mathbf{u}_r \\ \mathbf{u}_c \end{pmatrix} \end{pmatrix}.$$

As in the proof of the previous lemma, we can show that the Schur complement

$$\begin{pmatrix} S_{rr} & S_{rc} \\ S_{cr} & S_{cc} \end{pmatrix}$$

is symmetric and positive definite. The matrix F_{cc} is then given by

$$\begin{pmatrix} S_{rr} & S_{rc} & G_{r0} & B_r^t R \\ S_{cr} & S_{cc} & G_{c0} & B_c^t R \\ G_{r0}^t & G_{c0}^t & \mathbf{0} & \mathbf{0} \\ R^t B_r & R^t B_c & \mathbf{0} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{u}_r \\ \mathbf{u}_c \\ p_0 \\ \boldsymbol{\mu} \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ F_{cc} \begin{pmatrix} \mathbf{u}_c \\ p_0 \\ \boldsymbol{\mu} \end{pmatrix} \end{pmatrix}. \tag{4.2}$$

We assume that $\mathbf{u}_c$, p_0, and $\boldsymbol{\mu}$ give

$$F_{cc} \begin{pmatrix} \mathbf{u}_c \\ p_0 \\ \boldsymbol{\mu} \end{pmatrix} = 0,$$

and we will show that they are zero. Multiplying (4.2) by $\begin{pmatrix} \mathbf{u}_r^t & \mathbf{u}_c^t & -p_0^t & -\boldsymbol{\mu}^t \end{pmatrix}$, we obtain

$$\begin{pmatrix} \mathbf{u}_r \\ \mathbf{u}_c \end{pmatrix}^t \begin{pmatrix} S_{rr} & S_{rc} \\ S_{cr} & S_{cc} \end{pmatrix} \begin{pmatrix} \mathbf{u}_r \\ \mathbf{u}_c \end{pmatrix} + \begin{pmatrix} \mathbf{u}_r \\ \mathbf{u}_c \end{pmatrix}^t \begin{pmatrix} G_{r0} & B_r^t R \\ G_{c0} & B_c^t R \end{pmatrix} \begin{pmatrix} p_0 \\ \boldsymbol{\mu} \end{pmatrix} - \begin{pmatrix} p_0 \\ \boldsymbol{\mu} \end{pmatrix}^t \begin{pmatrix} G_{r0}^t & G_{c0}^t \\ R^t B_r & R^t B_c \end{pmatrix} \begin{pmatrix} \mathbf{u}_r \\ \mathbf{u}_c \end{pmatrix} = 0.$$

It follows that $\begin{pmatrix} \mathbf{u}_r \\ \mathbf{u}_c \end{pmatrix} = 0$ because the Schur complement matrix is symmetric and positive definite and the last two terms cancel each other. Equation (4.2) then reduces to

$$\begin{pmatrix} G_{r0} & B_r^t R \\ G_{c0} & B_c^t R \end{pmatrix} \begin{pmatrix} p_0 \\ \boldsymbol{\mu} \end{pmatrix} = 0.$$

This is equivalent to the equation in the assumption. Hence we have $\begin{pmatrix} p_0 \\ \boldsymbol{\mu} \end{pmatrix} = 0$. □

. . . . 4.1. Most triangulations of the domain Ω satisfy the assumption of Lemma 4.2 because the number of velocity unknowns $\mathbf{v}_r$ and $\mathbf{v}_c$ is usually grater than the number of unknowns p_0 and $\boldsymbol{\mu}$.

LEMMA 4.3. . .

$$B(\widetilde{\widetilde{W}}) = B(\widetilde{W}) = \widetilde{M}.$$

. . . . From $\widetilde{\widetilde{W}} \subset \widetilde{W}$, the inclusion $B(\widetilde{\widetilde{W}}) \subset B(\widetilde{W})$ is obvious.

We will show that

$$B(\widetilde{W}) \subset B(\widetilde{\widetilde{W}}).$$

Let $E(\mathbf{w}_n) = (\mathbf{w}_1, \ldots, \mathbf{w}_N) \in W$ be the zero extension of $\mathbf{w}_n \in W_n$. Since $\mathbf{w}_j|_{\Gamma_{ij}} = 0$ for $j \in m_i$, i.e., for functions on mortar interfaces, and $E(\mathbf{w}_n)$ has zero value at subdomain corners, the identity holds

$$B(E(\mathbf{w}_n)) = \widehat{B}\mathbf{w}_n, \tag{4.3}$$

where the matrix $\widehat{B}$ is defined in (3.14), and it is one-to-one from $\widetilde{W}_n$ onto $\widetilde{M}$,

$$\widehat{B}(\widetilde{W}_n) = \widetilde{M}. \tag{4.4}$$

For $\mathbf{w}_n \in \widetilde{W}_n$, the zero extension $E(\mathbf{w}_n) = (\mathbf{w}_1, \ldots, \mathbf{w}_N)$ satisfies

$$\int_{\Gamma_{ij}} \mathbf{w}_i \, ds = \mathbf{0} \qquad \forall\, i = 1, \ldots, N, \quad \forall \Gamma_{ij} \subset \partial\Omega_i,$$

and then the compatibility condition of the local Stokes problem holds:

$$\int_{\partial\Omega_i} \mathbf{w}_i \cdot \mathbf{n}_i \, ds = 0.$$

This implies that $E(\mathbf{w}_n)$, for $\mathbf{w}_n \in \widetilde{W}_n$, belongs to the space $\widetilde{\widetilde{W}}$. From this fact and (4.3) we obtain

$$\widehat{B}(\widetilde{W}_n) \subset B(\widetilde{\widetilde{W}}). \tag{4.5}$$

From the definition of $\widetilde{W}$ and $\widetilde{M}$, we have

$$B(\widetilde{W}) = \widetilde{M}. \tag{4.6}$$

Combining (4.6), (4.4), and (4.5), we prove $B(\widetilde{W}) \subset B(\widetilde{\widetilde{W}})$. ∎

REMARK 4.2. Lemma 4.3 says that the constraints

$$\int_{\partial\Omega_i} \mathbf{w}_i \cdot \mathbf{n}_i \, ds = 0$$

do not affect the range space $B(\widetilde{W})$.

We now provide the following well-known identity that is useful for the analysis of the condition number bound.

LEMMA 4.4. For $\boldsymbol{\lambda} \in \widetilde{M}$,

$$\langle F_{DP}\boldsymbol{\lambda}, \boldsymbol{\lambda}\rangle = \max_{\mathbf{w} \in \widetilde{\widetilde{W}} \setminus \{0\}} \frac{\langle B\mathbf{w}, \boldsymbol{\lambda}\rangle^2}{\langle S\mathbf{w}, \mathbf{w}\rangle}.$$

PROOF. The problem (3.6) is equivalent to the following min-max problem:

$$\max_{\boldsymbol{\lambda} \in B(\widetilde{\widetilde{W}})} \min_{\mathbf{w} \in \widetilde{\widetilde{W}}} \left\{ \frac{1}{2}\langle S\mathbf{w}, \mathbf{w}\rangle - \langle \mathbf{d}, \mathbf{w}\rangle + \langle B\mathbf{w}, \boldsymbol{\lambda}\rangle \right\}, \tag{4.7}$$

where $\mathbf{d}$ is the Schur complement forcing vector obtained from $\begin{pmatrix} \mathbf{f}_I^t & \mathbf{0}^t & \mathbf{f}_r^t & \mathbf{f}_c^t \end{pmatrix}^t$ after solving local Stokes problems.

Let P_W be the l^2-orthogonal projection

$$P_W : W \to \widetilde{\widetilde{W}}.$$

Note that $B(\widetilde{\widetilde{W}}) = \widetilde{M}$ from Lemma 4.3, and P is the projection from M onto $\widetilde{M}$ introduced in section 3. We consider

$$\mathbf{w} = (\mathbf{w}_1, \ldots, \mathbf{w}_N) \quad \text{such that} \quad \mathbf{w}_i = \begin{pmatrix} \mathbf{u}_r|_{\partial\Omega_i} \\ L_c^i \mathbf{u}_c \end{pmatrix},$$

where $(\mathbf{u}_r, \mathbf{u}_c)$ is from the solution of (3.6). The fifth and sixth rows in (3.6) give

$$\int_{\partial\Omega_i} \mathbf{w}_i \cdot \mathbf{n}_i \, ds = 0 \quad \forall i = 1, \ldots, N,$$
$$\int_{\Gamma_{ij}} (\mathbf{w}_i - \mathbf{w}_j) \, ds = 0 \quad \forall \, \Gamma_{ij}.$$

These equations imply that $\mathbf{w}$ belongs to $\widetilde{\widetilde{W}}$. By taking the Euler–Lagrangian in (4.7), we can see that $(\mathbf{w}, \boldsymbol{\lambda}) \in \widetilde{\widetilde{W}} \times \widetilde{M}$ from the solution of (3.6) satisfies

$$\begin{pmatrix} S_p & B_p^t \\ B_p & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{w} \\ \boldsymbol{\lambda} \end{pmatrix} = \begin{pmatrix} P_W^t \mathbf{d} \\ \mathbf{0} \end{pmatrix}, \tag{4.8}$$

where

$$S_p = P_W^t S P_W, \qquad B_p = P^t B P_W.$$

Since S is symmetric and positive definite (s.p.d.) on $\widetilde{\widetilde{W}}$, the equation for $\boldsymbol{\lambda}$ follows by eliminating $\mathbf{w}$ in (4.8):

$$B_p S_p^{-1} B_p^t \boldsymbol{\lambda} = B_p S_p^{-1} \mathbf{d},$$

which is the same as (3.7). Therefore we have the identity

$$B_p S_p^{-1} B_p^t = F_{DP}. \tag{4.9}$$

For $\boldsymbol{\lambda} \in \widetilde{M}$, we consider

$$\max_{\mathbf{w} \in \widetilde{\widetilde{W}} \setminus \{0\}} \frac{\langle B\mathbf{w}, \boldsymbol{\lambda} \rangle^2}{\langle S\mathbf{w}, \mathbf{w} \rangle}. \tag{4.10}$$

Since S is s.p.d. on $\widetilde{\widetilde{W}}$, the maximum in (4.10) occurs when $S_p \mathbf{w} = B_p^t \boldsymbol{\lambda}$, and it gives

$$\max_{\mathbf{w} \in \widetilde{\widetilde{W}} \setminus \{0\}} \frac{\langle B\mathbf{w}, \boldsymbol{\lambda} \rangle^2}{\langle S\mathbf{w}, \mathbf{w} \rangle} = \langle B_p S_p^{-1} B_p^t \boldsymbol{\lambda}, \boldsymbol{\lambda} \rangle. \tag{4.11}$$

From (4.9) and (4.11), the required identity follows. □

4.3. From (4.9), we can see that F_{DP} is s.p.d. on $\widetilde{M}$.

4.4. From Lemma 4.4, $E(\mathbf{w}_n) \in \widetilde{\widetilde{W}}$ for $\mathbf{w}_n \in \widetilde{W}_n$, and (3.10), we obtain the following lower bound of the FETI-DP operator:

$$\begin{aligned}\langle F_{DP}\boldsymbol{\lambda},\boldsymbol{\lambda}\rangle &= \max_{\mathbf{w}\in\widetilde{\widetilde{W}}\setminus\{0\}} \frac{\langle B\mathbf{w},\boldsymbol{\lambda}\rangle^2}{\langle S\mathbf{w},\mathbf{w}\rangle} \\ &\geq \max_{\mathbf{w}_n\in\widetilde{W}_n\setminus\{0\}} \frac{\langle BE(\mathbf{w}_n),\boldsymbol{\lambda}\rangle^2}{\langle SE(\mathbf{w}_n),E(\mathbf{w}_n)\rangle} \\ &= \langle \widehat{F}_{DP}\boldsymbol{\lambda},\boldsymbol{\lambda}\rangle.\end{aligned}$$

The following lemma can be found in Bramble and Pasciak [5].

LEMMA 4.5. $\mathbf{w}_i \in W_i$

$$C_1\beta^2\langle S_i\mathbf{w}_i,\mathbf{w}_i\rangle \leq |\mathbf{w}_i|^2_{1/2,\partial\Omega_i} \leq C_2\langle S_i\mathbf{w}_i,\mathbf{w}_i\rangle,$$

β Ω_i C_1 C_2 h_i H_i

Since we have chosen the inf-sup stable $P_1(h) - P_0(2h)$ finite element space, the constant β is independent of h_i and H_i.

We also have the following result, which is derived in [17, Lemma 5.1].

LEMMA 4.6. $\mathbf{w} = (\mathbf{w}_1,\ldots,\mathbf{w}_N) \in W$

$$\|\mathbf{w}_i - \mathbf{w}_j\|^2_{H_{00}^{1/2}(\Gamma_{ij})} \leq C \max_{l\in\{i,j\}} \left\{\left(1+\log\frac{H_l}{h_l}\right)^2\right\}\left(|\mathbf{w}_i|^2_{1/2,\partial\Omega_i} + |\mathbf{w}_j|^2_{1/2,\partial\Omega_j}\right).$$

DEFINITION 4.7. $\pi_{ij} : [H_{00}^{1/2}(\Gamma_{ij})]^2 \to W_{ij}$

$$\int_{\Gamma_{ij}} (\mathbf{v} - \pi_{ij}\mathbf{v})\cdot\boldsymbol{\lambda}_{ij}\,ds = 0 \ \ \forall\boldsymbol{\lambda}_{ij} \in M_{ij}.$$

From Lemma 2.2 in [1], π_{ij} is continuous on $H_{00}^{1/2}(\Gamma_{ij})$; i.e., there exists a constant C such that

$$\|\pi_{ij}\mathbf{v}\|_{H_{00}^{1/2}(\Gamma_{ij})} \leq C\|\mathbf{v}\|_{H_{00}^{1/2}(\Gamma_{ij})} \qquad \forall\mathbf{v} \in [H_{00}^{1/2}(\Gamma_{ij})]^2. \tag{4.12}$$

We now provide an upper bound of the FETI-DP operator.

LEMMA 4.8. $\boldsymbol{\lambda} \in \widetilde{M}$

$$\max_{\mathbf{w}\in\widetilde{\widetilde{W}}\setminus\{0\}} \frac{\langle B\mathbf{w},\boldsymbol{\lambda}\rangle^2}{\langle S\mathbf{w},\mathbf{w}\rangle} \leq C\frac{1}{\beta^2}\max_{i=1,\ldots,N}\left\{\left(1+\log\frac{H_i}{h_i}\right)^2\right\}\langle \widehat{F}_{DP}\boldsymbol{\lambda},\boldsymbol{\lambda}\rangle.$$

From the definition of π_{ij}, we have

$$\langle B\mathbf{w},\boldsymbol{\lambda}\rangle = \sum_{i=1}^{N}\sum_{j\in m_i}\int_{\Gamma_{ij}}(\mathbf{w}_i-\mathbf{w}_j)\cdot\boldsymbol{\lambda}_{ij}\,ds = \sum_{i=1}^{N}\sum_{j\in m_i}\int_{\Gamma_{ij}}\pi_{ij}(\mathbf{w}_i-\mathbf{w}_j)\cdot\boldsymbol{\lambda}_{ij}\,ds. \tag{4.13}$$

Let $\mathbf{z}_n \in W_n$ be $\mathbf{z}_n|_{\Gamma_{ij}} = \pi_{ij}(\mathbf{w}_i - \mathbf{w}_j)$. Since $\binom{1}{0}, \binom{0}{1} \in M_{ij}$ and $\mathbf{w} \in \widetilde{\widetilde{W}}$, we have

$$\int_{\Gamma_{ij}} \mathbf{z}_n \, ds = \int_{\Gamma_{ij}} (\mathbf{w}_i - \mathbf{w}_j) \, ds = 0, \tag{4.14}$$

so that $\mathbf{z}_n \in \widetilde{W}_n$. From (3.10), the definition of $\widehat{F}_{DP}$, (4.13) can be bounded by

$$\langle B\mathbf{w}, \boldsymbol{\lambda}\rangle^2 = \langle BE(\mathbf{z}_n), \boldsymbol{\lambda}\rangle^2 \le \langle \widehat{F}_{DP}\boldsymbol{\lambda}, \boldsymbol{\lambda}\rangle \langle SE(\mathbf{z}_n), E(\mathbf{z}_n)\rangle, \tag{4.15}$$

where $E(\mathbf{z}_n) = (\mathbf{z}_1, \ldots, \mathbf{z}_N)$ is the zero extension of $\mathbf{z}_n$. From Lemma 4.5, (2.5), (4.12), and Lemma 4.6, we obtain

$$\begin{aligned}
\langle SE(\mathbf{z}_n), E(\mathbf{z}_n)\rangle &= \sum_{i=1}^{N} \langle S_i \mathbf{z}_i, \mathbf{z}_i\rangle \\
&\le C\frac{1}{\beta^2} \sum_{i=1}^{N} |\mathbf{z}_i|^2_{1/2,\partial\Omega_i} \\
&\le C\frac{1}{\beta^2} \sum_{i=1}^{N} \sum_{j\in m_i} \|\mathbf{w}_i - \mathbf{w}_j\|^2_{H_{00}^{1/2}(\Gamma_{ij})} \\
&\le C\frac{1}{\beta^2} \max_{i=1,\ldots,N} \left\{ \left(1 + \log \frac{H_i}{h_i}\right)^2 \right\} \sum_{i=1}^{N} |\mathbf{w}_i|^2_{1/2,\partial\Omega_i} \\
&\le C\frac{1}{\beta^2} \max_{i=1,\ldots,N} \left\{ \left(1 + \log \frac{H_i}{h_i}\right)^2 \right\} \langle S\mathbf{w}, \mathbf{w}\rangle.
\end{aligned} \tag{4.16}$$

The estimates (4.15) and (4.16) give the required bound. ∎

From Remark 4.4 and Lemmas 4.4 and 4.8, we obtain the condition number bound of our FETI-DP algorithm.

THEOREM 4.9. [illegible] (3.10) [illegible]

$$\kappa(\widehat{F}_{DP}^{-1} F_{DP}) \le C\frac{1}{\beta^2} \max_{i=1,\ldots,N} \left\{ \left(1 + \log \frac{H_i}{h_i}\right)^2 \right\}.$$

5. Numerical results. In this section, we provide numerical tests for the proposed FETI-DP algorithm. The following Stokes problem is considered:

$$\begin{aligned}
-\triangle \mathbf{u} + \nabla p &= \mathbf{f} \quad \text{in } \Omega, \\
-\nabla \cdot \mathbf{u} &= 0 \quad \text{in } \Omega, \\
\mathbf{u} &= \mathbf{0} \quad \text{on } \partial\Omega,
\end{aligned} \tag{5.1}$$

where Ω is a unit rectangle in $\mathbb{R}^2$ and $\mathbf{f}$ is given from the exact solution

$$\mathbf{u} = \begin{pmatrix} \sin^3(\pi x)\sin^2(\pi y)\cos(\pi y) \\ -\sin^2(\pi x)\sin^3(\pi y)\cos(\pi x) \end{pmatrix} \quad \text{and} \quad p = x^2 - y^2.$$

We consider only the uniform partition of Ω. The notation $N = 4 \times 4$ means that Ω is partitioned into 4×4 square subdomains. With this partition, we triangulate

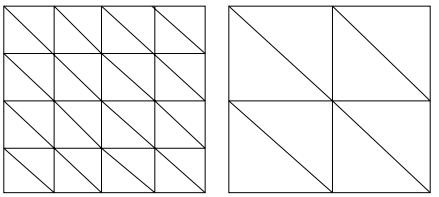


FIG. 1. *Triangulations for velocity (left) and for pressure (right) when* $n = 5$.

TABLE 1

CG iterations (condition number) when n *(the number of nodes) increases with a fixed* $N = 4 \times 4$ *(the number of subdomains):* $\widehat{F}_{DP}^{-1}F_{DP}$ *(with preconditioner),* F_{DP} *(without preconditioner).*

n	Matching		Nonmatching	
	F_{DP}	$\widehat{F}_{DP}^{-1}F_{DP}$	F_{DP}	$\widehat{F}_{DP}^{-1}F_{DP}$
5	12(5.23)	9(2.62)	16(8.35)	12(3.75)
9	24(2.50e+1)	13(4.39)	50(1.15e+2)	15(5.79)
17	37(6.68e+1)	15(5.94)	86(5.01e+2)	17(7.93)
33	45(1.45e+2)	17(7.75)	119(1.31e+3)	20(9.88)
65	58(2.69e+2)	19(9.85)	153(3.29e+3)	22(1.20e+1)

TABLE 2

CG iterations (condition number) when N *(the number of subdomains) increases with* $n = 5$ *and* 9*:* $\widehat{F}_{DP}^{-1}F_{DP}$ *(with preconditioner),* F_{DP} *(without preconditioner).*

n	N	Matching		Nonmatching	
		F_{DP}	$\widehat{F}_{DP}^{-1}F_{DP}$	F_{DP}	$\widehat{F}_{DP}^{-1}F_{DP}$
5	4×4	12(5.23)	9(2.62)	16(8.35)	12(3.75)
	8×8	12(5.42)	9(2.62)	16(9.18)	12(3.68)
	16×16	10(5.54)	9(2.55)	16(9.57)	11(3.42)
	32×32	10(5.61)	9(2.53)	16(10.88)	12(3.78)
9	4×4	24(2.50e+1)	13(4.39)	50(1.15e+2)	15(5.79)
	8×8	25(2.60e+1)	13(4.35)	53(1.19e+2)	15(6.21)
	16×16	24(2.62e+1)	12(4.27)	57(1.34e+2)	16(6.27)
	32×32	23(2.62e+1)	12(4.27)	56(1.25e+2)	16(6.24)

each subdomain in the following manner. Let $n = 4k + 1$ with k an integer. For matching triangulations, we take the same uniform triangulation in each subdomain with $(n-1)/2+1$ nodes, including end points on both horizontal and vertical edges. We denote it by $\Omega_i^{2h_i}$, a triangulation for the pressure. A triangulation for the velocity, $\Omega_i^{h_i}$, is then given by dividing each triangle in $\Omega_i^{2h_i}$ into four triangles, as in Figure 1. For nonmatching grids, we take randomly $(n-1)/2+1$ nodes on each horizontal and vertical edge and then generate a nonuniform structured triangulation $\Omega_i^{2h_i}$ from the nodes. A triangulation $\Omega_i^{h_i}$ is obtained from $\Omega_i^{2h_i}$, as before.

We solved the FETI-DP operator with and without preconditioner, varying N and n on both matching and nonmatching triangulations. The CG (conjugate gradient) iteration is terminated when the relative residual reduces by 10^{-6}.

In Tables 1 and 2, the number of CG iterations and the corresponding condition number are shown, varying N and n. Table 1 shows the numbers when $(n-1)$ increases two-fold with fixed $N = 4 \times 4$. The preconditioner performs well, and the condition numbers seem to exhibit $\log^2$-growth as n increases. The preconditioner is much more efficient for the nonmatching case. Table 2 shows the performance of the

TABLE 3

Errors (factors) on matching grids: $\|\mathbf{u}-\mathbf{u}^h\|_{1,*}$ *(broken* H^1*-norm error for velocity),* $\|\mathbf{u}-\mathbf{u}^h\|_0$ *(*L^2*-norm error for velocity),* $\|p-p^h\|_0$ *(*L^2*-norm error for pressure).*

$N = 4\times 4$	$n=5$	$n=9$	$\|\mathbf{u}-\mathbf{u}^h\|_{1,*}$	$\|\mathbf{u}-\mathbf{u}^h\|_0$	$\|p-p^h\|_0$
n	N	N			
5	4×4		3.37e-1	3.75e-3	1.07e-1
9	8×8	4×4	1.72e-1 (0.510)	1.02e-3 (0.272)	5.99e-2 (0.559)
17	16×16	8×8	8.64e-2 (0.502)	2.64e-4 (0.258)	3.08e-2 (0.514)
33	32×32	16×16	4.32e-2 (0.500)	6.65e-5 (0.258)	1.55e-2 (0.503)
65		32×32	2.16e-2 (0.500)	1.66e-5 (0.249)	7.79e-3 (0.502)

TABLE 4

Errors (factors) on nonmatching grids when n *increases with a fixed* $N = 4\times 4$*:* $\|\mathbf{u}-\mathbf{u}^h\|_{1,*}$ *(broken* H^1*-norm error for velocity),* $\|\mathbf{u}-\mathbf{u}^h\|_0$ *(*L^2*-norm error for velocity),* $\|p-p^h\|_0$ *(*L^2*-norm error for pressure).*

n	$\|\mathbf{u}-\mathbf{u}^h\|_{1,*}$	$\|\mathbf{u}-\mathbf{u}^h\|_0$	$\|p-p^h\|_0$
5	3.41e-1	3.79e-3	1.05e-1
9	1.78e-1 (0.521)	1.10e-3 (0.290)	6.08e-2 (0.579)
17	8.95e-2 (0.502)	2.85e-4 (0.259)	3.16e-2 (0.517)
33	4.48e-2 (0.500)	7.21e-5 (0.252)	1.58e-2 (0.500)
65	2.24e-2 (0.500)	1.81e-5 (0.251)	7.93e-3 (0.501)

TABLE 5

Errors (factors) on nonmatching grids when N *increases with* $n = 5$ *and* 9*:* $\|\mathbf{u}-\mathbf{u}^h\|_{1,*}$ *(broken* H^1*-norm error for velocity),* $\|\mathbf{u}-\mathbf{u}^h\|_0$ *(*L^2*-norm error for velocity),* $\|p-p^h\|_0$ *(*L^2*-norm error for pressure).*

n	N	$\|\mathbf{u}-\mathbf{u}^h\|_{1,*}$	$\|\mathbf{u}-\mathbf{u}^h\|_0$	$\|p-p^h\|_0$
5	4×4	1.78e-1	1.10e-3	6.08e-2
	8×8	8.95e-2 (0.502)	2.94e-4 (0.269)	3.28e-2 (0.539)
	16×16	4.49e-2 (0.501)	7.33e-5 (0.249)	1.63e-2 (0.496)
	32×32	2.25e-2 (0.501)	1.84e-5 (0.251)	8.18e-3 (0.501)
9	4×4	3.37e-1	3.75e-4	1.07e-1
	8×8	1.72e-1 (0.510)	1.02e-3 (0.272)	5.99e-2 (0.559)
	16×16	8.64e-2 (0.502)	2.64e-4 (0.258)	3.08e-2 (0.514)
	33×32	4.32e-2 (0.500)	6.65e-5 (0.258)	1.55e-2 (0.503)

preconditioner when N increases with $n = 5$ and 9. The CG iteration becomes stable as N increases.

We have further observed the convergent behaviors of the approximated solution. The H^1 and L^2-errors for velocity and pressure are examined. The errors and reduction factors are shown in Table 3 for various N and n on matching grids. Three cases are considered: when $n-1$ increases two-fold with a fixed $N = 4\times 4$ and when N increases two-fold in both horizontal and vertical edges of Ω with a fixed $n = 5$ or 9. For all cases, we can see that the H^1-error for velocity and the L^2-error for pressure reduce by half, and the L^2-error for velocity reduces by quarter. These factors are optimal for the $P_1(h) - P_0(2h)$ finite element space.

Tables 4–5 show errors and reduction factors for the nonmatching case. In Table 4, we observe the optimal convergence of errors as $(n-1)$ increases two-fold with a fixed $N = 4\times 4$. When $n = 5$ and 9, as N increases, the errors also show the optimal convergence in Table 5.

As mentioned in section 2, if the inf-sup constant of the space $V\times P$ is independent

TABLE 6
Inf-sup constant β_0 when N increases with $n = 5$ and $n = 9$.

N	$n = 5$		$n = 9$	
	Nonmatching	Matching	Nonmatching	Matching
4×4	0.5780	0.5785	0.5921	0.5924
8×8	0.5293	0.5294	0.5352	0.5353
16×16	0.5008	0.5010	0.5041	0.5042
32×32	0.4827	0.4828	0.4854	0.4848

TABLE 7
Inf-sup constant β_0 when n increases with a fixed $N = 4 \times 4$.

n	Nonmatching	Matching
5	0.5780	0.5785
9	0.5921	0.5294
17	0.5966	0.5967
33	0.5973	0.5979
65	0.5983	0.5983

of N and n, then the optimality of approximation can be shown for the space $V \times P$. Let β^* and β be the inf-sup constants for the space $V \times P$ and the $P_1(h) - P_0(2h)$ finite elements, respectively, and β_0 be the inf-sup constant for the space $V \times \Pi_0$; see (2.9) and (2.2) for the definition. The constant β^* depends on β and β_0 from the trick conceived by Boland and Nicolaides [4]. Hence, if the constant β_0 is independent of n and N, then the same holds for β^*. We showed that β_0 is independent of n and N numerically for various N and n. The results are given in Tables 6 and 7 for both matching and nonmatching cases. We observe that β_0 becomes stable as N or n increases.

REFERENCES

[1] F. B. BELGACEM, *The mortar finite element method with Lagrange multipliers*, Numer. Math., 84 (1999), pp. 173–197.

[2] F. B. BELGACEM, *The mixed mortar finite element method for the incompressible Stokes problem: Convergence analysis*, SIAM J. Numer. Anal., 37 (2000), pp. 1085–1100.

[3] C. BERNARDI, Y. MADAY, AND A. T. PATERA, *A new nonconforming approach to domain decomposition: The mortar element method*, in Nonlinear Partial Differential Equations and Their Applications, Collège de France Seminar, Vol. XI (Paris, 1989–1991), Pitman Res. Notes Math. Ser. 299, Longman Scientific and Technical, Harlow, UK, 1994, pp. 13–51.

[4] J. M. BOLAND AND R. A. NICOLAIDES, *Stability of finite elements under divergence constraints*, SIAM J. Numer. Anal., 20 (1983), pp. 722–731.

[5] J. H. BRAMBLE AND J. E. PASCIAK, *A domain decomposition technique for Stokes problems*, Appl. Numer. Math., 6 (1990), pp. 251–261.

[6] F. BREZZI AND M. FORTIN, *Mixed and Hybrid Finite Element Methods*, Springer Ser. Comput. Math. 15, Springer-Verlag, New York, 1991.

[7] B. COCKBURN AND C.-W. SHU, *The local discontinuous Galerkin method for time-dependent convection-diffusion systems*, SIAM J. Numer. Anal., 35 (1998), pp. 2440–2463.

[8] M. DRYJA AND W. PROSKUROWSKI, *A FETI-DP method for the mortar discretization of elliptic problems with discontinuous coefficients*, in Domain Decomposition Methods in Science and Engineering, Lecture Notes Comput. Sci. Eng. 40, Springer, Berlin, 2004, pp. 345–352.

[9] M. DRYJA AND O. B. WIDLUND, *A FETI-DP method for a mortar discretization of elliptic problems*, in Recent Developments in Domain Decomposition Methods (Zürich, 2001), Lecture Notes Comput. Sci. Eng. 23, Springer, Berlin, 2002, pp. 41–52.

[10] M. DRYJA AND O. B. WIDLUND, *A generalized FETI-DP method for a mortar discretization of elliptic problems*, in Domain Decomposition Methods in Science and Engineering (Cocoyoc, Mexico, 2002), UNAM, Mexico City, 2003, pp. 27–38.

[11] C. Farhat, M. Lesoinne, P. LeTallec, K. Pierson, and D. Rixen, *FETI-DP: A dual-primal unified FETI method.* I. *A faster alternative to the two-level FETI method*, Int. J. Numer. Methods Engrg., 50 (2001), pp. 1523–1544.

[12] C. Farhat and F.-X. Roux, *A method of finite element tearing and interconnecting and its parallel solution algorithm*, Int. J. Numer. Methods Engrg., 32 (1991), pp. 1205–1227.

[13] H. H. Kim and C.-O. Lee, *A preconditioner for the FETI-DP formulation with mortar methods in two dimensions*, SIAM J. Numer. Anal., 42 (2005), pp. 2159–2175.

[14] A. Klawonn, O. B. Widlund, and M. Dryja, *Dual-primal FETI methods for three-dimensional elliptic problems with heterogeneous coefficients*, SIAM J. Numer. Anal., 40 (2002), pp. 159–179.

[15] J. Li, *A dual-primal FETI method for incompressible Stokes equations*, Numer. Math., 102 (2005), pp. 257–275.

[16] J. Li, *Dual-Primal FETI Methods for Stationary Stokes and Navier–Stokes Equations*, Ph.D. thesis, Department of Computer Science, Courant Institute of Mathematical Sciences, New York University, New York, 2002.

[17] J. Mandel and R. Tezaur, *On the convergence of a dual-primal substructuring method*, Numer. Math., 88 (2001), pp. 543–558.

[18] D. Stefanica, *Choosing nonmortars: Does it influence the performance of FETI-DP algorithms?*, in Domain Decomposition Methods in Science and Engineering, Lecture Notes Comput. Sci. Eng. 40, Springer, Berlin, 2004, pp. 377–384.

[19] R. Stenberg, *Analysis of mixed finite elements methods for the Stokes problem: A unified approach*, Math. Comp., 42 (1984), pp. 9–23.

[20] B. I. Wohlmuth, *A mortar finite element method using dual spaces for the Lagrange multiplier*, SIAM J. Numer. Anal., 38 (2000), pp. 989–1012.

[21] B. I. Wohlmuth, *Discretization Methods and Iterative Solvers Based on Domain Decomposition*, Lecture Notes Comput. Sci. Eng. 17, Springer-Verlag, Berlin, 2001.

[22] J. Xu and J. Zou, *Some nonoverlapping domain decomposition methods*, SIAM Rev., 40 (1998), pp. 857–914.

SIAM J. SCI. COMPUT.
Vol. 28, No. 3, pp. 1153–1166
© 2006 Society for Industrial and Applied Mathematics

AN EXIT PROBABILITY APPROACH TO SOLVING HIGH DIMENSIONAL DIRICHLET PROBLEMS*

FABIAN M. BUCHMANN† AND WESLEY P. PETERSEN†

Abstract. We present an approach to solving high dimensional Dirichlet problems in bounded domains which is based on a variant of the Feynman–Kac formula that connects solutions to elliptic partial differential equations and functional integration. We integrate the resulting system of stochastic differential equations numerically with the Euler scheme. To correct for possible intermediate excursions of the simulated paths and to find good approximations for first exit times, we extend to higher dimensions a strategy which we introduced in [F. M. Buchmann, *J. Comput. Phys.*, 202 (2005), pp. 446–462] for stopping problems in one dimension. In addition, for Dirichlet problems, good approximations of the first exit points are needed to evaluate the boundary condition. To this end, we sample a random point on the tangent hyperplane of the boundary of the domain once a first exit is estimated. To detect excursions and adequately compute first exit times, we apply the same local half-space approximation. We therefore assume that the domain is smooth enough such that these approximations are possible. Numerical experiments in dimensions up to 128 show that resulting approximations are of very high quality. In particular, we observe that first order convergence behavior of the Euler scheme can be maintained.

Key words. stopped diffusions, stochastic differential equations, weak approximation, Dirichlet problems, Monte-Carlo simulations

AMS subject classifications. 35J25, 60G40, 60J60, 65C05

DOI. 10.1137/050622201

1. Introduction and notation.

In this work we present an approach to solve numerically Poisson's equation in high dimensional domains. Our approach is based on the stochastic representation of the solution which is given by some variant of the Feynman–Kac formula. Numerically, we have to integrate a system of stochastic differential equations (SDEs) and will approximate mathematical expectations by finite sum arithmetic means (Monte-Carlo approach). In the stochastic formulation, a Brownian path starting from a point (x) in the domain is simulated until it reaches the boundary for the first time, and the exit point must be estimated to compute the boundary contribution to an average of a functional. It is this special feature (stopping at the first exit) that makes the problem especially interesting from a numerical viewpoint. Namely, the numerical approximation of the first exit time and the first exit point is essential for good convergence properties of the resulting algorithm.

In more detail, we consider the following boundary value problem for $u : \mathbf{R}^n \to \mathbf{R}$:

$$\frac{1}{2}\triangle u(x) + g(x) = 0 \quad \text{for} \quad x \in D, \quad \text{and} \quad u(x) \to \psi(x) \quad \text{as} \quad x \to \partial D. \tag{1.1}$$

Here, D denotes a bounded domain in n-space. We will assume throughout this work that D, ∂D, the inhomogeneity g, and boundary condition ψ (both functions are $\mathbf{R}^n \to \mathbf{R}$) are such that a unique solution exists which is sufficiently smooth; see, for

*Received by the editors January 7, 2005; accepted for publication (in revised form) January 30, 2006; published electronically July 17, 2006. This article is the extended version of a contribution to the proceedings of the MC^{2}QMC conference, Juan-les-Pins, France, 2004.

http://www.siam.org/journals/sisc/28-3/62220.html

†Seminar for Applied Mathematics, Swiss Federal Institute of Technology, CH-8092 Zürich, Switzerland (fab@math.ethz.ch, wpp@math.ethz.ch). The work of the first author was supported in part under the TH-Gesuch funds 0-20981-2.

TABLE 1.1
Distributions

Symbol	Description
$\mathcal{U}(0,1)$	uniform in zero one
$\mathcal{N}(m,v)$	normal with mean m and variance $v>0$
$\mathcal{N}=\mathcal{N}(0,1)$	standard normal (mean 0, variance 1)
$\mathcal{IG}(\gamma,\delta)$	inverse Gaussian with parameters $\gamma>0$, $\delta>0$; see [8]

example, [9]. Then $u(x)$ is given by the Feynman–Kac formula

$$u(x)=\mathbb{E}\left[\psi(X(\tau^D))+\int_0^{\tau^D} g(X(s))\,\mathrm{d}s\right]. \tag{1.2}$$

In (1.2) X is an n-dimensional Brownian motion starting at $x\in D$ when $t=0$, and τ^D is the first exit time of the process $X(t)$ from D (which by continuity is also the first passage time to ∂D):

$$\tau^D=\inf_{t>0}\{X(t)\notin D\}=\inf_{t>0}\{X(t)\in\partial D\},\quad x\in D. \tag{1.3}$$

Consequently, $X(\tau^D)$ is the first exit point of the process from D (the point where the boundary ∂D is hit for the first time).

Throughout this paper we will write $\xi\sim\mathcal{D}$ if a random number ξ (or a random variable) has the distribution (or density) $\mathcal{D}$. In particular we will make use of the distributions summarized in Table 1.1.

2. Weak approximation of stopped diffusions in dimension one. We have presented an algorithm to simulate stopped diffusions (in a weak sense) in [5, 3, 4]. In order to make this article self-contained we summarize it briefly here.

Let $D=(a,b)\subset\mathbf{R}$ be a bounded interval and suppose we wish to approximate $\mathbb{E}[f(\tau)]$, where

$$f(\tau)=\int_0^{\tau} g(X(s))\,\mathrm{d}s\quad\text{for some}\quad g:\mathbf{R}\to\mathbf{R}. \tag{2.1}$$

We again assume that the function g is such that the integral in (2.1) is well defined. In (2.1), $X(t)$, $t\geq 0$, is a one-dimensional Brownian motion starting at $x\in D$ and $\tau=\inf\{t>0:X(t)\notin D\}$.

The following formulation in terms of SDEs is equivalent:

$$\mathrm{d}X(t)=\mathrm{d}W(t)\quad\text{and}\quad \mathrm{d}f(t)=g(X(t))\,\mathrm{d}t,\quad 0\leq t<\tau, \tag{2.2}$$

with initial conditions $X(0)=x$ and $f(0)=0$. A weak approximate solution of system (2.2) is now standard (see, for example, [13, 17]).

Our algorithm is based on the simplest possible scheme to numerically integrate the system (2.2), the Euler–Maruyama scheme. We simply fix a step size h and iterate (2.3) for $k=0,\ldots$ until (1.3) applies, where $X_0=x$ and $f_0=0$:

$$X_{k+1}=X_k+\sqrt{h}\xi,\quad \xi\sim\mathcal{N}(0,1), \tag{2.3a}$$

$$f_{k+1}=f_k+hg(X_k). \tag{2.3b}$$

If we apply this strategy to approximate $\mathbb{E}[f(\tau)]$ (see (2.1)), the main difficulty presents itself immediately. In (2.1), stopping time τ is not known but must be

determined by (1.3). On the other hand, we are only interested in finding a good approximation of the expectation of $f(\tau)$ (as in (1.2)). Suppose that we can approximate $f(\tau)$ and denote corresponding realizations $f^{(i)} := f^{(i)}_{K^{(i)}}$ (the number of steps changes from realization to realization and will be made precise in what follows; see Algorithm 2.1). We then approximate the expectation by a finite mean over the N realizations as

$$\mathbb{E}[f(\tau)] \approx \frac{1}{N}\sum_{i=1}^{N} f(\tau^{(i)}). \tag{2.4}$$

Our algorithm takes advantage of the fact that we only want the mean of the approximation of f to be close to $\mathbb{E}[f(\tau)]$ but do not need pathwise convergence. In other words, we need good approximations in a weak and not strong (or pathwise) sense. The better the approximate distributions are, the better the approximation of $\mathbb{E}f$ will be.

To describe our method, consider the one-dimensional case $D = (-\infty, b)$ with $x < b$. Then τ is the first hitting time of level b of the process $X(t)$. Our algorithm extends a method to approximate killed diffusions [1, 10] in a natural way: For the weak approximation of killed diffusions it suffices to know $\tau < T$ where $T > 0$ is a fixed (deterministic) value. In that case a path is killed if $\tau < T$; i.e., it does not contribute to an expectation at T if it reached level b before T. For stopped diffusions, however, an approximation of τ is needed because we need to know exactly the first hitting of b actually occurred. Algorithm 2.1 gives this approximation, and we can add a last Euler step of appropriate length (either $\mathcal{T}_1$ or $\mathcal{T}_2$) to the approximation of the integral (2.1).

Clearly, we start for every path with $X_0 = x$ and $f_0 = 0$. For $k = 0, 1, \ldots$ we proceed as described in Algorithm 2.1.

Remark 2.1. In n-dimensional cases, locally we apply a half-space approximation of D close to the boundary, which permits us to reduce our problem to the one-dimensional situation [10].

ALGORITHM 2.1

Set $y \leftarrow X_k$, generate $\xi \sim \mathcal{N}(0, h)$ and set $z \leftarrow y + \xi$. {preliminary Euler step}
if $z < b$ **then** {depending on the value of z there are two possibilities}
  Generate $u \sim \mathcal{U}(0, 1)$ and set $\mathcal{T}_1 \leftarrow -2(b - y)(b - z)/\log u$
  if $\mathcal{T}_1 \le h$ **then** {compare with the step size h: test for excursion}
    Set $f(\tau) \leftarrow f_k + \mathcal{T}_1 g(y)$ {last Euler step of length $\mathcal{T}_1$}
    stop
  else
    Set $f_{k+1} \leftarrow f_k + hg(y)$, $X_{k+1} \leftarrow z$ and **goto** item 1 {continue integrating}
  end if
else {$z \ge b$ implying $\tau \le t_{k+1}$}
  Generate $s \sim \mathcal{IG}((b - y)^2/h, (b - y)/(z - b))$ and set $\mathcal{T}_2 \leftarrow hs/(1 + s)$
  Set $f(\tau) \leftarrow f_k + \mathcal{T}_2 g(y)$ {last Euler step of length $\mathcal{T}_2$}
  stop
end if

Algorithm 2.1 is executed for N independent sample paths (the sample size) and then the mathematical expectation is approximated by the finite mean; see (2.4) (for the Monte-Carlo method, see [15] and references therein).

Having stated our method, let us summarize briefly the formula on which it is based. This will motivate our approach in higher dimensional settings (section 3); see [5] for the derivation. Let $\mathbb{P}_{y,h,z}[\cdot]$ be the law of a Brownian bridge pinned at time-space coordinates (t_k, y) and (t_{k+1}, z) (where $t_{k+1} = t_k + h$). Recall that we set $y = X_k$ and $z = X_{k+1}$ and suppose $D = (-\infty, b)$ with $x < b$; i.e.,

$$\tau = H_b(x) := \inf_{t>0} \{X_x(t) = b\} \tag{2.5}$$

is the first hitting time of level b. We then have [14] the following:

- For $y, z < b$,

$$\mathbb{P}_{y,h,z}\left[H_b(y) \le t\right] = \exp\left(-\frac{2}{t}(b-y)(b-z)\right), \quad t > 0. \tag{2.6}$$

- For $y < b < z$,

$$\begin{aligned} &\mathbb{P}_{y,h,z}[\tau \in \mathrm{d}t] \\ &= \frac{b-y}{\sqrt{2\pi}}\sqrt{\frac{h}{t^3(h-t)}}\exp\left(-\frac{(h(b-y)-t(z-y))^2}{2ht(h-t)}\right)\mathbf{1}_{\{0<t<h\}}\,\mathrm{d}t \end{aligned} \tag{2.7}$$

and after the substitution $s = t/(1-t) \ge 0$ (hence $t = s/(1+s)$)

$$= \sqrt{\frac{\gamma}{2\pi s^3}}\exp\left(-\frac{\gamma(s-\delta)^2}{2\delta^2 s}\right)\mathbf{1}_{\{s>0\}}\,\mathrm{d}s, \tag{2.8}$$

where $\gamma = (b-y)^2/h$ and $\delta = (b-y)/(z-b)$. We recall that a random variable ξ [illegible] $\gamma, \delta > 0$ if it has the density (2.8). In this case, we write $\xi \sim \mathcal{IG}(\gamma, \delta)$. We refer to [8] for a review on this distribution and to [16] for an algorithm to generate random variables $\xi \sim \mathcal{IG}(\gamma, \delta)$. This allows us (using (2.8)) to generate a random variable with density (2.7).

[illegible] 2.2. In the case that $z = b$ (which has probability zero in theory but might occur in finite-precision arithmetic on a computer) we set $\tau \approx t_{k+1}$ and hence stop the integration as $f(\tau) \approx f_k + hg(X_k)$.

3. Higher dimensions. We now turn our attention to the approximation of the solution $u(x)$ of (1.1) with $x \in D \subset \mathbf{R}^n$ with $n \ge 2$. From (1.1) we see that we need approximations of both τ and $X(\tau)$. We first describe how to extend Algorithm 2.1 presented in section 2 to higher dimensions and then show how to find $X(\tau)$.

3.1. Reduction to one dimension: Half-space approximation. We assume that for any two points $y, z \in \mathbf{R}^n$ there is a unique point on the boundary ∂D which is "closest" to y and z in some sense (we make this statement precise later on). Denote this point by X_b and suppose further that ∂D is sufficiently smooth in a sufficiently large neighborhood of X_b, such that we can approximate ∂D locally by a tangent hyperplane. For a bounded domain $D \subset \mathbf{R}^n$, every point $X_b \in \partial D$ can be mapped by a simple scaling and rotation to $(b, 0, \ldots, 0)^T$ with $b > 0$ (see section 3.4 for an efficient implementation of these rotations using Givens transformations). To derive our algorithm, we therefore assume such a transformation has been done and that near ∂D, D is given by the half-space $\{v \in \mathbf{R}^n : v^1 < b\}$ with boundary (hyperplane) $\{v \in \mathbf{R}^n : v^1 = b\}$. Suppose $x^1 < b$. We then have (compare with (2.5))

$$\tau = H_b(x) := \inf_{t>0} \left\{X_x^1(t) = b\right\}.$$

3.2. Approximation of the first exit time. For $y, z \in \mathbf{R}^n$ with $y^1, z^1 < b$ we have (compare with (2.6))

$$\mathbb{P}_{y,h,z}\left[H_b(y) \le t\right] = \exp\left(-\frac{2}{t}(b-y^1)(b-z^1)\right), \quad t > 0,$$

while for $y^1 < b$ but $z^1 > b$ it holds that (compare with (2.7))

$$\mathbb{P}_{y,h,z}[H_b(y) \in \mathrm{d}t] = \frac{b-y^1}{\sqrt{2\pi}}\sqrt{\frac{h}{t^3(h-t)}}\exp\left(-\frac{\left(h(b-y^1)-t(z^1-y^1)\right)^2}{2ht(h-t)}\right)\mathbf{1}_{\{0<t<h\}}\,\mathrm{d}t.$$

The following observation is immediate.

Remark 3.1. In the case of a boundary connected to only one of the components of an n-dimensional Brownian motion (and assuming that the boundary is locally flat, i.e., (locally) a hyperplane), the fact that these individual components are n one-dimensional Brownian motions leads to formulae as simple as in the one-dimensional setting. Furthermore, only the components connected to the boundary show up in the equations.

Therefore, extension of Algorithm 2.1 to higher dimensions is evident and we shall skip the details. Only the exit point estimation requires further elaboration.

3.3. Approximation of the exit point. In the previous section we described how to approximate the first exit time $\tau = \tau_D$ of Brownian motion from a domain D in n-space. As noted earlier, in order to approximate the solution $u(x)$ of Poisson's equation (1.1) by its stochastic representation (1.2) $u(x) = \mathbb{E}[\psi(X(\tau)) + f(\tau)]$, an approximation of the exit point $X(\tau)$ is required as soon as the boundary condition ψ is no longer a constant.

Using the notation introduced earlier, we cite the result which is the starting point for the approximation of $X(\tau)$; see [14, Lemma 7]. We recall that $\tau = \inf_{t>0}\{X^1(t) \ge b\}$. Assume $z^1 > b$. Given $X^1(s)$, $s \in [0,\tau]$, the joint distribution of $X^i(\tau)$ conditioned on $X(t) = z$ is given by [14, Lemma 6] (with $X(0) = 0$)

$$z^i\frac{\tau}{t} + \sqrt{\tau\left(1-\frac{\tau}{t}\right)}\cdot\xi^i, \quad \text{where} \quad \xi^i \sim \mathcal{N}(0,1), \quad i = 2,\dots,n. \tag{3.1}$$

The assumption that $z^1 > b$ (or, equivalently, that $z^1 \ge b$) can be replaced by the assumption that $\tau < t$ (or $\tau \le t$, respectively). We can therefore apply (3.1) as soon as the integration process is stopped (as soon as an approximation of τ is found by the extension of Algorithm 2.1 to higher dimensions using locally half-space approximations of D). The extension to the general case with $y = X_k$ and $z = X_{k+1}$ with $\tau \le t_{k+1}$ is then straightforward and yields for the resulting joint distribution of $X^i(\tau)$

$$y^i + (z^i - y^i)\frac{\tau}{h} + \sqrt{\tau\left(1-\frac{\tau}{h}\right)}\cdot\xi^i, \quad \text{where} \quad \xi^i \sim \mathcal{N}(0,1), \quad i = 2,\dots,n. \tag{3.2}$$

It remains only to incorporate the rotation of the map $X_b \mapsto (\|X_b\|, 0, \dots, 0)^T$. This rotation can be described by multiplication with an orthogonal matrix R, but we used a more efficient implementation using Givens rotations, described below. We then have $\|\tilde{X}_b^1\| = \|X_b\|$ because R is orthogonal. The exit point of the transformed

vector $\tilde{X}_b$ can be found from (3.2) by replacing $x^i \to \tilde{x}^i$ and $y^i \to \tilde{y}^i$. With this formula for the $\tilde{x}, \tilde{y}$ coordinates, we can thus [illegible] an approximation for $\tilde{X}(\tau)$ on the tangent hyperplane through $(\|X_b\|, 0, \dots, 0)^T$. Once $\tilde{X}_b$ is generated, we project back to ∂D via (3.6).

Rotating back, $(\|X_b\|, 0, \dots, 0)^T \mapsto X_b$ is easily affected by the inverse $R^{-1} = R^T$, which gives us the approximation for the exit point as $X(\tau) \approx R^{-1}\tilde{X}_b$, where this inversion is applied by $n-1$ pairwise rotations of (3.6).

ALGORITHM 3.1 Simulate one path (with index (i)) to get an approximation $\overline{u}^{(i)}$ for $u^{(i)} = \psi(X^{(i)}(\tau^{(i)})) + f^{(i)}(\tau)$ with $u(x) = \mathbb{E}[u^{(i)}]$. We assume that D is the unit n-sphere and further omit the sample index (i) for better readability.

$y \leftarrow x$, $f \leftarrow 0$ {initialize with initial conditions of SDEs}
for $k = 0, 1, 2, \dots$ **do** {integrate until τ; see Algorithm 2.1}
  $z^i = y^i + \sqrt{h}\xi$, $\xi \sim \mathcal{N}(0,1)$ for $i = 1, \dots, n$
  $X_b \leftarrow$ a point on ∂D closest to y and z
  $\tilde{y} \leftarrow Ry$ and $\tilde{z} \leftarrow Rz$ via (3.5)
  if $\tilde{z}^1 \geq 1$ **then** {D was clearly left}
    Generate $\mathcal{T}_2$ according to (2.7), (2.8) with input variables $\tilde{y}^1$, $\tilde{z}^1$ and $b = 1$.
    $f(\tau) \leftarrow f + \mathcal{T}_2 g(y)$
    stop
  else {check for an intermediate excursion}
    Generate $\mathcal{T}_1$ (see (2.6) with input variables $\tilde{y}^1$, $\tilde{z}^1$ and $b = 1$)
    if $\mathcal{T}_1 \leq h$ **then** {excursion detected}
      $f(\tau) \leftarrow f + \mathcal{T}_1 g(y)$
      stop
    else {continue integration}
      $f \leftarrow f + hg(y)$ and $y \leftarrow z$
    end if
  end if
end for
$\tilde{X}^1 \leftarrow 1$ and for $i = 2, \dots, n$ generate $\tilde{X}^i$ according to (3.2)
$\tilde{X} \leftarrow \tilde{X}/|\tilde{X}|$ {project back on ∂D}
$X(\tau) \leftarrow R^{-1}\tilde{X}$ using (3.6)
$\overline{u} \leftarrow \psi(X(\tau)) + f(\tau)$

3.4. On the implementation of the rotations. Once a suitable $X_b \in \partial D$ is found, we have to map the situation to the case where $\tilde{X}_b = (\|X_b\|, 0, \dots, 0)^T$. Then, the above operation is a rotation in n-space which can be written as $RX_b = \tilde{X}_b$ with an orthogonal $(n \times n)$-matrix R. In particular, we need $\tilde{y} = Ry$ and $\tilde{z} = Rz$ to generate $\mathcal{T}_1$ and $\mathcal{T}_2$, respectively.

Of course, one could simply calculate R and perform two matrix-vector products, but this would result in an algorithm scaling like $\mathcal{O}(n^2)$. Because these rotations have to be done in every step, this is clearly expensive computationally, especially if the dimension n is large. Also, suppose that we have some point $\tilde{u}$ in the transformed system and we additionally need the "original" point u, $u = R^T\tilde{u}$. This would be another $\mathcal{O}(n^2)$-operation.

We therefore used a procedure which has $\mathcal{O}(n)$ operations. It uses Givens rotations [11, section 3.4] and is described below. Because Givens rotations are unitary, both forward and inverse mappings are norm preserving.

To illustrate, let $X_b = (X_b^1, \ldots, X_b^n) =: b_{(1)} \in \mathbf{R}^n$ and define updated versions of X_b by $b_{(k)}$ for $k \geq 2$. Subscripts (k) indicate that the kth element of the updated $b_{(k)}$ vector is zeroed. The kth rotation to compute the update $b_{(k)} = R_{(k)} b_{(k-1)}$, for $k = 2, \ldots, n$ having $\cos, \sin$ parameters in the 1st and kth rows, uses

$$R_{(k)} = \begin{pmatrix} \cos(\varphi_{(k)}) & & & -\sin(\varphi_{(k)}) & & \\ & 1 & & & & \\ & & \ddots & & & \\ & & & 1 & & \\ \sin(\varphi_{(k)}) & & & \cos(\varphi_{(k)}) & & \\ & & & & 1 & \\ & & & & & \ddots \\ & & & & & & 1 \end{pmatrix} \tag{3.3}$$

with

$$\cos(\varphi_{(k)}) = \frac{b_{(k-1)}^1}{w_{(k-1)}} \quad \text{and} \quad \sin(\varphi_{(k)}) = \frac{-b_{(k-1)}^k}{w_{(k-1)}}$$

and

$$w_{(k-1)} = ((b_{(k-1)}^1)^2 + (b_{(k-1)}^k)^2)^{\frac{1}{2}}. \tag{3.4}$$

Proceeding in the same way, after $n-1$ such rotations, we have

$$b_{(n)} = R_{(n)} R_{(n-1)} \cdots R_{(2)} X_b. \tag{3.5}$$

It is easy to see that each $R_{(k)}$ in (3.3) is unitary, i.e., $R_{(k)}^T R_{(k)} = \mathbb{I} = R_{(k)} R_{(k)}^T$, and thus the inverse transformation is performed in the reverse order:

$$R^{-1} = R^T = R_{(2)}^T R_{(3)}^T \cdots R_{(n)}^T. \tag{3.6}$$

The rotations $\tilde{y} = Ry$ and $\tilde{z} = Rz$, and likewise their inverses, can be computed simultaneously.

The calculated trigonometric tables $\cos(\varphi_{(i)}), \sin(\varphi_{(i)})$ from (3.5) are stored in two arrays of length $n-1$ for the connected inverse transformation ($R^{-1} = R^T$).

The procedure used in our experiments used Givens rotations, but Householder transformations [7] would do as well. Namely, X_b may be projected into direction $e_1 = (1, 0, \ldots, 0)^T$ by a rank-one update H:

$$HX_b = \begin{pmatrix} \|X_b\| \\ 0 \\ \vdots \\ 0 \end{pmatrix},$$

where

$$H = \mathbb{I} - 2\frac{vv^T}{\|v\|^2} \tag{3.7}$$

and vector $v = X_b - \|X_b\| e_1$. The Householder transformation is also unitary; in fact, $H = H^T$. This transformation, HX_b, again requires $\mathcal{O}(n)$ operations. Givens rotations have a slight advantage over distributed memory machines (such as our Beowulf) because inner products, which require global communications to get $\|v\|$ and $\|X_b\|$, are not required.

4. Numerical experiments. We show results from extensive tests performed with the method developed in the previous sections. We first show results from experiments performed when the boundary condition ψ is constant (without loss of generality we took $\psi \equiv 0$). Then, sampling of the exit point is unnecessary and we can concentrate on the approximation of $\mathbb{E}[\tau]$. We next show results from experiments with nonconstant boundary conditions, hence sampling $X(\tau)$ on the boundary.

4.1. Constant boundary condition. We concentrate on the [illegible], i.e., Poisson's equation (1.1) with $g \equiv 1$ and $\psi \equiv 0$. Then $u(x) = \mathbb{E}[\tau^D(x)]$ is the mean first exit time. Note that for this problem numerical integration using the Euler scheme is exact and any errors in numerical simulation (other than the statistical error) are due to the stopping at the first exit.

4.1.1. Two-dimensional square. Our first test is such that no error arising due to the (local) approximation of ∂D by its tangent hyperplane at the point X_b is present. The flatter ∂D is, the smaller this error will be. We therefore start with a simple two-dimensional domain without local curvature, i.e., a square.

We set $D = (-1,1) \times (-1,1)$ and consider $\triangle u = -2$ with a zero valued boundary condition. Expanding $u(x,y)$ in eigenfunctions, $\cos(\frac{2k+1}{2}\pi x)\cos(\frac{2l+1}{2}\pi y)$, we find

$$u(0) = \frac{128}{\pi^4} \sum_{k,l=0}^{\infty} \frac{(-1)^{k+l}}{(2k+1)(2l+1)\left((2k+1)^2+(2l+1)^2\right)} \approx 0.5893708\ldots.$$

to compare with our numerical approximations.

When computing the closest boundary point, we do not take into account more than one boundary segment for an excursion test in each step, even if two boundaries are equally far away:

If $z \in D$, we simply take (one of) the boundary segment(s) that minimize(s) the sum of the euclidean distance of y and z to ∂D.

If $z \notin D \cup \partial D$, we assume that $(X(t))_t$ crosses the boundary ∂D at the same point as a linear interpolated version of the discrete path $(X_k)_k$ would.

In doing so, we introduce an additional error. We neglect this error and do not try to correct for it; see [1] for a discussion and possible solutions to this problem. From our results it will be apparent that for sufficiently small h, these errors can indeed be neglected: Asymptotically we get (at least) the desired $\mathcal{O}(h)$-behavior of the error (see Figure 4.1 and the conclusions thereafter).

In this first test, we compare the following methods:

(T): This method stops only if $z \notin D$. In this case, it approximates $\tau \approx t_k$.

(K): This method applies a [illegible] test and stops always (if $z \notin D$ or if an intermediate excursion is detected) at t_k. In other words, this method is similar to ours, but it ignores the last integration step of length $\mathcal{T}_1$ or $\mathcal{T}_2$, respectively.

(SK): This method is a mixture of methods (K) and (S): In the case of an excursion, integration is stopped at $t_k + \mathcal{T}_1$ and at t_k if $z \notin D$ (the random variable $\mathcal{T}_2$ is not generated).

(S): This is the advocated method which samples both $\mathcal{T}_1$ and $\mathcal{T}_2$.

From the results shown in Figure 4.1 we draw the following conclusions:

- The trivial stopping procedure as applied by method (T) is clearly not satisfying. Integration stops much too late on average (as the true solution is approached from above; see the left-hand plot) and resulting convergence order is only $\frac{1}{2}$.

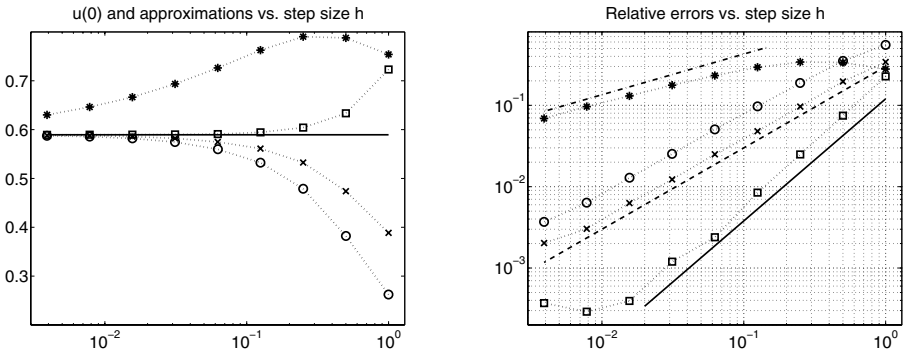


FIG. 4.1. *Results for two-dimensional square with $g \equiv 1$ and $\psi \equiv 0$. On the left, approximate solutions are shown versus step size h together with the exact solution (which is indicated by a solid line). On the right, corresponding relative errors are shown versus step size h. Three lines are added: a dash-dotted line with slope $\frac{1}{2}$, a dashed line with slope 1, and a solid line with slope $\frac{3}{2}$. In both plots, $\cdots * \cdots$ is method (T), $\cdots \circ \cdots$ is method (K), $\cdots \times \cdots$ is method (SK), and $\cdots \square \cdots$ is method (S).*

- Applying some sort of a posteriori test to detect intermediate excursions remedies the main difficulties. Convergence of all three methods (K), (SK), and (S) is much better.
- Method (K) stops integration too early on average (it always stops at the beginning of a time step). The same holds for method (SK), although applying a stopping test instead of a killing test to detect intermediate excursions reduces corresponding errors by approximately a factor of two.
- Method (S) is clearly superior to all the other methods tested: The approximation of $u(0)$ is very good, and connected errors converge with an order higher than 1 (approximately $\frac{3}{2}$).
- Method (S) stops integration for large step sizes (as did method (T)) too late on average. We speculate, that this is due to the fact that we always test only for an excursion across the boundary, even in situations when two boundary segments are close. This is, for example, the case when y and z are close to a corner.

4.1.2. Unit *n*-spheres. We next show simulations for unit n-spheres (unit n-balls); i.e., we set $D = B_1^n = \{x \in \mathbf{R}^n : |x| < 1\}$. The point on ∂D was chosen as follows (we denote it by X_b):

$z \in \partial D$: We set $X_b = z$.

$z \in D$: We set X_b to be the minimizer of the sum of the square of the distances to the boundary. For the unit n-sphere this yields $X_b \leftarrow (y+z)/|y+z|$, which is computationally very efficient and produces for h small enough adequate choices.

$z \notin D \cup \partial D$: We set X_b to the point where the line connecting y and z crosses ∂D.

We show results obtained with method (S) when applied to the exit problem in various dimensions. Results for dimensions $n = 2, 4, 8, 16$ are in Figure 4.2, and those for $n = 64, 128$ are in Figure 4.3. Sample size was always $N = 4 \times 10^6$, and confidence intervals were always smaller than the plot symbols.

From the results in Figures 4.2 and 4.3 we see that for the exit problem in unit n-spheres, asymptotically a convergence order of 1 can always be observed. This is

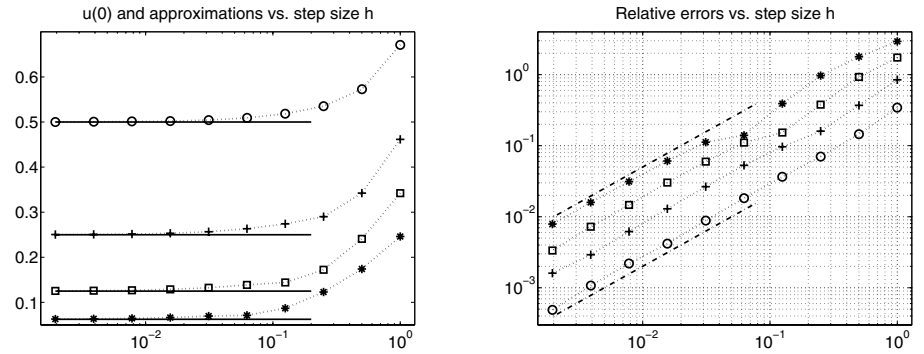


FIG. 4.2. *Results for the exit problem in unit n-spheres. On the left, approximate solutions are shown versus step size h together with corresponding exact solutions (which are indicated by a solid line). On the right, corresponding relative errors are shown versus step size h. Two dashed lines with slope* 1 *are added. In both plots,* $\cdots\circ\cdots$ *is* $n = 2$, $\cdots+\cdots$ *is* $n = 4$, $\cdots\square\cdots$ *is* $n = 8$, *and* $\cdots*\cdots$ *is* $n = 16$.

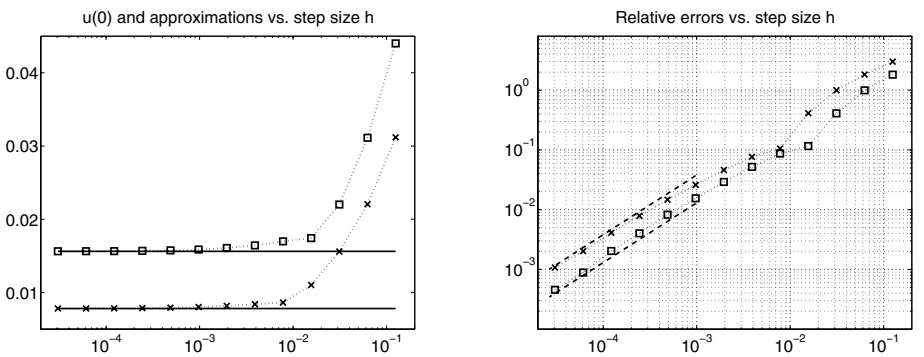


FIG. 4.3. *Results for the exit problem in unit n-spheres. On the left, approximate solutions are shown versus step size h together with corresponding exact solutions (which are indicated by a solid line). On the right, corresponding relative errors are shown versus step size h. Two dashed lines with slope* 1 *are added. In both plots,* $\cdots\square\cdots$ *is* $n = 64$ *and* $\cdots\times\cdots$ *is* $n = 128$.

encouraging, as it shows that the local approximation of ∂D by a tangent hyperplane seems to work, independent of the dimension. Note, however, that as dimension n is increased, a smaller step size h is needed to observe first order convergence.

4.2. Nonconstant boundary conditions. We now show some experiments with $n = 32$ dimensional unit spheres involving three different problems with one variant of the second problem. Explicit solutions are given in (4.1), (4.2a), (4.2b), and (4.3), respectively.

4.2.1. Constant inhomogeneity, linear boundary condition.

$$g(x) \equiv 1 \quad \text{and} \quad \psi(x) = \sum_{i=1}^{32} x^i \Longrightarrow u(x) = \frac{1}{32}\left(1 - |x|^2\right) + \sum_{i=1}^{32} x^i. \tag{4.1}$$

For this problem, integration is still exact (as again $g(x) \equiv 1$), but in contrast to the [illegible], an approximation for $X(\tau)$ is required in order to evaluate the boundary condition for every path as soon as $x \neq 0$. The boundary condition does not vary significantly and is linear.

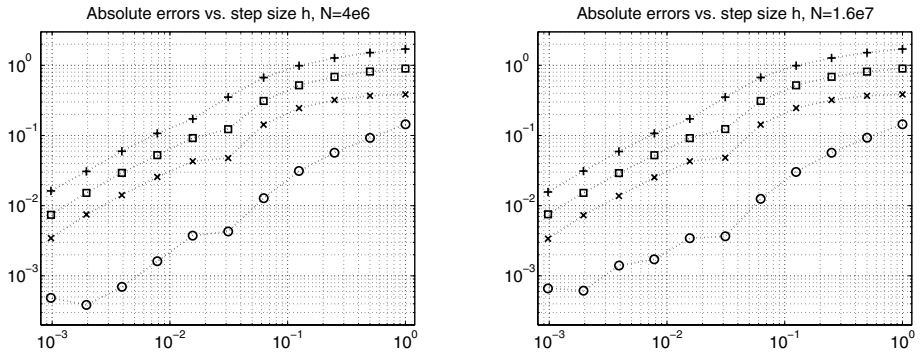


FIG. 4.4. *Results for test problem* (4.1). *Absolute errors versus step size h for two different sample sizes* $N = 4 \times 10^6$ *on the left and* $N = 1.6 \times 10^7$ *on the right. The solution is evaluated at* $x = (x^i)$ *with* $x^i = \alpha/32$ *for* $i = 1, \ldots, 32$. *In both plots,* $\cdots \circ \cdots$ *is* $\alpha = 0$, $\cdots \times \cdots$ *is* $\alpha = 1$, $\cdots \square \cdots$ *is* $\alpha = 2$, *and* $\cdots + \cdots$ *is* $\alpha = 4$.

For this problem we show results for $u(x)$ with $x = (\alpha/32)_{i=1,\ldots,32}$ with $\alpha = 0, 1, 2, 4$. Note that for higher α, x is closer to the boundary ∂D and therefore the probability that a point $X_b \in \partial D$ is the first exit point of a path starting at x is no longer uniformly distributed on ∂D (as is the case for $x = (0, \ldots, 0)^T$). To better distinguish the individual results for different values of α, we show plots of the absolute error; see Figure 4.4.

4.2.2. Quadratic inhomogeneity, quartic boundary condition. We make $g(x)$ harder to integrate and choose a nonlinear boundary condition. We show results of two variants of this problem:

$$g(x) = -\sum_{i=1}^{32} \left(x^i\right)^2 \quad \text{and} \quad \psi(x) = \frac{1}{6} \sum_{i=1}^{32} \left(x^i\right)^4 \implies u(x) = \psi(x). \tag{4.2a}$$

This problem still has an intrinsic symmetry: $g(\cdot)$ is constant on the spheres $|x| \equiv \|x\|_2 = r = \text{const}$, and $\psi(\cdot)$ is constant when $\|x\|_4^4 \equiv \sum_i (x^i)^4 = R = \text{const}$, respectively. We therefore consider the following extension:

$$g(x) = -\sum_{i=1}^{32} i \left(x^i\right)^2 \quad \text{and} \quad \psi(x) = \frac{1}{6} \sum_{i=1}^{32} i \left(x^i\right)^4 \implies u(x) = \psi(x). \tag{4.2b}$$

We show relative errors obtained versus step size h in Figure 4.5.

4.2.3. General (transcendental) inhomogeneity and boundary condition. The last example we use to test Algorithm 3.1 is defined as follows: We set for $k^i = 1, 2, \ldots$, with $k = (k^i)_{i=1,\ldots,32}$,

$$g(x) = \prod_{i=1}^{32} \cos(2\pi k^i x^i) \quad \text{and} \quad \psi(x) = \frac{g(x)}{2\pi^2 |k|^2}. \tag{4.3}$$

Then $u(x) = \psi(x)$, $x \in \partial D$. Results for $x^i = 0$, 5/100, and 1/10 are shown in Figures 4.6, 4.7, and 4.8, respectively.

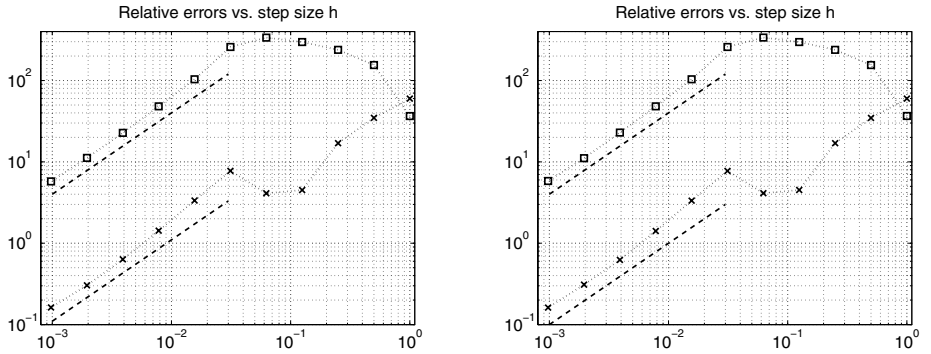


FIG. 4.5. *Results for test problems* (4.2a) *on the left and* (4.2b) *on the right. Shown are relative errors versus step size* h *for sample size* $N = 4 \times 10^6$. *The solution is evaluated at* $x = (x^i)$ *with* $x^i = 5/100$ *and* $x^i = 1/10$ *for* $i = 1, \ldots, 32$. *In both plots,* $\cdots \square \cdots$ *is* $x^i = 5/100$ *and* $\cdots \times \cdots$ *is* $x^i = 1/10$, *and two dashed lines with slope* 1 *are added.*

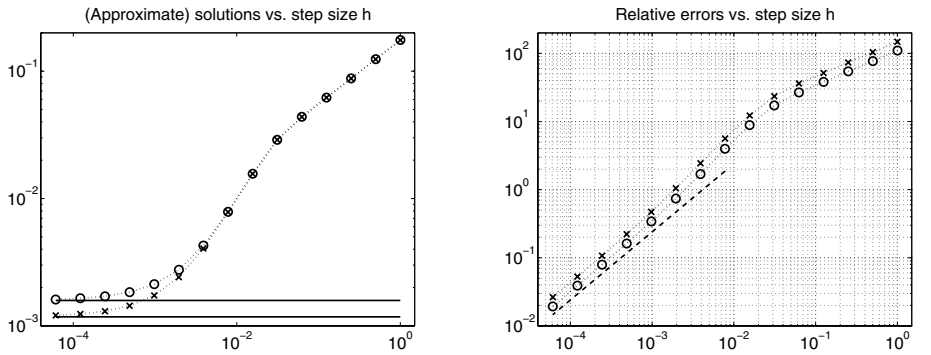


FIG. 4.6. *Results for test problem* (4.3) *when the solution is evaluated at* $x = (x^i)$ *with* $x^i = 0$ *for* $i = 1, \ldots, 32$ *for* $N = 10^6$ *paths. On the left, we show the approximate solutions versus step size* h *and indicate the exact solution with a solid line. On the right, we plot corresponding errors versus step size* h *and add a dashed line with slope* 1. *In both plots,* $\cdots \circ \cdots$ *is the result obtained for* $k = (1, \ldots, 1)$, *while* $\cdots \times \cdots$ *are those for* $k = (3, 2, 1, 1, \ldots, 1)$. *Compare with* $x^i = 5/100$ *in Figure* 4.7 *and* $x^i = 1/10$ *in Figure* 4.8.

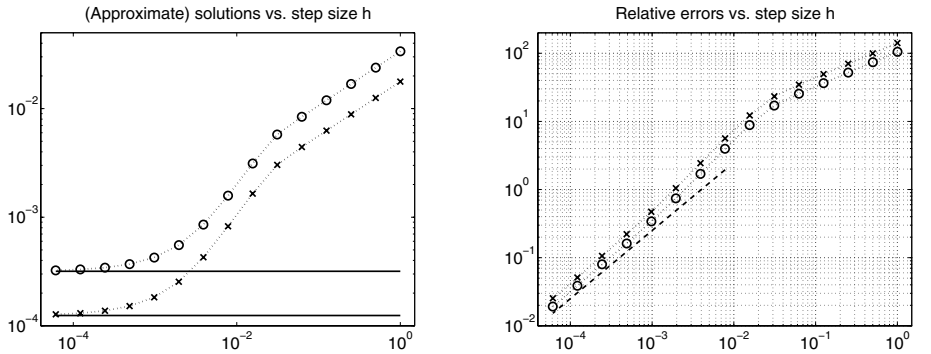


FIG. 4.7. *Results for test problem* (4.3) *when the solution is evaluated at* $x = (x^i)$ *with* $x^i = 5/100$ *for* $i = 1, \ldots, 32$ *for* $N = 10^6$ *paths. See Figures* 4.6 *and* 4.8 *for comparisons with* $x^i = 0$ *and* $x^i = 1/10$, *respectively.*

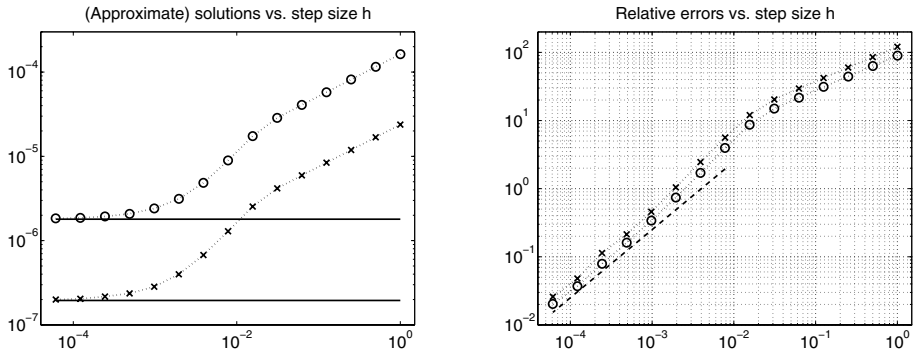


FIG. 4.8. *Results for test problem* (4.3) *when the solution is evaluated at* $x = (x^i)$ *with* $x^i = 1/10$ *for* $i = 1, \ldots, 32$ *for* $N = 10^6$ *paths. See Figures* 4.6 *and* 4.7 *to compare the results for* $x^i = 0$ *and* $x^i = 5/100$*, respectively.*

4.2.4. Discussion of the numerical results for 32-dimensional spheres. Problem (4.1) results show that there is some absolute error increase as the initial point moves out from the center. This is not surprising since the solution (4.1) increases approximately linearly as $|x| = r$ (the initial radius) increases. Thus the relative error is approximately constant in this range. Plots in Figure 4.5 of the relative error for problems (4.2a) and (4.2b) confirm this assessment. Additionally, the expected $\mathcal{O}(h)$-accuracy of our Euler method is evident.

5. Conclusions and final remarks. We have outlined a general procedure for estimating exit times and exit points for Monte-Carlo solutions of elliptic partial differential equations in n-dimensional spaces. Since the Feynman–Kac representation requires good estimates for these statistics, an accurate sampling method seems essential. In our procedure, two situations are evident: , , $X_k \in D$ and $X_{k+1} \in D$ both hold, but possible excursions are estimated as in (2.6); and , , $X_k \in D$ but $X_{k+1} \notin D$. In the latter situation, the sample path has clearly exited domain D; thus an estimate of τ^D, $t_k < \tau^D < t_{k+1}$, is required. Exit point $X(\tau^D)$ must be sampled only if an excursion was determined in the first case, and always in the second case (definite exit). For this exit point, a bridge process (3.2) is used. First, a transformation to a local tangent hyperplane coordinate system is made. This transformation makes it easier to estimate the exit point. This transformed exit point is then rotated back to the original coordinates to get $X(\tau^D)$. These local hyperplane $\leftrightarrow$ original coordinate transformations can be done with Givens rotations (3.5) or Householder reflections (3.7) in $\mathcal{O}(n)$ arithmetic operations.

Another procedure for estimating exit times is given in Jansons and Lythe [12]. Their method uses exponential timestepping and has the advantage that in one dimension, spatial sampling/step is a two-sided exponential distribution (see, for example, [2, equations (1.0.2) and (1.0.5), p. 153]). Tails of exponential distributions are obviously easier to sample than are normal distributions. In higher dimensions ($n > 1$), their situation is more complicated and yields a density proportional to $K_{n/2}(h^{-1/2}|x|)$, where $K_{n/2}$ is a modified Bessel function. It would be very interesting to see how their elegant procedure for estimating the exit times could be expanded to include sampling exit distributions. The walk-on-spheres method of Milstein and Tretyakov (see chapter 6 of [18]) provides another alternative. This procedure has an advantage over simple bounded increment methods [4, 6] in that the increments

are intrinsically isotropic in n-space, although the 3-point approach [4, 6] can be made isotropic with random orthogonal rotations [19]. However, their walk-on-spheres method becomes a 2-point formula in one dimension and appears to be of lower accuracy in our experiments [6].

What remains to be done, particularly for problems similar to our test in section 4.1.1, is a better analysis of corners or cusps. That will be a subject of future work, but data in Figure 4.1 show encouraging results by a crude procedure which simply chooses the nearest side, at least for near-center values of x.

REFERENCES

[1] P. Baldi, *Exact asymptotics for the probability of exit from a domain and applications to simulation*, Ann. Probab., 23 (1995), pp. 1644–1670.

[2] A. N. Borodin and P. Salminen, *Handbook of Brownian Motion—Facts and Formulae*, Birkhäuser-Verlag, Basel, Switzerland, 2002.

[3] F. M. Buchmann, *Computing Exit Times with the Euler Scheme*, Technical report 2003–02, Seminar for Applied Mathematics, Swiss Federal Institute of Technology, Zürich, Switzerland, 2003; also available online from http://e-collection.ethbib.ethz.ch/show?type=incoll&nr=877.

[4] F. M. Buchmann, *Solving High Dimensional Dirichlet Problems Numerically Using the Feynman-Kac Representation*, Ph.D. thesis, Swiss Federal Institute of Technology, Zürich, Switzerland, 2004.

[5] F. M. Buchmann, *Simulation of stopped diffusions*, J. Comput. Phys., 202 (2005), pp. 446–462.

[6] F. M. Buchmann and W. P. Petersen, *Solving Dirichlet problems numerically using the Feynman–Kac representation*, BIT, 43 (2003), pp. 519–540.

[7] J. W. Demmel, *Applied Numerical Linear Algebra*, SIAM, Philadelphia, PA, 1997.

[8] J. L. Folks and R. S. Chhikara, *The inverse Gaussian distribution and its statistical application—A review*, J. Roy. Statist. Soc. Ser. B, 40 (1978), pp. 263–289.

[9] M. Freidlin, *Functional Integration and Partial Differential Equations*, Ann. of Math. Stud. 109, Princeton University Press, Princeton, NJ, 1985.

[10] E. Gobet, *Weak approximation of killed diffusion using Euler schemes*, Stochastic Process. Appl., 87 (2000), pp. 167–197.

[11] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Johns Hopkins Ser. Math. Sci. 3, The Johns Hopkins University Press, Baltimore, MD, 1983.

[12] K. M. Jansons and G. D. Lythe, *Exponential timestepping with boundary test for stochastic differential equations*, SIAM J. Sci. Comput., (2003), pp. 1809–1822.

[13] P. E. Kloeden and E. Platen, *Numerical Solution of Stochastic Differential Equations*, Appl. Math. 23 (New York), Springer-Verlag, Berlin, 1992.

[14] H. R. Lerche and D. Siegmund, *Approximate exit probabilities for a Brownian bridge on a short time interval, and applications*, Adv. in Appl. Probab., 21 (1989), pp. 1–19.

[15] N. Madras, *Lectures on Monte Carlo Methods*, Fields Inst. Monogr. 16, AMS, Providence, RI, 2002.

[16] J. R. Michael, W. R. Schucany, and R. W. Haas, *Generating random variates using transformations with multiple roots*, Amer. Statist., 30 (1976), pp. 88–90.

[17] G. N. Milstein, *Numerical Integration of Stochastic Differential Equations*, Math. Appl. 313, Kluwer Academic Publishers, Dordrecht, The Netherlands, 1995.

[18] G. N. Milstein and M. V. Tretyakov, *Stochastic Numerics for Mathematical Physics*, Springer-Verlag, Berlin, 2004.

[19] G. W. Stewart, *The efficient generation of random orthogonal matrices with an application to condition estimators*, SIAM J. Numer. Anal., 17 (1980), pp. 403–409.

SIAM J. SCI. COMPUT.
Vol. 28, No. 3, pp. 1167–1185
© 2006 Society for Industrial and Applied Mathematics

NUMERICAL METHODS FOR DIFFERENTIAL EQUATIONS IN RANDOM DOMAINS*

DONGBIN XIU† AND DANIEL M. TARTAKOVSKY‡

Abstract. Physical phenomena in domains with rough boundaries play an important role in a variety of applications. Often the topology of such boundaries cannot be accurately described in all of its relevant detail due to either insufficient data or measurement errors or both. This topological uncertainty can be efficiently handled by treating rough boundaries as random fields, so that an underlying physical phenomenon is described by deterministic or stochastic differential equations in random domains. To deal with this class of problems, we propose a novel computational framework, which is based on using stochastic mappings to transform the original *deterministic/stochastic problem in a random domain* into a *stochastic problem in a deterministic domain*. The latter problem has been studied more extensively, and existing analytical/numerical techniques can be readily applied. In this paper, we employ both a stochastic Galerkin method and Monte Carlo simulations to solve the transformed stochastic problem. We demonstrate our approach by applying it to an elliptic problem in single- and double-connected random domains, and comment on the accuracy and convergence of the numerical methods.

Key words. random domain, stochastic inputs, differential equations, uncertainty quantification

AMS subject classifications. 60H15, 60H35, 65C20, 65C30

DOI. 10.1137/040613160

1. Introduction.

Physical phenomena described by differential equations in domains with rough geometries are ubiquitous and include applications ranging from surface imaging [24] to manufacturing of nano-devices [5]. Indeed, given a proper spatial resolution, virtually any natural or manufactured surface becomes rough. Consequently, there is a growing interest in experimental, theoretical, and numerical studies of deterministic and probabilistic descriptions of such surfaces and of solutions of differential equations defined on the resulting domains.

The early attempts to represent surface roughness and to study its effects on system behavior were based on simplified, easily parameterizable, [illegible] surface inhomogeneities, such as symmetrical asperities to represent indentations [32, 26] and semispheres to represent protrusions [17]. More general representations of surface roughness, which nevertheless admit simple parameterizations, include sinusoidal corrugations [13] and periodic linear segments [6]. A recent trend in describing surface roughness is to use fractal and fractal-like representation. These include deterministic Von Koch's [10, 6, 8] and Minkowski's [6] fractals, as well as their random generalizations [29].

*Received by the editors August 9, 2004; accepted for publication (in revised form) February 7, 2006; published electronically July 17, 2006.

http://www.siam.org/journals/sisc/28-3/61316.html

†Department of Mathematics, Purdue University, West Lafayette, IN 47907 (dxiu@math.purdue.edu).

‡Department of Mechanical and Aerospace Engineering, the University of California, San Diego, La Jolla, CA 92093, and Theoretical Division, Los Alamos National Laboratory, Los Alamos, NM 87545 (dmt@ucsd.edu). The work of this author was performed at Los Alamos National Laboratory (LA-UR 04-5497) under the auspices of the National Nuclear Security Agency of the U.S. Department of Energy under contract W-7405-ENG-36 and was supported in part by the U.S. Department of Energy under the DOE/BES Program in the Applied Mathematical Sciences, contract KC-07-01-01, and in part by the LDRD Program at Los Alamos National Laboratory.

Alternatively, one can use [illegible] fields to represent rough surfaces whose detailed topology cannot be ascertained due to the lack of sufficient information and/or measurement errors [15, 27]. In this paper, we adopt random representations of rough surfaces, because of their generality. Such an approach allows us not only to make predictions of the system behavior, but also to quantify the corresponding predictive uncertainties.

The presence of uncertainty in rough boundaries necessitates the development of new approaches for the analysis and numerical solution of differential equations defined on random domains. For example, classical variational formulations might not be suitable for such problems [1], and a finite difference approach [34] is shown to be accurate only for relatively simple rectangular irregularities. Most of the existing analyses deal with deterministic roughness and are very problem-specific. For example, the use of conformal mappings [7, 6] requires that a physical phenomenon be accurately described by Laplace's equation in two dimensions, while the use of the Liebmann method [8] requires a particular geometry configuration and Dirichlet boundary conditions on the rough surface.

Adoption of a probabilistic framework to describe rough surfaces renders even essentially deterministic problems stochastic; e.g., deterministic equations in random domains give rise to stochastic boundary-value problems. This necessitates the search for new stochastic analyses and algorithms. For example, rigorous bounds for elliptic problems on random domains have been derived for both Dirichlet [2] and Neumann [1] boundary conditions, and perturbation solutions of Laplace's equation in a domain bounded by a random fractal boundary with a small surface dimension have been obtained [29]. One of the very few existing numerical studies employed traditional Monte Carlo simulations [6], which turned out to be so computationally expensive as to become impractical.

In this paper, we present a computational framework that is applicable to a wide class of deterministic and stochastic differential equations defined on domains with random (rough) boundaries. A key component of this framework is the use of robust stochastic mappings to transform an original deterministic or stochastic differential equation defined on a [illegible] into a stochastic differential equation defined on a [illegible]. This allows us to employ the well-developed theoretical and numerical techniques for solving stochastic differential equations in deterministic domains. The idea of transforming irregular domains into their regular counterparts has been used in computational fluid dynamics [33] and optimization problems [4], among other applications. The proposed approach extends this idea to stochastic problems.

In section 2, we provide a general mathematical formulation of a random domain problem. A stochastic mapping technique, which is defined by a solution of an appropriate boundary value problem for Laplace's equation, is presented in section 3.2. A mathematical formulation of the transformed stochastic equations in deterministic domains is given in section 3.3. This section also contains the description of two numerical approaches, Monte Carlo simulations (section 3.3.1) and a stochastic Galerkin method (section 3.3.2), whose relative strengths are discussed in section 3.3.3. In section 4, we apply our computational approach to a two-dimensional elliptic problem defined on both single- and double-connected random domains. For moderately high random dimensions (up to 16-dimensional random space), the stochastic Galerkin method is shown to be more efficient than Monte Carlo simulations. We conclude by listing a few open issues in section 5.

2. Differential equations in random domains. Let $\omega \in \Omega$ be a random realization drawn from a complete probability space $(\Omega, \mathcal{A}, \mathcal{P})$, whose event space Ω generates its σ-algebra $\mathcal{A} \subset 2^{\Omega}$ and is characterized by a probability measure $\mathcal{P}$. For all $\omega \in \Omega$, let $D(\omega) \subset \mathbb{R}^d$ be a d-dimensional random domain ($d = 1, 2, 3$), bounded by boundary $\partial D(\omega)$, some or all of whose segments are random. We consider the following stochastic (linear or nonlinear) boundary value problem: for $\mathcal{P}$-almost everywhere (a.e.) in Ω, given $a : D(\omega) \to \mathbb{R}$ and $b : \partial D(\omega) \to \mathbb{R}$, find a stochastic solution $v : \bar{D}(\omega) \to \mathbb{R}$ such that

$$\begin{aligned} L(x; v) &= a(x) \quad \text{in } D(\omega), \\ B(x; v) &= b(x) \quad \text{on } \partial D(\omega). \end{aligned} \tag{2.1}$$

Here $x = (x_1, \ldots, x_d)$ is the coordinate in $\mathbb{R}^d$, L is a differential operator, and B is a boundary operator. The operator B can take various forms on different boundary segments, e.g., $B \equiv I$, where I is the identity operator, on Dirichlet segments and $B \equiv \mathbf{n} \cdot \nabla$ on Neumann segments whose outward unit normal vector is $\mathbf{n}$. The operators L and B, as well as the driving terms a and b, can be either deterministic or random. To simplify the presentation, we assume that the shape of the boundary $\partial D(\omega)$ is the only source of randomness. This is done without loss of generality, since incorporation of other sources of randomness is straightforward, and stochastic equations in deterministic domains have been studied extensively. Finally, we assume that the random boundary $\partial D(\omega)$ is sufficiently regular, and the boundary condition in (2.1) is properly posed, to guarantee the well-posedness a.e. $\omega \in \Omega$ of the stochastic boundary-value problem (SBVP) (2.1). We refer to this problem as a random domain problem or RDP.

Except for a few studies mentioned in section 1, and despite their practical and theoretical importance, RDPs have not been systematically analyzed. At the same time, the theory of stochastic partial differential equations in fixed deterministic domains is relatively mature. To take advantage of the existing methods for deterministic domains, we introduce a one-to-one mapping function and its inverse

$$\xi = \xi(x, \omega), \qquad x = x(\xi, \omega), \tag{2.2}$$

which transforms the random domain $D(\omega) \subset \mathbb{R}^d$ into a deterministic domain $E \subset \mathbb{R}^d$, whose coordinates are denoted as $\xi = (\xi_1, \ldots, \xi_d)$; i.e., for $\mathcal{P}$-a.e. $\omega \in \Omega$,

$$x \in D(\omega) \longleftrightarrow \xi \in E. \tag{2.3}$$

The stochastic mapping (2.3) transforms the deterministic differential operators L and B into their stochastic counterparts $\mathcal{L}$ and $\mathcal{B}$, respectively, and RDP (2.1) into the following SBVP. For $\mathcal{P}$-a.e. $\omega \in \Omega$, given $f : E \times \Omega \to \mathbb{R}$ and $g : \partial E \times \Omega \to \mathbb{R}$, find a stochastic solution $u : \bar{E} \times \Omega \to \mathbb{R}$ such that

$$\begin{aligned} \mathcal{L}(\xi, \omega; u) &= f(\xi, \omega) \quad \text{in } E, \\ \mathcal{B}(\xi, \omega; u) &= g(\xi, \omega) \quad \text{on } \partial E, \end{aligned} \tag{2.4}$$

where f and g are the transformed functions of a and b, respectively. Note that since the random mapping (2.3) depends on random $D(\omega)$, other possible sources of randomness, including coefficients of the differential operators in (2.1) and boundary and initial conditions, can be easily incorporated into the formulation of RDP (2.1) and do not change the following analysis.

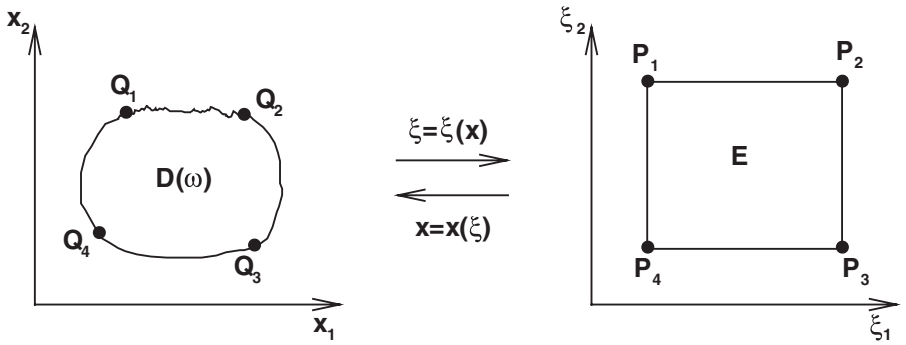


FIG. 2.1. *A schematic representation of a computational domain $D(\omega)$ with the random boundary segment $\widehat{Q_1Q_2}$ and its mapping onto a deterministic rectangular domain E, where four reference points Q_i are mapped onto P_i, $i = 1, 2, 3, 4$.*

The geometry of the deterministic domain E is to a large extent arbitrary and should be chosen in a way that simplifies the subsequent analysis and computation. Consider, for example, the domain $D(\omega)$ in Figure 2.1, whose boundary segment $\widehat{Q_1Q_2}(\omega)$ is random. Possible choices of the corresponding deterministic domain E include domains whose deterministic boundary segments remain unchanged, and the random boundary segment $\widehat{Q_1Q_2}(\omega)$ is replaced with (i) its ensemble mean $\mathbb{E}[\widehat{Q_1Q_2}(\omega)]$, where $\mathbb{E}$ denotes the expectation operator in the probability space, and (ii) its upper or lower bounds. (Depending on the problem, one can define such bounds based on either the total volume of $D(\omega)$ or the distance of $\partial D(\omega)$ to a certain location in the domain of interest, or in some other way.) Here, to facilitate numerical simulations in the transformed domain E, we choose it to have a canonical shape, such as the rectangle $\overline{P_1P_2P_3P_4}$ shown in Figure 2.1.

3. Computational framework. A computational framework that is applicable to a wide range of practical applications should provide (i) a general parameterization of the random boundary $\partial D(\omega)$, (ii) an effective way to obtain the stochastic mapping (2.2), and (iii) an efficient method for solving the subsequent SBVP (2.4) in deterministic domains. We construct such a framework by introducing a stochastic mapping based on solutions of Laplace's equations (section 3.2), and by employing two complementary numerical algorithms to solve the resulting SBVPs: a Monte Carlo method and a stochastic Galerkin method (section 3.3).

While the approach discussed in this section is applicable to one- and three-dimensional problems as well, here we consider the two-dimensional domain shown in Figure 2.1, to simplify the presentation. Without loss of generality, we assume that only the segment $\widehat{Q_1Q_2}$ of the overall boundary $\partial D(\omega)$ is random, and we represent it as $\widehat{Q_1Q_2}(\omega) = r(z, \omega)\mathbf{e}_1 + s(z, \omega)\mathbf{e}_2$, where $\mathbf{e}_1$ and $\mathbf{e}_2$ are unit vectors in the x_1 and x_2 directions, respectively; r and s are the parameterization functions for the segment; and z is the parameterization variable. We seek to map $D(\omega)$ onto a square $E = [0, 1] \times [0, 1]$.

3.1. Parameterization of random boundary. Parameterization of the random boundary $\widehat{Q_1Q_2}$ consists of the following steps. First, we use Reynolds decomposition to represent random functions, e.g., $A(\omega) = \langle A \rangle + A'(\omega)$, as the sums of

their ensemble means $\langle A\rangle \equiv \mathbb{E}[A(\omega)]$ and zero-mean fluctuations $A'(\omega)$. The latter term can be characterized by a variety of techniques, including its representation as a random field with a given point distribution and two-point correlation function. Here, following common practice, we approximate the fluctuation term by a finite number $K \geq 1$ of mutually uncorrelated random variables $Y_1(\omega), \ldots, Y_K(\omega)$ with zero mean and unit variance, so that $r(z,\omega)$ and $s(z,\omega)$ in (3.4) can be approximated as

$$r(z,\omega) \approx \langle r(z)\rangle + \sum_{k=1}^{K} \hat{r}_k(z) Y_k(\omega) \tag{3.1}$$

and

$$s(z,\omega) \approx \langle s(z)\rangle + \sum_{k=1}^{K} \hat{s}_k(z) Y_k(\omega). \tag{3.2}$$

The parameterization (3.1) and (3.2) is an approximation of the true processes, whose accuracy and robustness are the subject of ongoing research in the field of numerical generation of random processes. If these processes are non-Gaussian, this task becomes particularly challenging [16, 28, 30, 39]. (In such cases, the goal is often reduced to approximating pointwise marginal distribution functions and two-point covariance functions.) Analysis of the errors induced by the finite-term representations in (3.1) and (3.2), as well as of their efficiency, is beyond the scope of this paper. Instead we refer the interested reader to the references mentioned above.

The choice of a parameterization of the random boundary $\widehat{Q_1Q_2}$ defines the expansion coefficients $\{\hat{r}_k\}$ and $\{\hat{s}_k\}$. One popular choice is the Karhunen–Loève (KL) decomposition [23], which is an optimal decomposition in terms of the mean-square approximation error and has been used extensively to represent random inputs [3, 14, 36]. Other types of decomposition also can be employed. In section 4 of this paper, we explore both the KL expansion and a Fourier expansion.

3.2. Mapping of the random domain. We construct the stochastic mapping of $D(\omega)$ onto E via solutions of the Laplace equations

$$\frac{\partial^2 x_1}{\partial \xi_1^2} + \frac{\partial^2 x_1}{\partial \xi_2^2} = 0, \qquad \frac{\partial^2 x_2}{\partial \xi_1^2} + \frac{\partial^2 x_2}{\partial \xi_2^2} = 0 \quad \text{in } E, \tag{3.3}$$

subject to the boundary conditions

$$x_1(0,\xi_2) = x_1|_{\widehat{Q_4Q_1}}, \qquad x_1(1,\xi_2) = x_1|_{\widehat{Q_3Q_2}}, \tag{3.4a}$$

$$x_1(\xi_1,0) = x_1|_{\widehat{Q_4Q_3}}, \qquad x_1(\xi_1,1) = r(z,\omega) \tag{3.4b}$$

and

$$x_2(0,\xi_2) = x_2|_{\widehat{Q_4Q_1}}, \qquad x_2(1,\xi_2) = x_2|_{\widehat{Q_3Q_2}}, \tag{3.4c}$$

$$x_2(\xi_1,0) = x_2|_{\widehat{Q_4Q_3}}, \qquad x_2(\xi_1,1) = s(z,\omega), \tag{3.4d}$$

where $x_i|_{\widehat{Q_mQ_n}}$ denotes the x_i coordinate along the boundary segment $\widehat{Q_mQ_n}$. Similar mappings have been used extensively to solve numerically deterministic problems in structured body-fitted curvilinear coordinates [33], and more recently to analyze the interface dynamics in disordered media [19]. The properties and a more detailed

technical description of such mappings can be found in [33]. The stochastic mapping (3.3)–(3.4) can be viewed as a generalization of this idea to random domains. We also remark that the boundary conditions (3.4) are not expressed in a definitive manner. One can choose different distributions of boundary coordinates in x as boundary conditions in (3.4), in order to achieve better computational results by, e.g., clustering a mesh close to the regions of interests (see [33] for details).

As a result of (3.1) and (3.2), solutions of (3.3) can be expressed as

$$x_i(\xi,\omega) = \sum_{k=0}^{K} \hat{x}_{i,k} Y_k(\omega), \qquad i = 1, 2. \tag{3.5}$$

Then, denoting $\hat{r}_0 \equiv \langle r \rangle$ and $\hat{s}_0 \equiv \langle s \rangle$, substituting the expansions (3.1) and (3.5) into (3.3)–(3.4), and collecting the terms of each $\{Y_k(\omega)\}$, i.e., conducting a Galerkin projection (in the random space) of the resulting equations onto the basis spanned by $\{1, Y_1(\omega), \ldots, Y_K(\omega)\}$, we obtain $(K+1)$ equations for the expansion coefficients,

$$\frac{\partial^2 \hat{x}_{1,k}}{\partial \xi_1^2} + \frac{\partial^2 \hat{x}_{1,k}}{\partial \xi_2^2} = 0, \qquad \frac{\partial^2 \hat{x}_{2,k}}{\partial \xi_1^2} + \frac{\partial^2 \hat{x}_{2,k}}{\partial \xi_2^2} = 0, \quad k = 0, \ldots, K, \tag{3.6}$$

subject to the boundary conditions

$$\hat{x}_{1,k}(0,\xi_2) = x_1|_{\widehat{Q_4Q_1}} \cdot \delta_{k0}, \qquad \hat{x}_{1,k}(1,\xi_2) = x_1|_{\widehat{Q_3Q_2}} \cdot \delta_{k0}, \tag{3.7a}$$

$$\hat{x}_{1,k}(\xi_1,0) = x_1|_{\widehat{Q_4Q_3}} \cdot \delta_{k0}, \qquad \hat{x}_{1,k}(\xi_1,1) = \hat{r}_k(z) \tag{3.7b}$$

and

$$\hat{x}_{2,k}(0,\xi_2) = x_2|_{\widehat{Q_4Q_1}} \cdot \delta_{k0}, \qquad \hat{x}_{2,k}(1,\xi_2) = x_2|_{\widehat{Q_3Q_2}} \cdot \delta_{k0}, \tag{3.7c}$$

$$\hat{x}_{2,k}(\xi_1,0) = x_2|_{\widehat{Q_4Q_3}} \cdot \delta_{k0}, \qquad \hat{x}_{2,k}(\xi_1,1) = \hat{s}_k(z), \tag{3.7d}$$

where $\delta_{mn} = 0$ if $m \neq n$ and $\delta_{mn} = 1$ if $m = n$ is the Kronecker delta function. Solutions of the deterministic boundary value problems (3.6)–(3.7) give the stochastic mapping function $x_i(\xi,\omega)$ in (3.5), whose metrics terms and Jacobian can be readily evaluated.

3.3. Numerical solutions of the transformed equations. Since the stochastic mapping $x_i(\xi,\omega)$ in (2.2) is represented by (3.5) as a function of the random variables $\{Y_k(\omega)\}_{k=1}^K$, the metrics terms and the Jacobian corresponding to the coordinate transformation from x to ξ become functions of $\{Y_k(\omega)\}_{k=1}^K$ as well. In particular, the Jacobian takes the form

$$J(\xi,\omega) \equiv \frac{\partial(\xi_1,\ldots,\xi_d)}{\partial(x_1,\ldots,x_d)} = J(\xi, Y_1(\omega), \ldots, Y_K(\omega)). \tag{3.8}$$

Hence, according to the Doob–Dynkin lemma [25], a solution of the SBVP (2.4) can be described by a finite number of random variables $\{Y_k(\omega)\}_{k=1}^K$, i.e.,

$$u(x,\omega) = u(x, Y_1(\omega), \ldots, Y_K(\omega)). \tag{3.9}$$

The transformed SBVP (2.4) in the fixed domain E can be solved by a variety of techniques, including perturbation methods [18] and second-moment analyses [22]. In this paper, we use two complementary approaches, Monte Carlo simulations and stochastic Galerkin methods.

To facilitate numerical implementation, we further assume that the random variables $\{Y_k(\omega)\}_{k=1}^K$ are mutually independent, rather than just uncorrelated. (Such an assumption is not needed for Gaussian processes, for which independence and uncorrelation are equivalent.) While this assumption might introduce additional errors in the finite-term approximations (3.1) and (3.2) of target random processes, it enables one to define the corresponding Hilbert functional spaces via simple tensor product rules. In other words, the independence assumption is made for technical convenience, and has been employed in many applications in stochastic simulations, e.g., [3, 21, 36] and references therein. We also remark that similar assumptions are often made in sampling methods, where random number generators typically generate [illegible], not just uncorrelated, series of random numbers. It is worthwhile to point out that it is possible to construct multidimensional functional spaces based on a finite number of [illegible] random variables [31]. However, in its current form such a construction is not amenable to a straightforward numerical implementation.

3.3.1. Monte Carlo simulations. Monte Carlo simulations (MCS) are one of the most developed methods for solving stochastic differential equations. Although their convergence rate is relatively slow—a typical Monte Carlo simulation consisting of M realizations converges as $1/\sqrt{M}$ (see [11])—it is independent from the number of random variables $\{Y_k(\omega)\}_{k=1}^K$. We remark that the presence of random boundaries $\partial D(\omega)$ makes the direct use of MCS for solving the RDP (2.1) less than straightforward. Since each realization of the random domain $D(\omega_j), j = 1, \ldots, M$, is different, a remeshing procedure is needed to render the statistical averaging of a solution at any point x in the domain meaningful.

In our approach, we apply MCS to the transformed SBVP (2.4) instead. Since this SBVP is defined on the fixed deterministic domain E, the use of MCS becomes straightforward and consists of the following steps:

1. for a given number of realizations M, generate the independent random variables $\{Y_k(\omega_j)\}_{k=1}^K$, for $j = 1, \ldots, M$;
2. for each $j = 1, \ldots, M$, use (3.1) and (3.2) to construct realizations of the random boundary $\widehat{Q_1Q_2}(\omega_j) = r(\omega_j)\mathbf{e}_1 + s(\omega_j)\mathbf{e}_2$, and solve Laplace's equations (3.6)–(3.7) to obtain realizations of the stochastic mapping $x_i(\xi, \omega_j), i = 1, \ldots, d$;
3. for each $j = 1, \ldots, M$, evaluate the metrics terms and the Jacobian $J(\xi, \omega_j)$, and solve the SBVP (2.4) in the deterministic domain $\xi \in E \subset \mathbb{R}^d$;
4. postprocess the results to evaluate the solution's statistics, e.g., $\langle u(\xi) \rangle \equiv \mathbb{E}[u] = M^{-1} \sum_{j=1}^M u(\xi, \omega_j)$.

For each realization $j = 1, \ldots, M$, step 3 solves a deterministic problem on a deterministic domain E, which can be accomplished with any suitable spatial discretization scheme.

3.3.2. Stochastic Galerkin method. Stochastic Galerkin (SG) methods are polynomial-type expansions in the random space. In general, these methods exhibit fast convergence rates, provided that solutions are sufficiently smooth in the random space. The first SG methods employed polynomial chaos expansions [14], which are based on the Wiener–Hermite expansion. More recent developments of the generalized polynomial chaos employ non-Hermite polynomials to improve efficiency for a wide class of problems. Such generalizations include global polynomial expansions based on hypergeometrical polynomials [37, 36, 38], piecewise polynomial expansions [3, 9], and wavelet basis expansions [20, 21]. Here we briefly review the framework based on global polynomial expansions.

Let $\{Y_i\}_{i=1}^K$ be independent random variables with probability density functions $\rho_i : \Gamma_i \to \mathbb{R}^+$, and assume that their images $\Gamma_i \equiv Y_i(\Omega)$ are bounded intervals in $\mathbb{R}$ for $i = 1, \ldots, K$. Then

$$\rho(y) = \prod_{i=1}^{K} \rho_i(Y_i) \qquad \forall y \in \Gamma \tag{3.10}$$

is the joint probability density of $y = (Y_1, \ldots, Y_K)$ with the support

$$\Gamma \equiv \prod_{i=1}^{K} \Gamma_i \subset \mathbb{R}^K. \tag{3.11}$$

This allows one to rewrite (2.4) as a $(d + K)$-dimensional differential equation

$$\begin{aligned} \mathcal{L}(\xi, y; u) &= f(\xi, y), \quad (\xi, y) \in E \times \Gamma, \\ \mathcal{B}(\xi, y; u) &= g(\xi, y), \quad (\xi, y) \in \partial E \times \Gamma. \end{aligned} \tag{3.12}$$

Let us define a one-dimensional polynomial space in $L^2_{\rho_i}(\Gamma_i)$, the space for all square integrable functions in Γ_i with respect to measure $\rho_i dY_i$,

$$W_i^{p_i} \equiv \{v : \Gamma_i \to \mathbb{R} : v \in \operatorname{span}\{\phi_m(Y_i)\}_{m=0}^{p_i}\}, \qquad i = 1, \ldots, K, \tag{3.13}$$

where $\{\phi_m(Y_i)\}$ form a set of orthogonal polynomials satisfying the orthogonality conditions

$$\int_{\Gamma_i} \rho_i(Y_i)\phi_m(Y_i)\phi_n(Y_i)dY_i = h_m^2 \delta_{mn}, \tag{3.14}$$

with normalization factors $h_m^2 = \int_{\Gamma_i} \rho_i(Y_i)\phi_m^2(Y_i)dY_i$. Then, the corresponding K-dimensional polynomial space is defined by

$$W_{K,p} \equiv \bigotimes_{|\mathbf{p}| \le p} W_i^{p_i}, \tag{3.15}$$

where the tensor product is over all possible combinations of the multi-index $\mathbf{p} = (p_1, \ldots, p_K) \in \mathbb{N}_0^K$ such that the length of the multi-index satisfies $|\mathbf{p}| = \sum_{k=1}^K p_k \le p$. The total number of basis functions in (3.15) is $(K + p)!/(K!p!)$.

The type of orthogonal polynomials $\{\phi_m(Y_i)\}$ in (3.14) is determined by the probability density function of $Y_i(\omega)$, for $i = 1, \ldots, K$. For example, uniform distributions are best approximated with Legendre polynomials, while Hermite polynomials are most appropriate for Gaussian distributions. A detailed list of such correspondences and the rates of their convergence can be found in [37, 36].

Finally, an SG formulation of (3.12) is as follows. For all $\xi \in \bar{E}$, find $u_p(\xi, y) \in W_{K,p}$ such that

$$\int_\Gamma \rho(y)\mathcal{L}(\xi, y; u_p)\, v(y)dy = \int_\Gamma \rho(y) f(\xi, y)v(y)d\mathbf{y} \quad \forall v(y) \in W_{K,p},\ \xi \in E, \tag{3.16}$$

$$\int_\Gamma \rho(y)\mathcal{B}(\xi; u_p)v(y)dy = \int_\Gamma \rho(y) g(\xi, y)v(y)dy \quad \forall v(y) \in W_{K,p},\ \xi \in \partial E. \tag{3.17}$$

The weak formulation (3.16)–(3.17) results in a set of deterministic problems in the fixed domain E and can be further discretized by any standard discretization technique, including finite elements and finite volumes. Construction of stochastic Sobolev spaces for stochastic diffusion equations is discussed in detail in [3]. For more implementation details and numerical examples, see [37, 36].

3.3.3. Selection of a numerical method. The dimensionality of the problem (3.12) is $d+K$, although only d dimensions involve differentiations. The total number K of random variables $\{Y_i\}_{i=1}^K$ that is needed to represent the random boundary $\partial D(\omega)$ by (3.1) and (3.2) with a given accuracy depends on, among other factors, the correlation length of $\partial D(\omega)$. Although one may choose different decomposition methods in (3.1) and (3.2), in general, the value of K increases when the correlation length becomes smaller.

SG methods converge with increased order of p. In fact, an exponential convergence has been proved for stochastic diffusion equations in [3] and shown numerically for various stochastic equations in [36, 37, 38]. However, when $K \gg 1$ is very large, the total number of expansion terms $(K+p)!/K!p!$ increases quickly, and this effectively reduces the convergence rate with respect to the number of expansion terms. In this case, it may be necessary to resort to MCS, as their convergence rate, $1/\sqrt{M}$, albeit slower, is asymptotically independent of the value of K.

Thus, for low to moderate values of K (less than 10), SG methods are preferred, and for large values of $K \gg 1$, the Monte Carlo method is preferred. However, if high accuracy of a stochastic solution is required, then SG methods can be more efficient even for relatively large values of K. (For example, see [35] for computations with $K = 38$ for a three-dimensional time-dependent heat transfer problem.)

4. Applications to elliptic equations in random domains. In this section, we apply the computational framework described above to elliptic problems in single- and double-connected random domains.

4.1. Formulation. Consider a deterministic Poisson equation with homogeneous Dirichlet boundary conditions in a random domain: for $\mathcal{P}$-a.e. $\omega \in \Omega$, $c, a : D(\omega) \to \mathbb{R}$, find $v : \bar{D}(\omega) \to \mathbb{R}$ such that

$$\begin{aligned} \nabla \cdot [c(x)\nabla v(x,\omega)] &= a(x) \quad \text{in } D(\omega), \\ v(x,\omega) &= 0 \quad \text{on } \partial D(\omega). \end{aligned} \tag{4.1}$$

The stochastic mapping (2.2) transforms (4.1) into a stochastic Poisson equation on a deterministic domain E: for $\mathcal{P}$-a.e. $\omega \in \Omega$, $\kappa, f, \alpha_{ij} : E \times \Omega \to \mathbb{R}, 1 \le i, j \le d$, find $u : \bar{E} \times \Omega \to \mathbb{R}$ such that

$$\sum_{i=1}^{d} \frac{\partial}{\partial \xi_i} \left[\kappa(\xi,\omega) \sum_{j=1}^{d} \left(\alpha_{ij}(\xi,\omega) \frac{\partial u}{\partial \xi_j} \right) \right] = J^{-1} f(\xi,\omega) \quad \text{in } E, \tag{4.2}$$

$$u(\xi,\omega) = 0 \qquad \text{on } \partial E,$$

where the random fields κ and f are the transformations of c and a, respectively; J is the transformation Jacobian defined in (3.8); and

$$\alpha_{ij}(\xi,\omega) = J^{-1} \nabla \xi_i \cdot \nabla \xi_j, \qquad 1 \le i, j \le d. \tag{4.3}$$

The use of (3.1) and (3.2) to parameterize the random boundary $\partial D(\omega)$ with a finite number of random variables $\{Y_i(\omega)\}_{i=1}^K$ yields a strong form of (4.2),

$$\sum_{i=1}^{d} \frac{\partial}{\partial \xi_i} \left[\kappa(\xi,y) \sum_{j=1}^{d} \left(\alpha_{ij}(\xi,y) \frac{\partial u}{\partial \xi_j} \right) \right] = J^{-1} f(\xi,y) \qquad \text{in } E \times \Gamma, \tag{4.4}$$

subject to the boundary condition

$$u(\xi, y) = 0 \qquad \text{on } \partial E \times \Gamma. \tag{4.5}$$

In what follows, we solve the SBVP (4.4)–(4.5) by MCS and SG methods. Note that the left-hand side of (4.5) typically takes the form of a complicated function of the random vector y, even though the representation of the input, i.e., random boundary ∂D, is linear in y. Thus, in SG methods the projection of (4.5) onto the basis functions can be nontrivial. In this paper, we employ a typical approach that uses the quadrature/cubature rule with sufficient accuracy to ensure the accuracy of the projection (e.g., [37]).

4.2. Diffusion in a channel with a rough surface. Consider steady-state diffusion, described by (4.1) with $f \equiv 0$, in a two-dimensional channel $D(\omega) = \{(x_1, x_2) | 0 \le x_1 \le L_x, s(x_1, \omega) \le x_2 \le L_y\}$. To be specific, we set $L_x = 5$, $L_y = 1$, and treat the rough bottom boundary as a random field $s = s(x_1, \omega)$ with zero mean $\langle s(x_1, \omega) \rangle = 0$ and an exponential two-point covariance function

$$C_S(x_1, z_1) = \mathbb{E}\left[s(x_1, \omega) s(z_1, \omega)\right] = \exp\left(-\frac{|x_1 - z_1|}{b}\right), \tag{4.6}$$

where $b > 0$ is the correlation length. In the computational examples below, $b = L_x/5 = 1$, which corresponds to a boundary of moderate roughness. Finally, we prescribe Dirichlet boundary conditions $u = 1$ at $x_2 = L_y$ and $u = 0$ elsewhere.

We employ the finite-term KL-type expansion (3.2) to decompose the boundary process. The expansion coefficients $\{\hat{s}_k\}_{k=1}^K$ are given by $\hat{s}_k(x_1) = \sqrt{\lambda_k}\psi_k(x_1)$, where $\{\lambda_k, \psi_k(x_1)\}$ are the eigenvalues and eigenfunctions of the integral equations

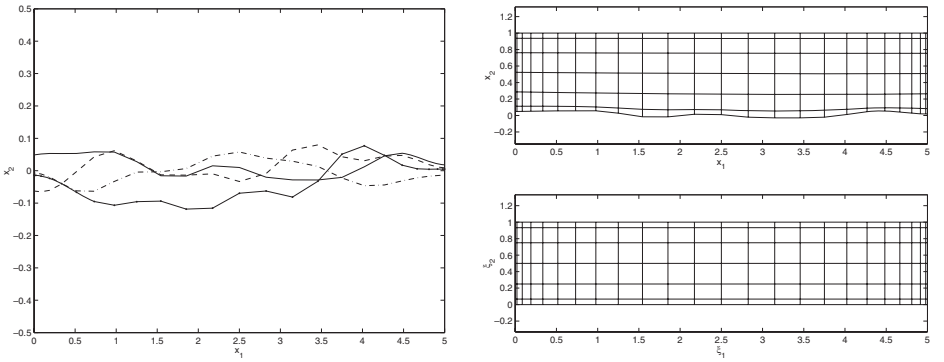
$$\int C_S(x_1, z_1)\psi_k(z_1) dz_1 = \lambda_k \psi_k(x_1), \qquad k = 1, \dots, K. \tag{4.7}$$

Then the decomposition (3.2) becomes

$$s(x_1, \omega) \approx \sigma \sum_{k=1}^{K} \hat{s}_k(x_1) Y_k(\omega) = \sigma \sum_{k=1}^{K} \sqrt{\lambda_k}\psi_k(x_1) Y_k(\omega). \tag{4.8}$$

We set $\{Y_i(\omega)\} \sim U(-1, 1)$ to be independent uniform random variables in $(-1, 1)$ and use the parameter $0 < \sigma < 1$ to control the maximum deviation of the rough surface. (In the computational examples in this section, we set $\sigma = 0.1$.) We employ Legendre polynomials as the basis functions (3.13) in the SG method, since they form the optimal basis corresponding to uniform random inputs [37].

It is worthwhile to stress again that the expansion (4.8) introduces two sources of errors—the errors due to the finite K-term truncation and the errors due to the assumption of independence of $\{Y_k(\omega)\}$. The truncation error is typically controlled by selecting the value of K to ensure that the eigenvalues $\{\lambda_k\}$ with $k > K$ are sufficiently small. For example, our numerical tests reveal that the expansion (4.8) with $K = 10$ captures 95% of the L_2 norm. Hence, we employ a ten-term KL expansion to represent the random boundary, and show some realizations of the boundary in Figure 4.1(a). Theoretical error estimates of such truncations can be found in [12]. Error analysis on the independence assumption is lacking and lies outside the scope of the present study.



(a) (b)

FIG. 4.1. *Channels with a rough wall generated with the ten-term ($K = 10$) KL expansion* (4.8). (a) *Four sample realizations of the bottom boundary $s(x_1, \omega_j)$ ($j = 1, \ldots, 4$).* (b) *A sample realization of the channel in the physical domain (x_1, x_2) and in the mapped domain in (ξ_1, ξ_2). Chebyshev meshes are used in both domains.*

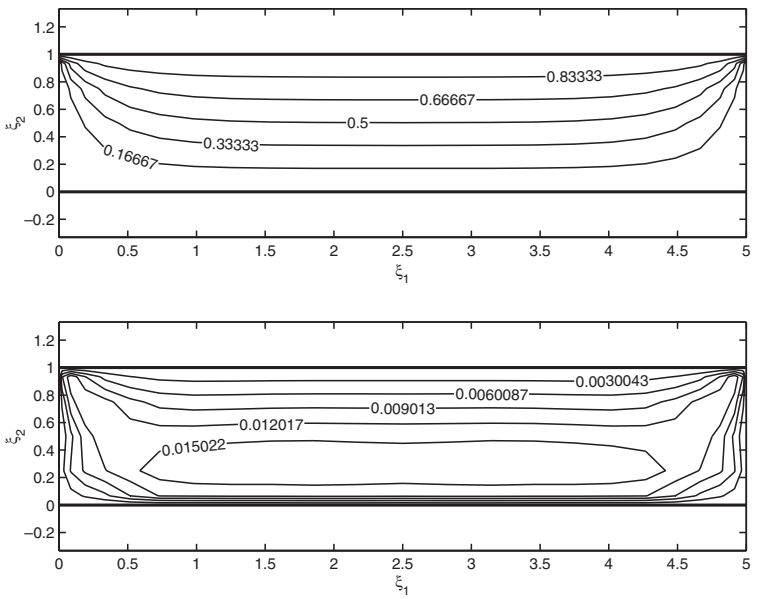


FIG. 4.2. *The mean and STD of the dependent variable u, computed with the SG method.*

Figure 4.1(b) shows a realization of the channel $D(\omega_j)$ mapped onto the corresponding rectangular domain $E = [0, L_x] \times [0, L_y]$. The mapping Laplace equations (3.3) are solved by the Chebyshev collocation method. Figure 4.1(b) also shows the Chebyshev collocation mesh points in both the physical domain D and the mapped domain E.

The stochastic elliptic problem (4.2) is solved in the mapped rectangular domain E with the SG method described in section 3.3.2. The Chebyshev collocation method is used in both ξ_1 and ξ_2 directions. Figure 4.2 shows the first two ensemble moments of the solution, i.e., its mean (top) and standard deviation (STD) (bottom). We

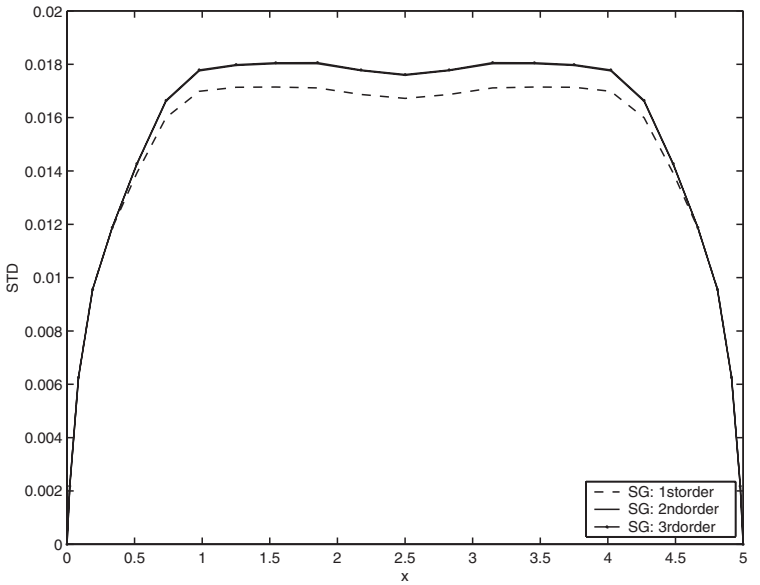


FIG. 4.3. *The STD profiles along the cross section $y = 0.25$, computed with the first-, second-, and third-degree Legendre polynomials.*

observe that the STD reaches its maximum close to, but not at, the random bottom boundary.

To ascertain the convergence of the polynomial chaos expansion, we examine the STD profile along the cross section $y = 0.25$, where the STD is close to its maximum. Figure 4.3 shows the STD profiles obtained with different orders of Legendre expansions. Once can see that the second order is sufficient for the Legendre expansion to converge. Although not shown here, the convergence of the mean solution is similar to that of the STD.

MCS are also conducted to verify the results obtained by the SG method. Figure 4.4 compares the STD profile along the cross section $y = 0.25$ computed via the second-order Legendre expansion with those obtained from MCS. We observe that as the number of realizations increases, the MCS results converge to the converged SG results. With about 2,000 realizations, the MCS results agree well with SG results. In this case, at second order ($p = 2$) and ten random dimensions ($K = 10$), the SG method requires $(K + p)!/K!p! = 66$ basis functions and is computationally more efficient than MCS. In CPU time, the second-order SG method is approximately 20 times faster than MCS with 2,000 realizations.

4.3. Diffusion in double-connected domains with rough exclusion. Consider steady-state diffusion, described by (4.1) with $f \equiv 0$, in a double-connected domain $D(\omega)$ consisting of the square $[-2, 2] \times [-2, 2]$ with a randomly perturbed unit circular exclusion. A sample realization of such a domain is shown in Figure 4.5(a), and several sample realizations of the surface of the random exclusion are depicted in Figure 4.5(b). The boundary conditions are $u = 1$ along the surface of the exclusion and $u = 0$ along the outer square.

The inner random boundary is constructed and parameterized by superimposing a zero-mean random perturbation process on the radius $R(\theta)$ of a unit circle centered

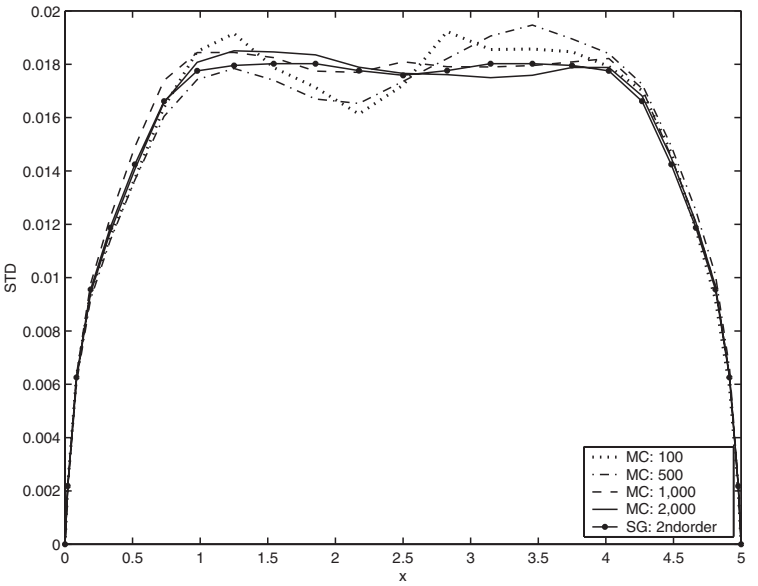
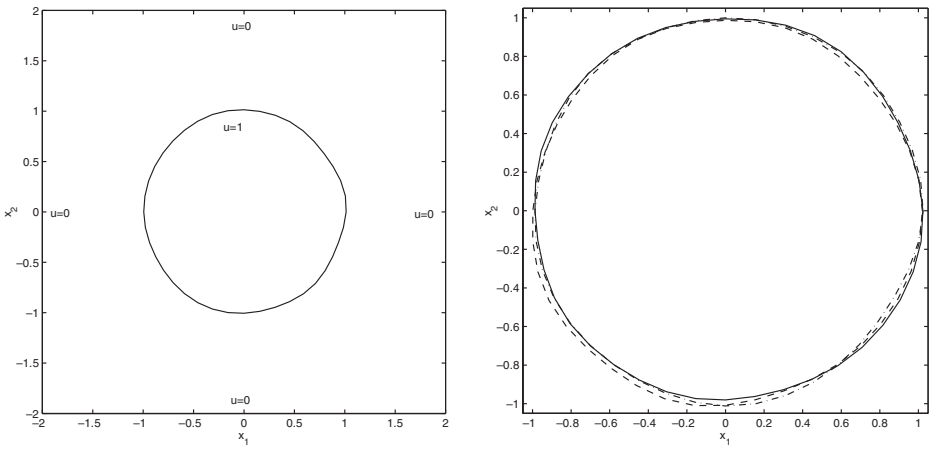


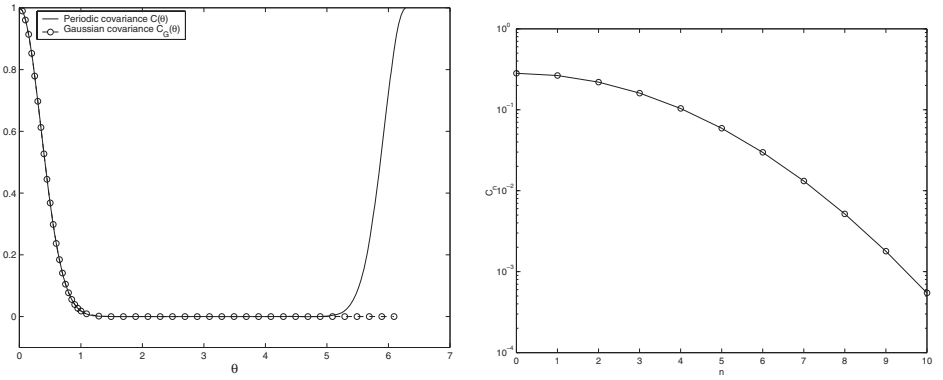
FIG. 4.4. *The STD profiles along the cross section* $y = 0.25$*, computed with the SG method (the second-order Legendre chaos) and the MCS consisting of* 100*,* 500*,* 1,000*, and* 2,000 *realizations.*



(a) (b)

FIG. 4.5. *A double-connected domain with a rough circular exclusion generated with the* 16*-term* $(N = 16)$ *Fourier expansion* (4.9). (a) *A sample realization of the random computational domain.* (b) *Three sample realizations of the circular exclusion* $R(\theta, \omega_j)$ $(j = 1, \ldots, 4)$.

at $(0, 0)$, where $0 \leq \theta \leq 2\pi$. In other words, the inner boundary is defined as $(x_1, x_2) = (R\cos\theta, R\sin\theta)$, for $0 \leq \theta \leq 2\pi$, where $R(\theta, \omega) = 1 + \sigma R'(\theta, \omega)$ is the radius. Here $0 < \sigma < 1$ is used to control the maximum deviation of the process. We require that both $\langle R' \rangle = 0$ and the statistics of $R(\theta, \omega)$ be [illegible] on the circle, i.e., that $\langle R'(\theta_1) R'(\theta_2) \rangle = C_R(|\theta_1 - \theta_2|)$.



(a) (b)

FIG. 4.6. (a) *Periodic covariance function $C(\theta)$ (solid line) based on the nonperiodic Gaussian function $C_G = \exp(-\theta^2/b^2)$ with $b = 0.5$ (dashed line with circles).* (b) *Decay of the Fourier cosine coefficients of the periodic covariance function.*

Since the random process $R'(\theta, \omega)$ is periodic, we use a Fourier-type expansion

$$R'(\theta, \omega) = \sum_{n=-N}^{N} R_n(\omega) e^{-in\theta} \tag{4.9}$$

for its decomposition into a finite number of random variables (see (3.1) and (3.2)). In this expansion, the coefficients $R_n(\omega) = R_n^r(\omega) + iR_n^i(\omega)$ are complex random variables. It is straightforward to show that if R_n^r and R_n^i are statistically independent for all n, and have zero mean and the variance of $C_n/4$, where $C_n = \frac{1}{\pi}\int_0^{2\pi} C_R \cos(n\theta) d\theta$ are the coefficients of the Fourier cosine series of the covariance function C_R, then the random field $R'(\theta, \omega)$ given by (4.9) has the prescribed covariance function C_R. (It is worth noting that in periodic domains, Fourier expansion is in fact the KL expansion.)

A periodic covariance function $C_R(|\theta_1 - \theta_2|)$ with the period of 2π is constructed by extending the standard Gaussian covariance function $C_G = \exp(-(\theta_1 - \theta_2)^2/b^2)$ to the periodic domain $(0, 2\pi)$, where b is the correlation length. Figure 4.6(a) shows such a periodic covariance function and contrasts it with the standard nonperiodic Gaussian covariance function. Figure 4.6(b) demonstrates the decay of the Fourier cosine coefficients $\{C_n\}$ for the periodic covariance function with the correlation length $b = 0.5$. Based on the decay of C_n, we choose $N = 8$ ($C_9 = 0.0052, C_1 = 0.2821, C_9/C_1 < 2\%$).

In the following examples, we set the coefficients $R_n^r, R_n^i, n = 1, \ldots, N$, in the expansion (4.9) to be independent uniform random variables of $U(-1, 1)$. This results in a 16-dimensional ($K = 2N = 16$) random space Γ for problem (4.2). In the computational examples below, we set $\sigma = 0.02$. We again use Legendre expansion in the SG method to represent the uniform random inputs.

To map the double-connected random domain $D(\omega)$ onto a single-connected deterministic domain E, we introduce an artificial cut in the domain D, as shown in Figure 4.7. This cut dissects the overall boundary into four segments: the inner boundary $\widehat{AB}$, the cut $\overline{BC}$, the outer boundary $\widehat{CD}$, and the cut $\overline{DA}$. Note that the segments $\overline{BC}$ and $\overline{AD}$ are at the same physical locations, although while solving the

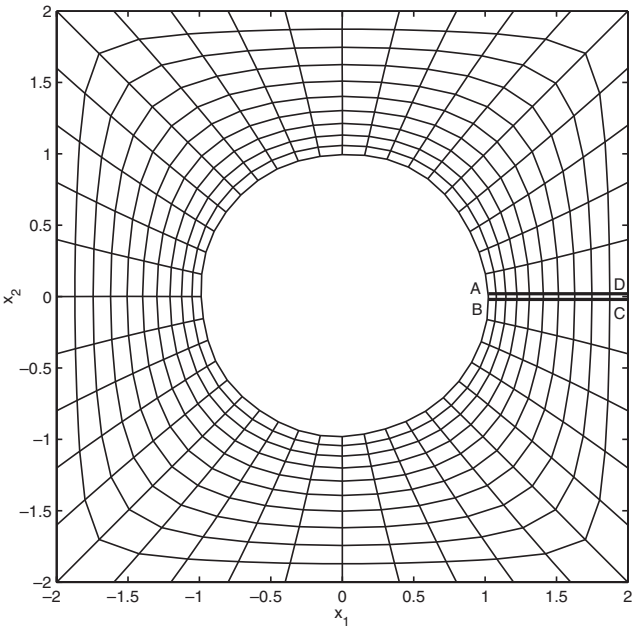


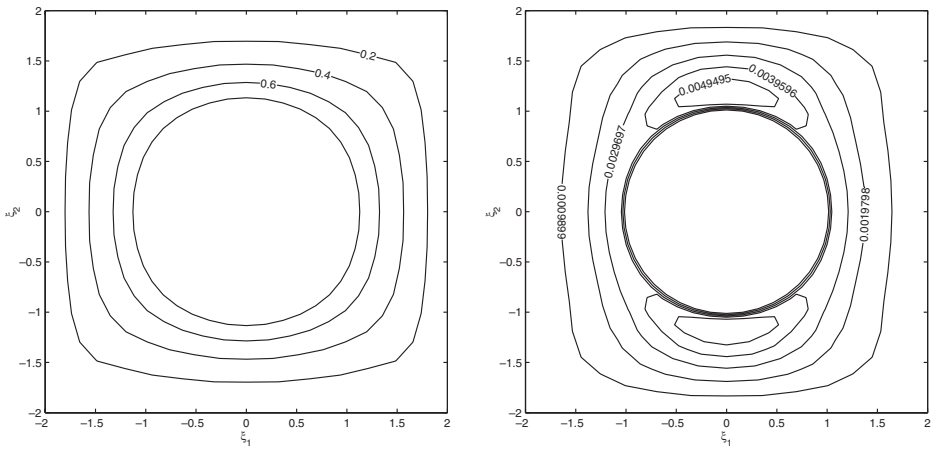
FIG. 4.7. *Random double-connected domain constructed with the* 16*-term* ($N = 2K = 16$) *Fourier expansion, and the corresponding computational mesh.*

Laplace equations (3.3) they are treated as two distinct boundaries. To emphasize this point, these two cuts are shown as two separate lines in Figure 4.7.

The mapped domain in (ξ_1, ξ_2) is a rectangle of $\overline{ABCD}$, whose length $\overline{AB}$ and $\overline{CD}$ is 2π and width $\overline{BC}$ and $\overline{DA}$ is 1. To solve (3.6), we use the Fourier collocation method in the ξ_1-direction (because of the periodicity) and the second-order finite difference method in the ξ_2-direction. Figure 4.7 shows a sample resulting mesh in (x_1, x_2).

The mapped stochastic Poisson equation (4.2) is solved with the 40-point Fourier collocation scheme in the ξ_1-direction, and the ten-point second-order finite difference scheme in the ξ_2-direction. (Numerical tests have been conducted to ensure resolution independence of the solutions in the physical domain.) The resulting mean and standard deviation of the solution are shown in Figure 4.8. Note that in the construction of the inner boundary, the dominating mode ($n = 1$) in the Fourier expansion (4.9) has a period of 2π (the $\sin\theta$ and $\cos\theta$ modes). This results in the two local extrema in the standard deviation field.

These numerical solutions are compared with those obtained from MCS. Figure 4.9 compares the STD profiles obtained with both methods along the radius of $R \approx 1.21$. Due to the smallness of the random input ($\sigma = 0.02$), both MCS and the SG method converge quickly. In fact, the solutions (mean and STD) obtained with the first- and second-order Legendre-chaos expansions almost completely coincide. Also, we observe the convergence of the Monte Carlo method as the number of realizations increases. This example has 16 dimensions in random space ($K = 16$). At first order ($p = 1$), the SG method needs $(K + p)!/K!p! = 17$ terms, and is more efficient than the Monte Carlo method.



(a) (b)

FIG. 4.8. (a) *The mean* $\langle u(x)\rangle$ *and* (b) *STD* $\sigma_u(x)$ *of the random state variable* $u(x,\omega)$*, computed with the SG method.*

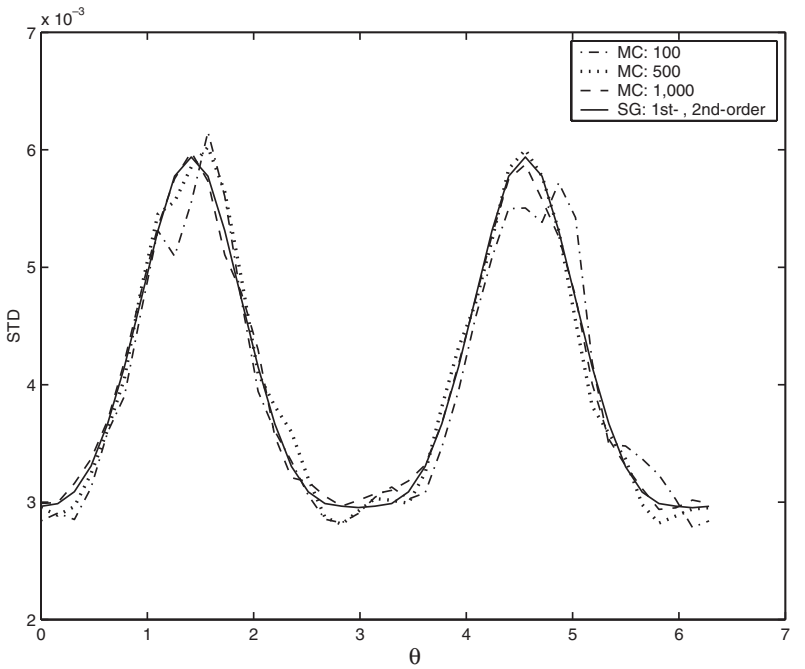


FIG. 4.9. *The STD profiles along the radial cross section* $R = 1.21$*, computed with the SG method (the first- and second-order Legendre chaos) and the MCS consisting of* 100*,* 500*, and* 1,000 *realizations.*

5. Summary. We have proposed a computational framework for the computation of deterministic/stochastic differential equations defined on [illegible]. A key component of the proposed approach is the use of stochastic mappings to transform the original problem into a (better understood) problem of stochastic equations in [illegible]. While a variety of such stochastic mappings can be designed, here we have implemented a mapping defined by solutions of the Laplace equations. We have used numerical examples involving single-connected and double-connected random domains to demonstrate the versatility of the mapping technique.

This random mapping enables us to apply the existing analytical/numerical methods for the resulting transformed stochastic equations in fixed domains. In random spaces, we have presented methods based on Monte Carlo simulations and stochastic Galerkin methods, although we emphasize that other suitable techniques can be used. In physical spaces, we have used a variety of discretizational methods, including the Chebyshev collocation method, Fourier collocation method, and finite difference method. We applied our approach to the elliptic problems in the single- and double-connected random domains, and examined the convergence and efficiency of MCS and SG methods.

Several issues need to be addressed:

- [illegible] There are several sources of errors, in addition to the discretizational errors in solving the transformed SBVP (2.4). In particular, the errors in solving the mapping problems (3.3) and their effect on the Jacobian evaluation (3.8) and the solution of SBVP (2.4) are important. Also, the accurate approximation of the random processes defining rough boundaries by a finite number of random variables in (3.1) and (3.2) plays a key role in the proposed approach. This by itself is an active research area. Furthermore, the effect of the representation errors on the overall accuracy of the present approach is unclear. For example, for an elliptic equation, a small error in the approximation of the shape of a Neumann boundary may be magnified in the final solution [1]. Hence a complete error analysis of the proposed method is required, in addition to the error analysis of the boundary representation.
- [illegible] We have assumed that the random boundary $\partial D(\omega)$ is sufficiently regular for the RDP (2.1) to be well-posed. For example, for the elliptic problems considered in section 4, the [illegible] regularity requirements on the boundary $\partial D(\omega)$ to guarantee the well-posedness, e.g., coercivity, of the transformed stochastic elliptic equation (4.2) is unclear. We thus have restricted ourselves to domains with random boundaries of low to moderate roughness in the numerical examples. The rigorous regularity requirements of the boundary are problem-specific and remain an open issue for a wide class of important problems.
- [illegible] In many cases, the transformed stochastic equations are expected to be more complicated than their original counterparts (compare, for example, (4.1) and (4.2)). However, such transformed equations need to be solved only in the region close to the random boundary. Figure 5.1 shows a schematic of the decomposition of the computational domain into a "boundary region" and a "far-field region," in which the transformed and original equations, respectively, are to be solved. These two equations are coupled through random boundary conditions on the interface between the two domains.

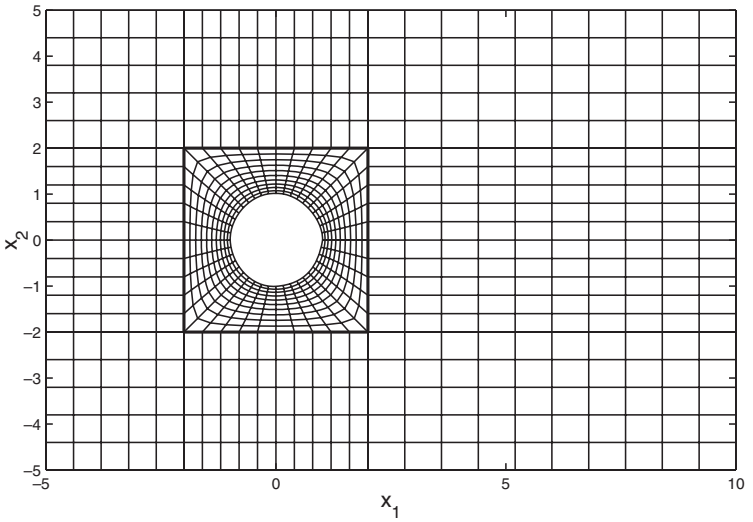


FIG. 5.1. *Schematic representation of a computational mesh for a double-connected domain with a random inclusion. The interface between the "boundary region" and "far-field region" is shown in bold.*

REFERENCES

[1] I. BABUŠKA AND J. CHLEBOUN, *Effects of uncertainties in the domain on the solution of Neumann boundary value problems in two spatial dimensions.*, Math. Comput., 71 (2002), pp. 1339–1370.

[2] I. BABUŠKA AND J. CHLEBOUN, *Effects of uncertainties in the domain on the solution of Dirichlet boundary value problems*, Numer. Math., 93 (2003), pp. 583–610.

[3] I. BABUŠKA, R. TEMPONE, AND G. E. ZOURARIS, *Galerkin finite element approximations of stochastic elliptic differential equations*, SIAM J. Numer. Anal., 42 (2004), pp. 800–825.

[4] D. BEGIS AND R. GLOWINSKI, *Application de la mèthode des èlèments finis à l'approximation d'un problème de domaine optimal. Mèthode de rèsolution des problèmes approchès*, Appl. Math. Optim., 2 (1975), pp. 130–169.

[5] A. BEJAN, *Shape and Structure, from Engineering to Nature*, Cambridge University Press, New York, 2000.

[6] M. BLYTH AND C. POZRIKIDIS, *Heat conduction across irregular and fractal-like surfaces.*, Int. J. Heat Mass Transf., 46 (2003), pp. 1329–1339.

[7] M. BRADY AND C. POZRIKIDIS, *Diffusive transport across irregular and fractal walls.*, Proc. R. Soc. Lond. A Math. Phys. Sci., 442 (1993), pp. 571–583.

[8] D. CAJUEIRO, V. SAMPAIO, C. DE CASTILHO, AND R. ANDRADE, *Fractal properties of equipotentials close to a rough conducting surface*, J. Phys. Condensed Matter, 11 (1999), pp. 4985–4992.

[9] M. DEB, I. BABUŠKA, AND J. ODEN, *Solution of stochastic partial differential equations using Galerkin finite element techniques*, Comput. Methods Appl. Mech. Engrg., 190 (2001), pp. 6359–6372.

[10] C. EVERTSZ AND B. MANDELBROT, *Harmonic measure around a linearly self-similar tree*, J. Phys. A, 25 (1992), pp. 1781–1797.

[11] G. FISHMAN, *Monte Carlo: Concepts, Algorithms, and Applications*, Springer-Verlag, New York, 1996.

[12] P. FRAUENFELDER, C. SCHWAB, AND R. TODOR, *Finite elements for elliptic problems with stochastic coefficients*, Comput. Methods Appl. Mech. Engrg., 194 (2005), pp. 205–228.

[13] M. FYRILLAS AND C. POZRIKIDIS, *Conductive heat transport across rough surfaces and interfaces between two conforming media*, Int. J. Heat Mass Transf., 44 (2001), pp. 1789–1801.

[14] R. GHANEM AND P. SPANOS, *Stochastic Finite Elements: A Spectral Approach*, Springer-Verlag, New York, 1991.

[15] J. GREENWOOD AND J. WILLIAMSON, *Contact of nominally flat surfaces*, Proc. Roy. Soc. A, 295 (1966), pp. 300–319.

[16] M. Grigoriu, *Simulation of stationary non-Gaussian translation processes*, J. Eng. Mech., 124 (1998), pp. 121–126.
[17] H. Homeier and D. Kingham, *Effects of local field variations on the contrast of a field-ion microscope*, J. Phys. D, 16 (1983), pp. L115–L120.
[18] M. Kleiber and T. Hien, *The Stochastic Finite Element Method*, John Wiley, New York, 1992.
[19] Y. N. Lazarev, P. V. Petrov, and D. M. Tartakovsky, *Interface dynamics in randomly heterogeneous porous media*, Adv. Water Resour., 28 (2005), pp. 393–403.
[20] O. Le Maitre, O. Knio, H. Najm, and R. Ghanem, *Uncertainty propagation using Wiener-Haar expansions*, J. Comput. Phys., 197 (2004), pp. 28–57.
[21] O. Le Maitre, H. Najm, R. Ghanem, and O. Knio, *Multi-resolution analysis of Wiener-type uncertainty propagation schemes*, J. Comput. Phys., 197 (2004), pp. 502–531.
[22] W. Liu, T. Belytschko, and A. Mani, *Probabilistic finite elements for nonlinear structural dynamics*, Comput. Methods Appl. Mech. Engrg., 56 (1986), pp. 61–81.
[23] M. Loève, *Probability Theory*, 4th ed., Springer-Verlag, New York, Berlin, 1977.
[24] M. Miller, A. Cerezo, M. Hetherington, and G. Smith, *Atom Probe Field Ion Microscopy*, Monographs on the Physics and Chemistry of Materials, Oxford University Press, Oxford, UK, 1996.
[25] B. Øksendal, *Stochastic Differential Equations. An Introduction with Applications*, 5th ed., Springer-Verlag, Berlin, 1998.
[26] S. Richardson, *A model for the boundary condition of a porous material, Part* 2, J. Fluid Mech., 49 (1971), pp. 327–336.
[27] J. Rudzitis, V. Padamans, E. Bordo, and R. Haytham, *Random process model of rough surfaces contact*, Meas. Sci. Technol., 9 (1998), pp. 1093–1097.
[28] S. Sakamoto and R. Ghanem, *Simulation of multi-dimensional non-Gaussian non-stationary random fields*, Prob. Eng. Mech., 17 (2002), pp. 167–176.
[29] K. Sarkar and C. Meneveau, *Gradients of potential fields on rough surfaces: Perturbative calculation of the singularity distribution function $f(\alpha)$ for small surface dimension*, Phys. Rev. E, 47 (1993), pp. 95–966.
[30] M. Shinozuka and G. Deodatis, *Simulation of stochastic processes by spectral representations*, Appl. Mech. Rev., 44 (1991), pp. 191–203.
[31] C. Soize and R. Ghanem, *Physical systems with random uncertainties: Chaos representations with arbitrary probability measure*, SIAM. J. Sci. Comput., 26 (2004), pp. 395–410.
[32] G. Taylor, *A model for the boundary condition of a porous material, Part* 1, J. Fluid Mech., 49 (1971), pp. 319–326.
[33] J. Thompson, Z. Warsi, and C. Mastin, *Numerical Grid Generation*, Elsevier Science, Amsterdam, 1985.
[34] M. Vignes-Adler, P. Adler, and P. Gougat, *Transport processes along fractals. The Cantor-Taylor brush*, Physicochem. Hydrodynam., 8 (1987), pp. 401–422.
[35] X. Wan, D. Xiu, and G. E. Karniadakis, *Modeling uncertainty in three-dimensional heat transfer problems*, in Advanced Computational Methods in Heat Transfer VIII, Proceedings of the Eighth International Conference on Advanced Computational Methods in Heat Transfer, Lisbon, Portugal, 2004, B. Sunden, C. Brebbia, and A. Mendes, eds., WIT Press, Southampton, UK, 2004, pp. 15–24.
[36] D. Xiu and G. E. Karniadakis, *Modeling uncertainty in steady state diffusion problems via generalized polynomial chaos*, Comput. Methods Appl. Math. Engrg., 191 (2002), pp. 4927–4948.
[37] D. Xiu and G. E. Karniadakis, *The Wiener–Askey polynomial chaos for stochastic differential equations*, SIAM J. Sci. Comput., 24 (2002), pp. 619–644.
[38] D. Xiu and G. E. Karniadakis, *Modeling uncertainty in flow simulations via generalized polynomial chaos*, J. Comput. Phys., 187 (2003), pp. 137–167.
[39] F. Yamazaki and M. Shinozuka, *Digital generation of non-Gaussian stochastic fields*, J. Eng. Mech., 114 (1988), pp. 1183–1197.

SIAM J. SCI. COMPUT.
Vol. 28, No. 3, pp. 1186–1201
© 2006 Society for Industrial and Applied Mathematics

INTEGRATION PRECONDITIONING MATRIX FOR ULTRASPHERICAL PSEUDOSPECTRAL OPERATORS*

ELSAYED M. E. ELBARBARY†

Abstract. In this paper, we give an integration rectangular preconditioner for ultraspherical pseudospectral operators. We present an explicit solution of matrix equations with the structure provided by the pseudospectral discretization of the nth-order differential operator, subject to various types of boundary conditions. This occurs in two steps. In the first step we introduce an integration rectangular preconditioner. In the second step we use a new strategy to impose the boundary conditions, on the condition that the coefficients matrix has a known inverse. Computational examples are given.

Key words. ultraspherical polynomials, pseudospectral methods, differentiation matrix, integration matrix, pseudospectral preconditioner matrix

AMS subject classifications. 65D25, 65F30, 65G05

DOI. 10.1137/050630982

1. Introduction.

In recent decades, pseudospectral methods have proven efficient for obtaining accurate solutions to ordinary and partial differential equations. This leads to computations with full matrices that do not satisfy any good property. The pseudospectral discretization of the nth-order linear one dimensional differential equation, subject to appropriate boundary conditions, produces a badly conditioned linear system that is very difficult to solve. It is well known that the conditioning of the nth-order pseudospectral differentiation matrix scales at best as $O(N^{2n})$, where N is the order of the approximation [20]. The ill-conditioning of the pseudospectral matrices is particularly important since the associated algebraic systems are typically solved by iterative methods. Aspects of numerical stability and accuracy during the solution process for systems of linear equations arising from this method lead us to a modification of the solution path. It was already proposed in [9]: the prior integration of the differential equation. This is in contrast to the good condition number of the resulting linear equations.

The use of integration operators for the treatment of differential equations by orthogonal polynomials dates back to Clenshaw [3]. The integration form of differential equations was applied by Clenshaw and Curtis [4] in spectral space and by El-Gendi [8] in point space. El-Gendi [8] has argued, however, that it is better to directly compute the function values rather than the Chebyshev coefficients [24]. These methods do not extend to nonconstant coefficient problems, since we cannot determine the Green function analytically. Another alternative method, which does not require knowledge of the Green function, was suggested by Jeffreys and Jeffreys [21] and employed in [14, 15, 25, 26, 31] for similar purposes. This procedure is accomplished by starting with Chebyshev approximation for the highest-order derivative and generating approximations to the lower-order derivatives through integration of the highest-order derivative.

*Received by the editors May 8, 2005; accepted for publication (in revised form) February 7, 2006; published electronically July 17, 2006.
http://www.siam.org/journals/sisc/28-3/63098.html
†Department of Mathematics, Faculty of Education, Ain Shams University, Roxy, Hiliopolis, Cairo, Egypt. Current address: Department of Mathematics, AlJouf Teachers College, AlJouf, Skaka, P.O. Box 269, Saudi Arabia (elsayed_elbarbary@yahoo.com).

Pseudospectral methods have been developed by presenting suitable preconditioners. Previous works on preconditioners for pseudospectral operators resulted in the development of preconditioners based on a low-order approximation of the inverse operator. Orszag [27] suggested that one use a finite difference operator based on the collocation points as a preconditioner of the Chebyshev pseudospectral discretization of the Poisson equation. For the first-order problems, the corresponding strategy of using finite difference operators does not lead to analogous satisfactory results [10]. Some other preconditioners have been proposed and successfully tested in [2, 6, 7, 10, 22, 23, 28, 29].

Recent works using the integration preconditioning idea [5, 18] follow a different approach. Hesthaven [18] applies the boundary conditions by a penalty method, through appropriate adjusting coefficients in the spectral integrator, while Coutsias, Hagstrom, and Torres [5] choose a definite integral that introduces the first k expansion coefficients as arbitrary parameters, which are chosen to satisfy the boundary conditions after the equation has been integrated for the higher coefficients.

Recently, Ghoreishi and Hosseini [13] presented a pseudospectral preconditioner on arbitrary grids closely related to the idea of Hesthaven [18].

The aim of this paper is to present the optimal preconditioner (exact inverse) of the nth-order one dimensional differential operators, subject to various types of boundary conditions. This takes place in two steps. First, we introduce an integration rectangular preconditioner related to the idea of Hesthaven [18]. Second, we use a new strategy to impose the boundary conditions, on the condition that the coefficients matrix has a known inverse.

The remaining part of this paper is organized as follows. In section 2, we introduce the spectral and pseudospectral differentiation matrices based on ultraspherical polynomials and discuss the properties of ultraspherical polynomials. Here, we focus on the new property that results, a relation between the ultraspherical polynomials and their derivatives, which is used to define the nth-order integration operator. We also continue by briefly revising the Hesthaven work [18]. In section 3, we use the spectral integration matrix and its inverse (spectral differentiation matrix) in a restricted domain to define the well-known pseudospectral differentiation matrix. We will define rectangular pseudospectral preconditioners based on the spectral integration matrix in a restricted domain. Section 4 addresses the performance of the rectangular pseudospectral preconditioner for the advective operator. We suggest a new strategy to impose the boundary conditions to obtain an optimal preconditioner (exact inverse). In section 5, we present a similar analysis for the diffusion operator, subject to Dirichlet as well as Robin boundary conditions. Also, a similar treatment for the Helmholtz equation is presented in section 6. Numerical computations have been done on the Helmholtz equation and singular perturbation problem in section 7. Section 8 contains the conclusion of this paper.

2. Pseudospectral preconditioner matrix. We assume that $\{C_n^{(\lambda)}(x)\}_{n=0}^{\infty}$ is a family of ultraspherical polynomials in the finite domain $x \in [-1, 1]$, with the first two being

$$C_0^{(\lambda)}(x) = 1, \qquad C_1^{(\lambda)}(x) = (2\lambda + 1)x,$$

while the remaining polynomials are given through the recursion formula [1, 18, 30]

$$(2n + 2\lambda + 1)xC_n^{(\lambda)}(x) = (n + 1)C_{n+1}^{(\lambda)}(x) + (n + 2\lambda)C_{n-1}^{(\lambda)}(x).$$

If $u(x) \in C^{\infty}[-1,1]$, then we will write

$$u_N(x) = \sum_{k=0}^{N} \hat{u}_k^{(0)} C_k^{(\lambda)}(x),$$

where $\hat{u}_k^{(0)}$ represents the continuous expansion coefficients. Utilizing the orthogonality of the orthogonal polynomials in $L_w^2[-1,1]$ with weight function $w(x) = (1-x^2)^{\lambda}$, we have

$$\hat{u}_k^{(0)} = \frac{1}{\gamma_k^*} \int_{-1}^{1} u(x) C_k^{(\lambda)}(x) w(x) dx,$$

where the continuous normalization factor, γ_k^*, is defined as [18]

$$\gamma_k^* = 2^{2\lambda+1} \frac{\Gamma^2(\lambda+1)\Gamma(k+2\lambda+1)}{k!(2k+2\lambda+1)\Gamma^2(2\lambda+1)}, \quad k = 0,1,2,\ldots,N.$$

The ith derivative of $u(x)$ is approximated by series expansion in the form

$$u_N^{(i)}(x) = \sum_{k=0}^{N} \hat{u}_k^{(i)} C_k^{(\lambda)}(x),$$

and the ith order spectral differentiation matrix $D_s^{(i)}$ is introduced, providing that

$$\hat{\mathbf{u}}^{(i)} = D_s^{(i)} \hat{\mathbf{u}}^{(0)},$$

where $\hat{\mathbf{u}}^{(i)} = (\hat{u}_0^{(i)}, \hat{u}_1^{(i)}, \ldots, \hat{u}_N^{(i)})^T$.

The discrete expansion of $u(x)$ is given by [18],

$$u_N(x) = \sum_{k=0}^{N} \tilde{u}_k C_k^{(\lambda)}(x),$$

where

$$\tilde{u}_k = \frac{1}{\gamma_k} \sum_{i=0}^{N} \omega_i \; u(x_i) C_k^{(\lambda)}(x_i), \quad n = 0,1,\ldots,N. \tag{2.1}$$

The grid points (Gauss–Lobatto points) are ordered as

$$-1 = x_0 < x_1 < \cdots < x_{N-1} < x_N = 1,$$

where

$$\{x_i\}_{i=0}^{N} = \left\{ x \in [-1,1] \mid (1-x^2) \frac{dC_N^{(\lambda)}(x)}{dx} = 0 \right\}.$$

The discrete normalization factor, $\gamma_k = \gamma_k^*$, $k = 0,1,\ldots,N-1$, while

$$\gamma_N = 2^{2\lambda+1} \frac{\Gamma^2(\lambda+1)\Gamma(N+2\lambda+1)}{NN!\Gamma^2(2\lambda+1)}.$$

The Gauss–Lobatto weights, ω_i, are given through the following formula:

$$\omega_i = \begin{cases} 2^{2\lambda+1}\frac{(N-1)!\Gamma(\lambda+1)\Gamma(\lambda+2)}{\Gamma(N+2\lambda+2)}, & i=0,N, \\ -2^{2\lambda+1}\frac{\Gamma^2(\lambda+1)\Gamma(N+2\lambda+1)}{N!(N+2\lambda+1)\Gamma^2(2\lambda+1)}\left[C_N^{(\lambda)}(x_i)\frac{d}{dx}C_{N-1}^{(\lambda)}(x_i)\right]^{-1}, & i=1,2,\ldots,N-1. \end{cases}$$

Thus $u_N(x_i) = u(x_i)$, $i = 0, 1, \ldots, N$; in other words, the discrete projection is exact at the points $\{x_i\}_{i=0}^N$.

The kth-order pseudospectral differentiation matrix $D_p^{(k)}$ is introduced on the condition that

$$\mathbf{u}^{(k)} = D_p^{(k)}\mathbf{u}^{(0)},$$

where

$$\mathbf{u}^{(k)} = (u^{(k)}(x_0), u^{(k)}(x_1), \ldots, u^{(k)}(x_N))^T,$$

and the entries of the pseudospectral differentiation matrix, $D_p^{(k)}$, are given by

$$D_{ij}^{(k)} = \omega_j \sum_{n=k}^{N} \frac{1}{\gamma_n} C_n^{(\lambda)}(x_j)\frac{d^k}{dx^k}C_n^{(\lambda)}(x_i).$$

It is well known that $D_p^{(k)}$ is a centro-antisymmetric full matrix with a condition number proportional to N^{2k} [10].

The following theorem has fundamental importance in developing the integration operator. It gives a relation between the ultraspherical polynomials and their derivatives.

THEOREM 2.1. [illegible]

$$C_n^{(\lambda)}(x) = \sum_{m=0}^{k} \eta_{k,m,n}\frac{d^k}{dx^k}C_{n+k-2m}^{(\lambda)}(x), \qquad n > k,$$

[illegible]

$$\eta_{k,m,n} = \frac{(-1)^m\binom{k}{m}}{\prod_{\ell\neq m}^{k_{\ell=0}}(2k+2n+2\lambda-2\ell-2m+1)}.$$

As special cases, we have the following:

(1) $k = 1$,

$$C_n^{(\lambda)}(x) = \frac{\frac{d}{dx}C_{n+1}^{(\lambda)}(x)}{(2n+2\lambda+1)} - \frac{\frac{d}{dx}C_{n-1}^{(\lambda)}(x)}{(2n+2\lambda+1)}, \qquad n > 1; \tag{2.2}$$

(2) $k = 2$,

$$C_n^{(\lambda)}(x) = \frac{\frac{d^2}{dx^2}C_{n+2}^{(\lambda)}(x)}{(2n+2\lambda+1)(2n+2\lambda+3)} - \frac{2\frac{d^2}{dx^2}C_n^{(\lambda)}(x)}{(2n+2\lambda+3)(2n+2\lambda-1)}$$
$$+ \frac{\frac{d^2}{dx^2}C_{n-2}^{(\lambda)}(x)}{(2n+2\lambda+1)(2n+2\lambda-1)}, \qquad n > 2; \tag{2.3}$$

(3) $k = 3$,

$$\begin{aligned} C_n^{(\lambda)}(x) &= \frac{\frac{d^3}{dx^3}C_{n+3}^{(\lambda)}(x)}{(2n+2\lambda+1)(2n+2\lambda+3)(2n+2\lambda+5)} \\ &\quad - \frac{3\frac{d^3}{dx^3}C_{n+1}^{(\lambda)}(x)}{(2n+2\lambda+1)(2n+2\lambda+5)(2n+2\lambda-1)} \\ &\quad + \frac{3\frac{d^3}{dx^3}C_{n-1}^{(\lambda)}(x)}{(2n+2\lambda+1)(2n+2\lambda+3)(2n+2\lambda-3)} \\ &\quad - \frac{\frac{d^3}{dx^3}C_{n-3}^{(\lambda)}(x)}{(2n+2\lambda+1)(2n+2\lambda-1)(2n+2\lambda-3)}, \qquad n > 3. \end{aligned} \tag{2.4}$$

The recursions (2.2) and (2.3) are well-known recurrence formulas. Recurrence relations for higher-order derivatives may likewise be obtained by repeatedly combining the relations (2.2)–(2.3). The proof of this theorem can be easily seen by induction.

We now introduce the row vectors

$$\mathbf{C} = [C_0^{(\lambda)}(x), C_1^{(\lambda)}(x), \ldots, C_N^{(\lambda)}(x)]$$

and

$$\mathbf{C}_x = \left[\frac{d}{dx}C_0^{(\lambda)}(x), \frac{d}{dx}C_1^{(\lambda)}(x), \ldots, \frac{d}{dx}C_N^{(\lambda)}(x)\right].$$

Hesthaven [18] observed that the two row vectors $\mathbf{C}$ and $\mathbf{C}_x$ are related through the integral operator $B_{[1]}^{(-1)}$, as

$$\mathbf{C} = \mathbf{C}_x B_{[1]}^{(-1)}. \tag{2.5}$$

The notation $B_{[k]}$ denotes a matrix with its first k rows set to zero. The elements of the matrix $B_{[1]}^{(-1)}$ can be determined from the recursion relations (2.2). Consider that $\frac{d}{dx}C_0^{(\lambda)}(x) = 0$, so the first row of integral operator $B_{[1]}^{(-1)}$ can set to zero.

We note that, in the case of finite order expansion, (2.5) is correct except for the last elements in $\mathbf{C}$. For example, in the case of $N = 4$,

$$\mathbf{C} - \mathbf{C}_x B_{[1]}^{(-1)} = \left(0, 0, 0, 0, -\frac{5}{24} + \frac{35}{12}x^2 - \frac{35}{8}x^4\right).$$

Thus it happens that, in the more general case, the last k elements in $\mathbf{C}$ are not correct for the kth derivative. Also, we note that the matrix $B_{[1]}^{(-1)}$ is singular.

Let us introduce the spaces Q_m^n by

$$Q_m^n = \operatorname{span}\{C_k^{(\lambda)}(x) \mid m \le k \le n\}.$$

Hence, in the absence of the first row and last column of the matrix $B_{[1]}^{(-1)}$, the matrix $B_{[1]}^{(-1)}$ will be reduced to $B^{(-1)}$. The matrix $B^{(1)}$ is a uniquely defined inverse of the integration matrix $B^{(-1)}$ in the following restricted domain:

$$B^{(1)} : Q_1^N \longrightarrow Q_0^{N-1} \quad \text{and} \quad B^{(-1)} : Q_0^{N-1} \longrightarrow Q_1^N.$$

The spectral differentiation matrix $D_s^{(1)}$ is determined from $B^{(1)}$ by adding the first column and the last row, which must be identically zero.

Introducing the vectors

$$\mathbf{u} = (u(-1), u(x_1), \ldots, u(x_{N-1}), u(1))^T \quad \text{and} \quad \tilde{\mathbf{u}} = (\tilde{u}_0, \tilde{u}_1, \ldots, \tilde{u}_N)^T,$$

we see that the two vectors $\mathbf{u}$ and $\tilde{\mathbf{u}}$ are related through the similarity transformations [18]

$$\mathbf{u} = T^{-1}\tilde{\mathbf{u}}, \qquad \tilde{\mathbf{u}} = T\mathbf{u},$$

where the entries of the $(N+1) \times (N+1)$ orthogonal matrices, T^{-1} and T, are given as

$$T_{ij}^{-1} = C_j^{(\lambda)}(x_i), \qquad T_{ij} = \frac{\omega_j}{\gamma_i} C_i^{(\lambda)}(x_j), \qquad i, j = 0, 1, \ldots, N.$$

We observe that

$$(T^{-1}T)_{ij} = \omega_j \sum_{n=0}^{N} \frac{1}{\gamma_n} C_n^{(\lambda)}(x_i) C_n^{(\lambda)}(x_j) = \delta_{i,j}, \tag{2.6}$$

implying that

$$T^{-1}T = I,$$

where $\delta_{i,j}$ is the Kronecker symbol. Hesthaven [18] used the spectral differentiation matrix $D_s^{(1)}$ to define the pseudospectral differentiation matrix $D_p^{(1)}$ as follows:

$$D_p^{(1)} = T^{-1} D_s^{(1)} T.$$

Likewise, Hesthaven [18] made an analogous analysis for the higher derivatives.

He presented singular integration preconditioners of the first- and second-order pseudospectral differentiation matrices as

$$P^{(-1)} = T^{(-1)} B_{[1]}^{(-1)} T, \qquad P^{(-2)} = T^{(-1)} B_{[2]}^{(-2)} T,$$

and he has developed his preconditioners by overcoming the singularity problem by defining modified spectral integration matrices $\tilde{\mathrm{B}}_{[1]}^{(-1)}$ and $\tilde{\mathrm{B}}_{[2]}^{(-2)}$. For the first-order differentiation matrix he has replaced the last element in the first row by a certain parameter. For the second-order differentiation matrix, he has replaced the last two elements of the first and second rows by a certain four parameters. These parameters can be determined, providing that the matrices $\tilde{\mathrm{B}}_{[1]}^{(-1)}$ and $\tilde{\mathrm{B}}_{[2]}^{(-2)}$ are nonsingular, by properly choosing the parameters, a choice that allows for optimizing the preconditioner.

In the next section, we consider an alternative procedure to identify a rectangular preconditioner matrix for an nth-order linear differential operator.

3. Rectangular pseudospectral preconditioner matrix. Imposing the boundary conditions strongly would alter the appropriate rows and columns in the differentiation matrix. The delicate symmetries of the differentiation matrix are ruined, and the construction of the inverse is no longer clear. Funaro and Gottlieb [11] introduced a penalty method which enforced the boundary condition only weakly through

a boundary correction rather than modifying the coefficient matrix. Such a penalty method has been studied for the collocation formulation with preconditioning [11]. Further details on the penalty method can be found in, e.g., [12, 16, 17, 19, 20]. Now, let us consider the construction of a preconditioner in two steps. The first step is done in absence of the boundary conditions, and the obtained coefficients matrix is a rectangular matrix. In the second step, the enforcing of the boundary conditions is done by adding auxiliary equations, on the condition that the coefficients matrix becomes a square matrix and its inverse is known.

We next introduce the row vectors

$$\mathbf{C}^{(0)}_{0,N-k} = [C^{(\lambda)}_0(x), C^{(\lambda)}_1(x), \ldots, C^{(\lambda)}_{N-k}(x)]$$

and

$$\mathbf{C}^{(k)}_{k,N} = \left[\frac{d^k}{dx^k}C^{(\lambda)}_k(x), \frac{d^k}{dx^k}C^{(\lambda)}_{k+1}(x), \ldots, \frac{d^k}{dx^k}C^{(\lambda)}_N(x)\right],$$

The row vectors $\mathbf{C}^{(0)}_{0,N-k}, \mathbf{C}^{(k)}_{k,N}$ are related through the integration matrix $B^{(-k)}$ as follows:

$$\mathbf{C}^{(0)}_{0,N-k} = \mathbf{C}^{(k)}_{k,N} B^{(-k)},$$

where

$$B^{(-k)} = [b^k_{ij}], \quad i,j = 0,1,2,\ldots,N-k,$$

$$b^k_{ij} = \begin{cases} \eta_{k,\alpha,j}, & j = i+2\alpha,\ \alpha = 0,1,\ldots,k, \\ 0 & \text{otherwise.} \end{cases}$$

As special cases, we have the following:

(1) $k = 1$,

$$B^{(-1)} = [b^1_{ij}], \quad i,j = 0,1,2,\ldots,N-1,$$

$$b^1_{ij} = \begin{cases} \frac{-1}{(2j+2\lambda+1)}, & j = i+2, \\ \frac{1}{(2j+2\lambda+1)}, & j = i, \\ 0 & \text{otherwise.} \end{cases}$$

(2) $k = 2$,

$$B^{(-2)} = [b^2_{ij}], \quad i,j = 0,1,2,\ldots,N-2,$$

$$b^2_{ij} = \begin{cases} \frac{1}{(2j+2\lambda+1)(2j+2\lambda-1)}, & j = i+4, \\ \frac{-2}{(2j+2\lambda+3)(2j+2\lambda-1)}, & j = i+2, \\ \frac{1}{(2j+2\lambda+1)(2j+2\lambda+3)}, & j = i, \\ 0 & \text{otherwise.} \end{cases}$$

The eigenvalues of the matrix $B^{(-k)}$ are given by

$$\mu_i = \eta_{k,0,i}, \qquad i = 0,1,\ldots,N-k,$$

and the eigenvalues of the matrix $B^{(k)}$ are $\frac{1}{\mu_i}$, $i = 0, 1, \ldots, N-k$.

Define the following rectangular matrices:

$$E_k = [e_{ij}], \quad e_{ij} = \frac{\omega_j}{\gamma_i} C_i^{(\lambda)}(x_j), \quad i = 0, 1, 2, \ldots, N-k, \quad j = 0, 1, 2 \ldots, N,$$
$$H_k = [h_{ij}], \quad h_{ij} = C_j^{(\lambda)}(x_i), \quad i = 0, 1, 2, \ldots, N, \quad j = k, k+1, \ldots, N,$$
$$R^{(k)} = [r_{ij}^{(k)}], \quad r_{ij}^{(k)} = \frac{d^k}{dx^k} C_j^{(\lambda)}(x_i), \quad i = 0, 1, 2, \ldots, N, \quad j = k, k+1, \ldots, N,$$
$$S_k = [s_{ij}], \quad s_{ij} = \frac{\omega_j}{\gamma_i} C_i^{(\lambda)}(x_j), \quad i = k, k+1, \ldots, N, \quad j = 0, 1, 2 \ldots, N,$$
$$V_k = [v_{ij}], \quad v_{ij} = C_j^{(\lambda)}(x_i), \quad i = 0, 1, 2, \ldots, N, \quad j = 0, 1, 2, \ldots, N-k.$$

It is easy to verify the relations

$$D_p^{(k)} = R^{(k)} S_k,$$
$$V_k = R^{(k)} B^{(-k)},$$
$$R^{(k)} = V_k B^{(k)},$$
$$S_k H_k = I, \text{ while } H_k S_k \neq I,$$

which can be used to compute the pseudospectral differentiation matrix as follows:

$$D_p^{(k)} = V_k B^{(k)} S_k. \tag{3.1}$$

In view of the equation

$$E_k V_k = I, \quad \text{while } V_k E_k \neq I, \tag{3.2}$$

we introduce the rectangular $(N-k+1) \times (N+1)$ matrix

$$P^{(-k)} = B^{(-k)} E_k,$$

which represents the suggested rectangular integration preconditioner. Using $P^{(-k)}$ and (3.1) and (3.2), we will get

$$P^{(-k)} D_p^{(k)} = S_k.$$

The matrix S_k is a rectangular $(N-k+1) \times (N+1)$ matrix. Namely, the use of this rectangular preconditioner in the pseudospectral discretization makes the number of equations plus the number of boundary conditions equal the number of unknowns. Therefore, we do not delete equations corresponding to the boundary conditions. We note that the matrix S_k is a part of matrix T. Then by adding the first k rows of the matrix T to the matrix S_k, become identically. So, we will add k auxiliary equations as

$$\sum_{n=0}^{N} \frac{\omega_n}{\gamma_i} C_i^{(\lambda)}(x_n) u(x_n) = \beta_i, \qquad i = 1, 2, \ldots, k,$$

where β_i are parameters to be determined from the boundary conditions.

In view of (2.1), we see that the parameters β_i, $i = 1, 2, \ldots, k$, are exactly the coefficients of the k lowest modes in the expansion of the solution.

According to what we mentioned above, the matrix S_k becomes T, which has an inverse T^{-1}. We then provide the explicit solution to the system of algebraic equations that arise from the pseudospectral discretization.

In the next sections, we will apply the previous idea upon the advective and diffusive operators, subject to various types of boundary conditions.

4. Preconditioning of the advective operator. Let us consider the first-order problem

$$\frac{du(x)}{dx} = f(x), \qquad u(1) = u_0.$$

We seek a polynomial solution for using the discrete approximation

$$D_p^{(1)}\mathbf{u} = \mathbf{f}, \qquad u(x_0) = u_0.$$

Here $\mathbf{u}$ and $\mathbf{f}$ refer to the grid vectors of the two functions $u(x)$ and $f(x)$ at the Gauss–Lobatto collocation points associated with the polynomial basis $C_n^{(\lambda)}(x)$. Since the condition number of $D_p^{(1)}$ is proportional to N^2, this may be nontrivial for N large. Now we will keep the structure of $D_p^{(1)}$, use the rectangular preconditioners, and delay the use of the boundary conditions for a next step.

To proceed, we apply the rectangular preconditioner $P^{(-1)}$, as

$$S_1\ \mathbf{u} = B^{(-1)}E_1\ \mathbf{f} = P^{(-1)}\mathbf{f}. \tag{4.1}$$

Equation (4.1) consists of N equations in $(N+1)$ unknowns (in the absence of the boundary conditions).

By adding auxiliary equation

$$\sum_{n=0}^{N} \frac{\omega_n}{\gamma_0} C_0^{(\lambda)}(x_n)u(x_n) = \beta_1,$$

the matrices S_1 and T became identical.

From this one can write

$$\mathbf{u} = T^{-1}\begin{pmatrix} \beta_1 \\ P^{(-1)}\mathbf{f} \end{pmatrix}. \tag{4.2}$$

The value of parameter β_1 can be determined from the boundary condition as

$$\beta_1 = u(x_0) - \sum_{j=1}^{N} p_j^{(-1)} C_j^{(\lambda)}(x_0),$$

where $p_j^{(-1)}$, $j = 1, 2, \ldots, N$, are the elements of the vector $P^{(-1)}\mathbf{f}$. Therefore, we could obtain $\mathbf{u}$ directly from (4.2).

5. Preconditioning of the diffusive operator. In the next two subsections we shall show how the specification of boundary conditions allows us to determine the values of the added parameters.

5.1. Dirichlet boundary conditions. Let us consider the homogeneous elliptic problem

$$\frac{d^2u(x)}{dx^2} = f(x), \quad u(1) = u_0, \quad u(-1) = u_N, \tag{5.1}$$

and seek solutions of polynomial form leading to a discrete form of (5.1) as

$$D_p^{(2)}\mathbf{u} = \mathbf{f}, \quad u(x_0) = u_0, \quad u(x_N) = u_N,$$

where $\mathbf{u}$ and $\mathbf{f}$ refer to the vectors of $u(x)$ and $f(x)$ at the Gauss–Lobatto points. Following the development of the rectangular preconditioner for the advection operator, we apply the preconditioner $P^{(-2)}$ as

$$S_2\mathbf{u} = B^{(-2)}E_2\mathbf{f} = P^{(-2)}\mathbf{f}. \tag{5.2}$$

This equation consists of $N-1$ equations in $(N+1)$ unknowns (without boundary conditions).

By adding auxiliary equations

$$\sum_{n=0}^{N} \frac{\omega_n}{\gamma_0} C_0^{(\lambda)}(x_n)u(x_n) = \beta_1 \tag{5.3}$$

and

$$\sum_{n=0}^{N} \frac{\omega_n}{\gamma_1} C_1^{(\lambda)}(x_n)u(x_n) = \beta_2, \tag{5.4}$$

the matrices S_2 and T became identical. Therefore,

$$\mathbf{u} = T^{-1}\begin{pmatrix} \beta_1 \\ \beta_2 \\ P^{(-2)}\mathbf{f} \end{pmatrix}.$$

In other words,

$$u_k = \beta_1 + \beta_2 C_1^{(\lambda)}(x_k) + \varphi(x_k), \quad k = 0, 1, \ldots, N, \tag{5.5}$$

where $\varphi(x) = \sum_{j=2}^{N} p_j^{(-2)} C_j^{(\lambda)}(x)$, and $p_j^{(-2)}$, $j = 2, 3, \ldots, N$, are the elements of the vector $P^{(-2)}\mathbf{f}$.

The values of parameters β_1 and β_2 are determined from the boundary conditions as

$$\beta_1 = \frac{1}{2}(u(x_0) + u(x_N) - \varphi(x_0) - \varphi(x_N)),$$

$$\beta_2 = \frac{1}{2(2\lambda+1)}(u(x_0) - u(x_N) - \varphi(x_0) + \varphi(x_N)).$$

Therefore, we could obtain $\mathbf{u}$ directly from (5.5).

5.2. Robin boundary conditions. Consider the Poisson problem

$$\frac{d^2u(x)}{dx^2} = f(x),$$

subject to Robin boundary conditions

$$\gamma_0 u_x(1) + \gamma_1 u(1) = \gamma_2, \qquad \delta_0 u_x(-1) + \delta_1 u(-1) = \delta_2. \tag{5.6}$$

The analysis for the Robin problem follows the same route described for the Dirichlet problem until (5.5). When we apply the boundary conditions (5.6) in (5.5), the parameters β_1 and β_2 are determined as follows:

$$\beta_2 = \frac{\gamma_2\delta_1 - \gamma_1\delta_2 - \gamma_1\delta_1\varphi(x_0) - \gamma_0\delta_1\varphi'(x_0) + \gamma_1\delta_0\varphi'(x_N) + \gamma_1\delta_1\varphi(x_N)}{(2\lambda+1)(-\gamma_1\delta_0 + \gamma_0\delta_1 + \gamma_1\delta_1 + \gamma_1\delta_1)},$$

$$\beta_1 = \frac{\gamma_2 - \gamma_1\varphi(x_0) - \gamma_0\varphi'(x_0)}{\gamma_1} - \beta_2\frac{(2\lambda+1)(\gamma_0+\gamma_1)}{\gamma_1}.$$

We note that in the case of Neumann boundary conditions, $\gamma_1 = \delta_1 = 0$, the parameters are undetermined. Thus, the Neumann problems remain open, and we will stop at (5.2) to solve the resulting algebraic equations and the boundary conditions equations, while the Robin problems are solved on the condition that $\gamma_1 \neq 0$.

6. Preconditioning of the Helmholtz equation. In this section, we are interested in using the pseudospectral method with integration preconditioning to solve the Helmholtz equation

$$\alpha u(x) - u''(x) = f(x), \qquad u(1) = u_0, u(-1) = u_N, \tag{6.1}$$

and seek solutions of polynomial form leading to a discrete form of (6.1) as

$$\alpha \mathbf{u} - D_p^{(2)} \mathbf{u} = \mathbf{f}, \qquad u(x_0) = u_0, u(x_N) = u_N.$$

We apply the preconditioner $P^{(-2)}$ as

$$\alpha P^{(-2)} \mathbf{u} - S_2 \mathbf{u} = P^{(-2)} \mathbf{f}.$$

By adding the auxiliary equations (5.3) and (5.4), we will get

$$\alpha \begin{pmatrix} \mathbf{0} \\ \mathbf{0} \\ P^{(-2)} \end{pmatrix} \mathbf{u} - T\mathbf{u} = \begin{pmatrix} 0 \\ 0 \\ P^{(-2)} \mathbf{f} \end{pmatrix} - \begin{pmatrix} \beta_1 \\ \beta_2 \\ 0 \\ \vdots \\ 0 \end{pmatrix}.$$

Hence,

$$\tilde{\mathrm{A}} \mathbf{u} = \left(\alpha T^{-1} \begin{pmatrix} \mathbf{0} \\ \mathbf{0} \\ P^{(-2)} \end{pmatrix} - I \right) \mathbf{u} = T^{-1} \begin{pmatrix} 0 \\ 0 \\ P^{(-2)} \mathbf{f} \end{pmatrix} - \begin{pmatrix} \beta_1 + \beta_2 C_1^{(\lambda)}(x_0) \\ \beta_1 + \beta_2 C_1^{(\lambda)}(x_1) \\ \beta_1 + \beta_2 C_1^{(\lambda)}(x_2) \\ \vdots \\ \beta_1 + \beta_2 C_1^{(\lambda)}(x_N) \end{pmatrix}. \tag{6.2}$$

The elements of the coefficient matrix $\tilde{\mathrm{A}}$ are given by

$$a_{i,j} = \sum_{l=2}^{N} \alpha C_l^{(\lambda)}(x_i) \sum_{k=0}^{N-2} b_{l-2,k}^2 \frac{\omega_j}{\gamma_k} C_k^{(\lambda)}(x_j) - \delta_{i,j}, \qquad i, j = 0, 1, \ldots, N.$$

The unknowns in (6.2) are given by $\beta_1, \beta_2, u(x_1), u(x_2), \ldots, u(x_{N-1})$. Then we get

$$A\mathbf{v} = T^{-1} \begin{pmatrix} 0 \\ 0 \\ P^{(-2)} \mathbf{f} \end{pmatrix} - \begin{pmatrix} a_{0,0} u_0 + a_{0,N} u_N \\ a_{1,0} u_0 + a_{1,N} u_N \\ a_{2,0} u_0 + a_{2,N} u_N \\ \vdots \\ a_{N,0} u_0 + a_{N,N} u_N \end{pmatrix}, \tag{6.3}$$

where $\mathbf{v} = (\beta_1, u(x_1), u(x_2), \ldots, u(x_{N-1}), \beta_2)^T$ and

$$A = \begin{pmatrix} 1 & a_{0,1} & a_{0,2} & \cdots & a_{0,N-1} & C_1^{(\lambda)}(x_0) \\ 1 & a_{1,1} & a_{1,2} & \cdots & a_{1,N-1} & C_1^{(\lambda)}(x_1) \\ 1 & a_{2,1} & a_{2,2} & \cdots & a_{2,N-1} & C_1^{(\lambda)}(x_2) \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & a_{N,1} & N_{0,2} & \cdots & a_{N,N-1} & C_1^{(\lambda)}(x_N) \end{pmatrix}.$$

Hence, we get the solution by solving the system of equations in (6.3).

7. Numerical results. In this section, we present two numerical examples using the integration preconditioning pseudospectral Chebyshev ($PChP$) (Legendre (PLP)) method. The numerical results will be compared with those obtained by the standard pseudospectral Chebyshev (ChP) (Legendre (LP)) method. The following tables contain a "condition number," which can be estimated as follows:

$$Cond(A) = \sqrt{\frac{\text{Maximum eigenvalue of } (AA^T)}{\text{Minimum eigenvalue of } (AA^T)}}.$$

Calculations were performed on a PC running MATHEMATICA software.

7.1. Our first example is the Helmholtz equation

$$u(x) - u''(x) = -\frac{x+1}{2}, \qquad u(\pm 1) = 0, \tag{7.1}$$

with the exact solution

$$u(x) = \frac{\sinh(x+1)}{\sinh(2)} - \frac{x+1}{2}.$$

In the case of $\alpha = 1$, $u_0 = u_N = 0$, and $f(x) = -\frac{x+1}{2}$, we solve the system of equations (6.3). The numerical results are compared with those obtained by the classical pseudospectral method in Table 1.

TABLE 1
Numerical results for the Chebyshev basis.

	ChP		$PChP$	
N	Condition number	Maximal error	Condition number of the matrix A	Maximal error
16	9.40×10^{02}	3.05×10^{-16}	10.39	1.80×10^{-16}
32	1.48×10^{04}	1.39×10^{-16}	19.83	2.53×10^{-16}
64	2.35×10^{05}	1.79×10^{-16}	38.63	2.64×10^{-16}
128	3.76×10^{06}	1.37×10^{-16}	76.20	2.97×10^{-16}
256	6.01×10^{07}	4.62×10^{-14}	151.31	5.27×10^{-16}
512	9.62×10^{08}	7.10×10^{-13}	301.54	4.99×10^{-16}
1024	1.54×10^{10}	3.22×10^{-12}	601.98	1.08×10^{-15}

7.2. Our second example has variable coefficients, and the solution develops two boundary layers of width $O(\epsilon)$ near $x = \pm 1$. The equation is

$$\epsilon u''(x) - xu'(x) - u(x) = 0, \quad u(\pm 1) = 1, \tag{7.2}$$

and the exact solution is $u(x) = e^{\frac{x^2-1}{2\epsilon}}$.

Applying the preconditioner $P^{(-2)}$ to (7.2) and adding the auxiliary equations (5.3) and (5.4), we will get

$$\left(\epsilon I - T^{-1}\begin{pmatrix}\mathbf{0}\\ \mathbf{0}\\ P^{(-2)}(FD_p^{(-1)}+I)\end{pmatrix}\right)\mathbf{u} = \begin{pmatrix}\beta_1+\beta_2 C_1^{(\lambda)}(x_0)\\ \beta_1+\beta_2 C_1^{(\lambda)}(x_1)\\ \beta_1+\beta_2 C_1^{(\lambda)}(x_2)\\ \vdots\\ \beta_1+\beta_2 C_1^{(\lambda)}(x_N)\end{pmatrix}, \tag{7.3}$$

where F is the diagonal matrix with diagonal elements x_i, $i = 0, 1, \ldots, N$.

Re-sort the unknowns in (7.3) by the form $\beta_1, u(x_1), u(x_2), \ldots, u(x_{N-1}), \beta_2$ to get a coefficient matrix similar to that obtained from (7.3) after replacing the first and last columns with corresponding columns in the matrix A.

Tables 2, 4, and 6 contain the results of our experiments for the Chebyshev basis and $\epsilon = 1$, $\epsilon = 10^{-1}$, and $\epsilon = 10^{-2}$, respectively. Tables 3, 5, and 7 contain the results for the Legendre basis and $\epsilon = 1$, $\epsilon = 10^{-1}$, and $\epsilon = 10^{-2}$, respectively.

TABLE 2
Numerical results for $\epsilon = 1$ (Chebyshev basis).

	ChP		*PChP*	
N	Condition number	Maximal error	Condition number	Maximal error
32	1.72×10^{04}	1.79×10^{-14}	19.12	6.66×10^{-16}
64	2.74×10^{05}	7.14×10^{-14}	37.32	9.99×10^{-16}
128	4.39×10^{06}	5.74×10^{-13}	73.67	1.78×10^{-15}
256	7.02×10^{07}	2.22×10^{-12}	146.37	2.99×10^{-15}
512	1.12×10^{09}	9.52×10^{-12}	291.76	3.89×10^{-15}
1024	1.80×10^{10}	4.61×10^{-11}	582.53	7.44×10^{-15}

TABLE 3
Numerical results for $\epsilon = 1$ (Legendre basis).

	LP		*PLP*	
N	Condition number	Maximal error	Condition number	Maximal error
32	9.76×10^{03}	2.07×10^{-14}	18.97	7.77×10^{-16}
64	1.51×10^{05}	1.65×10^{-13}	37.16	9.99×10^{-16}
128	2.37×10^{06}	5.46×10^{-13}	75.51	1.33×10^{-15}
256	3.76×10^{07}	1.40×10^{-12}	146.20	2.55×10^{-15}
512	5.99×10^{08}	1.96×10^{-11}	291.59	3.11×10^{-15}
1024	7.21×10^{09}	3.21×10^{-11}	582.21	6.81×10^{-15}

8. Conclusion. The method discussed in this article is quite useful for deriving efficient algorithms for treatment of one dimensional boundary value problems. The present approach is a point space analogue of the spectral method in [5]. It chooses a definite integral that fixes the arbitrary constants of integration by bringing in the lowest modes through adjustable parameters. Namely, the additional parameters are exactly the coefficients of the k lowest modes in the expansion of the solution. The differentiation nullifies the lowest terms in the expansion, and the additional equations

TABLE 4
Numerical results for $\epsilon = 10^{-1}$ (Chebyshev basis).

	ChP		*PChP*	
N	Condition number	Maximal error	Condition number	Maximal error
64	9.41×10^{04}	3.42×10^{-14}	196.54	2.43×10^{-15}
128	1.51×10^{06}	2.13×10^{-13}	390.29	5.38×10^{-14}
256	2.41×10^{07}	9.10×10^{-13}	777.99	1.50×10^{-14}
512	3.85×10^{08}	3.33×10^{-12}	1.55×10^{03}	1.96×10^{-14}
1024	6.17×10^{09}	1.49×10^{-11}	3.10×10^{03}	3.62×10^{-14}

TABLE 5
Numerical results for $\epsilon = 10^{-1}$ (Legendre basis).

	LP		*PLP*	
N	Condition number	Maximal error	Condition number	Maximal error
64	5.18×10^{04}	1.59×10^{-13}	193.41	9.99×10^{-16}
128	8.14×10^{05}	9.82×10^{-14}	387.04	1.22×10^{-15}
256	1.29×10^{07}	6.61×10^{-13}	774.68	1.67×10^{-15}
512	2.06×10^{08}	1.11×10^{-11}	1.55×10^{03}	2.22×10^{-15}
1024	1.10×10^{09}	2.72×10^{-11}	3.13×10^{03}	3.71×10^{-15}

TABLE 6
Numerical results for $\epsilon = 10^{-2}$ (Chebyshev basis).

	ChP		*PChP*	
N	Condition number	Maximal error	Condition number	Maximal error
64	1.19×10^{04}	5.44×10^{-10}	2.15×10^{03}	4.88×10^{-10}
128	1.92×10^{05}	1.14×10^{-13}	2.20×10^{03}	7.01×10^{-14}
256	3.07×10^{06}	5.25×10^{-13}	4.34×10^{03}	1.71×10^{-13}
512	4.93×10^{07}	1.56×10^{-12}	8.64×10^{03}	2.04×10^{-13}
1024	7.89×10^{08}	7.71×10^{-12}	1.73×10^{03}	3.69×10^{-13}

TABLE 7
Numerical results for $\epsilon = 10^{-2}$ (Legendre basis).

	LP		*PLP*	
N	Condition number	Maximal error	Condition number	Maximal error
64	6.79×10^{03}	3.70×10^{-10}	1.45×10^{03}	2.71×10^{-10}
128	1.05×10^{05}	1.07×10^{-13}	2.13×10^{03}	4.11×10^{-15}
256	1.65×10^{06}	7.19×10^{-13}	4.27×10^{03}	6.77×10^{-15}
512	2.63×10^{07}	1.14×10^{-11}	8.57×10^{03}	1.07×10^{-14}
1024	3.28×10^{08}	3.13×10^{-11}	1.62×10^{04}	2.13×10^{-14}

introduced are simply the leading lines in the transform matrix which produces these coefficients. These parameters are finally set to satisfy the boundary conditions. The method is shown to work well on a test problem involving a stiff ordinary differential equation. The higher dimensional problems require further study in order to establish the usefulness of the pseudospectral integration preconditioners within a more general context.

REFERENCES

[1] M. Abramowitz and I. A. Stegun, *Handbook of Mathematical Functions with Formulas, Graphs and Mathematical Tables*, Dover, New York, 1972.

[2] C. Canuto and A. Quarteroni, *Preconditioned minimal residual methods for Chebyshev spectral calculations*, J. Comput. Phys., 60 (1985), pp. 315–337.

[3] C. W. Clenshaw, *The numerical solution of linear differential equations in Chebyshev series*, Proc. Cambridge Philos. Soc., 53 (1957), pp. 134–149.

[4] C. W. Clenshaw and A. R. Curtis, *A method for numerical integration on an automatic computer*, Numer. Math., 2 (1960), pp. 197–205.

[5] E. A. Coutsias, T. Hagstrom, and D. Torres, *An efficient spectral method for ordinary differential equations with rational function coefficients*, Math. Comp., 65 (1996), pp. 611–635.

[6] M. Deville and E. Mund, *Chebyshev pseudospectral solution of second-order elliptic equations with finite element preconditioning*, J. Comput. Phys., 60 (1985), pp. 517–533.

[7] M. O. Deville and E. H. Mund, *Finite-element preconditioning for pseudospectral solutions of elliptic problems*, SIAM J. Sci. Statist. Comput., 11 (1990), pp. 311–342.

[8] S. E. El-Gendi, *Chebyshev solution of differential, integral, and integro-differential equations*, Comput. J., 12 (1969), pp. 282–287.

[9] L. Fox and I. B. Parker, *Chebyshev Polynomials in Numerical Analysis*, Oxford University Press, London, 1968.

[10] D. Funaro, *A preconditioning matrix for the Chebyshev differencing operator*, SIAM J. Numer. Anal., 24 (1987), pp. 1024–1031.

[11] D. Funaro and D. Gottlieb, *A new method of imposing boundary conditions in pseudospectral approximations of hyperbolic equations*, Math. Comp., 51 (1988), pp. 599–613.

[12] D. Funaro and D. Gottlieb, *Convergence results for pseudospectral approximations of hyperbolic systems by a penalty-type boundary treatment*, Math. Comp., 57 (1991), pp. 585–596.

[13] F. Ghoreishi and S. M. Hosseini, *Preconditioned implementation of pseudospectral methods on arbitrary grids*, Appl. Math. Comput., 148 (2004), pp. 15–34.

[14] L. Greengard, *Spectral integration and two-point boundary value problems*, SIAM J. Numer. Anal., 28 (1991), pp. 1071–1080.

[15] D. Hatziavramidis and H.-C. Ku, *An integral Chebyshev expansion method for boundary value problems of O.D.E. type*, Comput. Math. Appl., 11 (1985), pp. 581–586.

[16] J. S. Hesthaven and D. Gottlieb, *A stable penalty method for the compressible Navier–Stokes equations:* I. *Open boundary conditions*, SIAM J. Sci. Comput., 17 (1996), pp. 579–612.

[17] J. S. Hesthaven, *A stable penalty method for the compressible Navier–Stokes equations:* II. *One-dimensional domain decomposition schemes*, SIAM J. Sci. Comput., 18 (1997), pp. 658–685.

[18] J. S. Hesthaven, *Integration preconditioning of pseudospectral operators.* I. *Basic linear operators*, SIAM J. Numer. Anal., 35 (1998), pp. 1571–1593.

[19] J. S. Hesthaven, *A stable penalty method for the compressible Navier–Stokes equations:* III. *Multidimensional domain decomposition schemes*, SIAM J. Sci. Comput., 20 (1999), pp. 62–93.

[20] J. S. Hesthaven, *Spectral penalty methods*, Appl. Numer. Math., 33 (2000), pp. 23–41.

[21] H. Jeffreys and B. S. Jeffreys, *Methods of Mathematical Physics*, Cambridge University Press, London, 1956.

[22] S. D. Kim and S. V. Parter, *Preconditioning Chebyshev spectral collocation method for elliptic partial differential equations*, SIAM J. Numer. Anal., 33 (1996), pp. 2375–2400.

[23] S. D. Kim and S. V. Parter, *Preconditioning Chebyshev spectral collocation by finite-difference operators*, SIAM J. Numer. Anal., 34 (1997), pp. 939–958.

[24] B. Mihaila and I. Mihaila, *Numerical approximations using Chebyshev polynomial expansions: El-gendi's method revisited*, J. Phys. A, 35 (2002), pp. 731–746.

[25] H. Nasr, I. A. Hassanien, and H. M. El-Hawary, *Chebyshev solution laminar boundary layer flow*, Int. J. Comput. Math., 33 (1990), pp. 127–132.

[26] H. Nasr and H. M. El-Hawary, *Chebyshev method for the solution of boundary value problems*, Int. J. Comput. Math., 40 (1991), pp. 251–258.

[27] S. A. Orszag, *Spectral methods for problems in complex geometries*, J. Comput. Phys., 37 (1980), pp. 70–92.

[28] A. Pinelli, C. Benocci, and M. Deville, *Chebyshev pseudospectral solution of advection-diffusion equations with mapped finite difference preconditioning*, J. Comput. Phys., 112 (1994), pp. 1–11.

[29] J. Shen, F. Wang, and J. Xu, *A finite element multigrid preconditioner for Chebyshev-collocation methods*, Appl. Numer. Math., 33 (2000), pp. 471–477.
[30] G. Szego, *Orthogonal Polynomials*, 4th ed., Amer. Math. Soc. Colloq. Publ. 23, AMS, Providence, RI, 1975.
[31] A. Zebib, *A Chebyshev method for the solution of boundary value problems*, J. Comput. Phys., 53 (1984), pp. 443–455.

SIAM J. SCI. COMPUT.
Vol. 28, No. 4, pp. 1203–1227
© 2006 Society for Industrial and Applied Mathematics

LAGRANGIAN NUMERICAL APPROXIMATIONS TO ONE-DIMENSIONAL CONVOLUTION-DIFFUSION EQUATIONS*

LAURENT GOSSE† AND GIUSEPPE TOSCANI‡

Abstract. This work focuses on the numerical analysis of one-dimensional nonlinear diffusion equations involving a convolution product. First, homogeneous friction equations are considered. Algorithms follow recent ideas on mass transportation methods and lead to simple schemes which can be proved to be stable, to decrease entropy, and to converge toward the unique solution of the continuous problem. In particular, for the first time, homogeneous cooling states are displayed numerically. Further, we present results on the more delicate fourth-order thin-film equation for which a nonnegativity-preserving scheme is derived. The dead core phenomenon is presented for the Hele–Shaw cell.

Key words. granular flows, friction equations, Lagrangian approximation, Wasserstein metric, homogeneous cooling states, Hele–Shaw cell

AMS subject classifications. Primary, 65M06, 65M12; Secondary, 76T25, 76X05, 76D08

DOI. 10.1137/050628015

1. Introduction. One objective for the present paper is to study some constructive approximations for one-dimensional (1D) dissipative Boltzmann equations modeling granular flows; they read as

$$\partial_t f + v\partial_x f = Q(f,f) + R, \qquad t,x,v \in \mathbb{R}_*^+ \times \mathbb{R} \times \mathbb{R},$$

for kinetic densities $f(t,x,v) \geq 0$. Here $Q(f,f)$ stands for the term modeling the inelastic collisions [1, 2, 27, 30], whereas R renders the effects of, e.g., external heating processes, such as Brownian agitation, which leads to a diffusion term; see [11] for a large survey on granular materials. First, it is customary now to decouple the transport and collision steps relying on a time-splitting technique (see, e.g., [28, 26, 18]); thus it makes sense to restrict ourselves to homogeneous densities $f(t,v)$. We shall therefore concentrate on the following Cauchy problem for $t,v \in \mathbb{R}_*^+ \times \mathbb{R}$:

$$\partial_t f(t,v) = \partial_v \left\{ f(t,v) \int_{\mathbb{R}} \mathcal{I}'(v-\omega) f(t,\omega).d\omega \right\} + \mu\partial_v(vf(t,v)) + \sigma\partial_{vv} f(t,v), \tag{1.1}$$

where $\mathcal{I}$ stands for an interaction potential. Generally, it is taken as $\mathcal{I}(\xi) = |\xi|^{\gamma+2}/\gamma + 2$, $\gamma \geq 0$. Equation (1.1) rewrites as a convolution Fokker–Planck equation:

$$\partial_t f = \partial_v \big\{ f\partial_v \big(f *_v \mathcal{I}(v) \big) + \sigma\partial_v f \big\} + \mu\partial_v(vf).$$

Several particular cases of (1.1) are of interest; the first one corresponds to setting $\sigma = \mu = 0$ and models dissipative flows of granular media. This model was proposed by McNamara and Young [27] and then studied from a mathematical viewpoint in

*Received by the editors March 29, 2005; accepted for publication (in revised form) December 16, 2005; published electronically July 31, 2006. This work was supported by the Wolfgang Pauli Institute (Wien) in the frame of the "Nanoscience" Programme and the European network HYKE, funded by the EC under contract HPRN-CT-2002-00282.

http://www.siam.org/journals/sisc/28-4/62801.html

†Istituto per le Applicazioni del Calcolo (sezione di Bari), Via G. Amendola 122, 70126 Bari, Italy (l.gosse@ba.iac.cnr.it).

‡Dipartimento di Matematica, Universita di Pavia, Via Ferrata 1, 27100 Pavia, Italy (toscani@dimat.unipv.it).

[1, 2, 7, 13, 14]. Roughly speaking, it describes a huge number of particles that undergo inelastic collisions (see [30, 26]) which lead to a monotonic decay of the temperature. Steady states display a very elementary structure, namely a Dirac mass concentrated onto the null velocity; however, a rescaling process allows us to derive a particular class of intermediate asymptotics called [illegible] [2, 24, 30]. These states have been singled out mainly because they are minimizers of a "free energy functional" (which is to be intensively used in section 3.1)

$$\mathcal{J}(f) := \int_{\mathbb{R}^2} f(v)f(w)\mathcal{I}(v-w).dv.dw - \int_{\mathbb{R}} f(v)v^2.dv, \tag{1.2}$$

among probability measures on $\mathbb{R}$. However, although stability could be hoped for, it has recently been shown that these states are too poor to play a role of similar importance as Barenblatt–Pattle source solutions for porous media equations; detailed proofs are available in [10]. It will be one of our objectives to display numerically such homogeneous cooling states in section 3.2; to the authors' knowledge, this has not been performed before.

Another objective is to develop on the numerical analysis of the so-called lubrication approximation of the Navier–Stokes equations for the movement of thin films driven by surface tension (see [25]); such a derivation leads to a fourth-order Cauchy problem,

$$\partial_t u + \partial_x(a(u)\partial_{xxx}u) = 0, \qquad u(t=0,.) = u_0 \geq 0, \quad t,x \in \mathbb{R}_*^+ \times \mathbb{R}. \tag{1.3}$$

The theory for this equation is still sparse (see [3, 4, 5, 6, 16]); especially, little is known about uniqueness of strong, i.e., $H^2(\mathbb{R})$ nonnegative, solutions. A special and somewhat more tractable case of (1.3) corresponds to $a(u) = u$ and is generally referred to as the Hele–Shaw cell: we shall focus on it in section 5, where certain ideas proposed in [29] will be expanded. Let us first explain why this problem can fall inside the class of "convolutive diffusion equations." Our main trick is to consider, for a small number $0 < \varepsilon \ll 1$, a smoothered equation,

$$\partial_t u^\varepsilon + \partial_x\big(a(u^\varepsilon)\partial_{xxx}u^\varepsilon *_x \Gamma_\varepsilon(x)\big) = \partial_t u^\varepsilon + \partial_x\big(a(u^\varepsilon)u^\varepsilon *_x \Gamma_\varepsilon'''(x)\big) = 0,$$

where Γ_ε is a standard Friedrichs mollifier. At this point, one can derive an approximate problem for (1.3) written in the convenient form for $t,x \in \mathbb{R}_*^+ \times \mathbb{R}$,

$$\partial_t u^\varepsilon + \partial_x\big(a(u^\varepsilon)\partial_x(u^\varepsilon *_x \Gamma_\varepsilon''(x))\big) = 0, \qquad u(t=0,.) = u_0 \geq 0. \tag{1.4}$$

However, even if we share some features with the aforementioned models of granular flows (especially for $a(u) = u$), a major difficulty arises from the fact that $-\Gamma_\varepsilon''$ will never be a convex function; it is nonetheless possible to obtain a numerical process able to preserve nonnegativity of solutions, a property commonly considered important in practice [34, 21].

Going a bit deeper into the contents of this work, we aim at showing that modern Wasserstein techniques can be successful at producing simple numerical schemes for (1.1) which are positivity-preserving, stable, and consistent with the continuous problem. Indeed, it has been shown in [15], and we shall briefly recall in section 2.1 how one can pass by means of a Lagrangian-type change of variables from (1.1) to the much more tractable equation holding for $t, \varrho \in \mathbb{R}_*^+ \times (0,1)$,

$$\partial_t V(t,\varrho) + \sigma\partial_\varrho\left(\frac{1}{\partial_\varrho V(t,\varrho)}\right) + \mu V(t,\varrho) + \int_0^1 \mathcal{I}'\big(V(t,\varrho) - V(t,\varrho')\big).d\varrho' = 0, \tag{1.5}$$

which reduces, in the case $\sigma = \mu = 0$, to some sort of integro-differential equation. This trick is reminiscent of a previous work [19] and allows us to circumvent the issues which are connected to numerical approximation of probability measures with finite differences; see [9, 17, 26] for more in this direction. One feature of paramount importance for (1.5) is the easiness in proving contractivity in any p-Wasserstein metric for various numerical approximations, just relying on the standard Jensen's inequality. It is also not difficult to convert such an estimate into a BV bound, thus showing compactness for a sequence of approximations for the V variable. Such a program is to be realized in sections 2.2–2.3. The BV framework (functions of bounded variation) corresponds exactly to measure solutions for (1.1), and hence provides a convenient approach to the problem. Section 4 is concerned with the modifications brought by the immersion of such granular media inside a thermal fluid. Following [12], we are most of all interested in classical solutions even if we shall keep on using numerical schemes based on the "reciprocal equation" (1.5). Convergence is shown in section 4.1, whereas the "tails" of the solutions for large $|v|$ (see [2, 7, 8, 9, 13]) are to be studied numerically in section 4.2. Section 5 is entirely devoted to the study of the Hele–Shaw cell's equation for which we can propose a nonnegativity-preserving scheme based on similar ideas. An energy estimate is given in section 5.2, and some numerical results are shown in section 5.3; an extension to mobilities $a(u) \neq u$ is briefly discussed in Remark 2 and in more detail in section 6. Finally, concluding remarks are given in section 7, and Appendices A and B contain the proofs of two important contraction estimates. Appendix C deals with some special features of the Fourier scheme (6.2) for (1.3) with $a(u) = u$.

2. Lagrangian approximation of 1D friction equations; $\sigma = \mu = 0$. The object under the scope is now

$$\partial_t f(t,v) = \partial_v \left\{ f(t,v) \int_{\mathbb{R}} \mathcal{I}'(v-\omega) f(t,\omega).d\omega \right\}; \qquad t, v \in \mathbb{R}_*^+ \times \mathbb{R}, \tag{2.1}$$

as proposed in [27] from the Boltzmann equation in a so-called quasi-elastic limit in the case $\mathcal{I}(\xi) = |\xi|^3/3$, i.e., $\gamma = 1$. We use the notation $\mathbb{R}_*^+ = (0, +\infty)$ throughout the paper. Since (2.1) is known to admit Dirac-type similarity solutions, it is convenient to introduce the following functional framework for $p \geq 1$:

$$\mathcal{M}_p(\mathbb{R}) = \left\{ v \mapsto f(.,v) \text{ probability measure such that } \int_{\mathbb{R}} |v|^p f(.,v).dv < +\infty \right\},$$

equipped with the topology of weak convergence of measures. A distinguished class of metrics on $\mathcal{M}_p(\mathbb{R})$ is given by the so-called Wasserstein (or Monge–Kantorovitch) distance [19, 24, 32], which can be written as the usual L^p distance of the [illegible] (to be defined in section 2.1),

$$d_p(f,g) := \left(\int_0^1 |V(.,\varrho) - W(.,\varrho)|^p .d\varrho \right)^{1/p}, \qquad 1 \leq p < +\infty.$$

By a weak solution to the Cauchy problem for (2.1), we mean any $f \in C^1(\mathbb{R}_*^+; \mathcal{M}_p(\mathbb{R}))$ satisfying for all $\varphi \in C^1(\mathbb{R})$ and $t > 0$,

$$\begin{aligned} \frac{d}{dt} \int_{\mathbb{R}} \varphi(v) f(t,v).dv &= - \int_{\mathbb{R}^2} \mathcal{I}'(v-w) \varphi'(v) f(t,v) f(t,w).dv.dw \\ &= \frac{1}{2} \int_{\mathbb{R}^2} \mathcal{I}'(v-w)(\varphi'(w) - \varphi'(v)) f(t,v) f(t,w).dv.dw, \end{aligned} \tag{2.2}$$

and $f(t,.) \to f(t=0,.) = f_0 \in \mathcal{M}_p(\mathbb{R})$ weakly as a measure as $t \to 0$. The second equality holds because $\mathcal{I}'$ is an odd function. Obviously, choosing $\varphi(v) = 1, v$ in (2.2) yields conservation of mass and momentum; thus it makes sense to restrict ourselves to centered probability measures. Further, inserting $\varphi(v) = v^2$ leads to

$$\frac{d}{dt}\theta(t) := \frac{d}{dt}\int_{\mathbb{R}^2} v^2 f(t,v).dv = -\int_{\mathbb{R}^2} \mathcal{I}'(v-w)(v-w)f(t,v)f(t,w).dv.dw \leq 0.$$

Thus the temperature is meant to decrease monotonically in time (at most at an exponential rate; see [30] and Figure 1) because $\mathcal{I}'$ is monotone nondecreasing. We also deduce from [24] the following uniqueness theorem.

THEOREM 2.1. [illegible] $f_0 \in \mathcal{M}_p(\mathbb{R})$ [illegible] $p = 2$ [illegible] $\mathcal{I}$ [illegible] (2.1) [illegible]

The proof is given in Appendix A.

2.1. The reciprocal mapping. From [10], the decay towards stable similarity solutions (homogeneous cooling states) can be expected to be slow, and because of their singular structure, one cannot use high-order discretizations. This clearly constitutes a numerical difficulty we propose to overcome by means of the change of variable described as follows:

- Let us introduce the distribution function associated with the initial probability measure f_0,

$$\varrho_0(v) = \int_{-\infty}^{v} f_0(\omega).d\omega \in (0,1), \qquad \varrho_0 \in BV(\mathbb{R}),$$

which is obviously nondecreasing in the v variable. We can thus define a (nondecreasing) pseudoinverse:

$$\begin{aligned} v_0 : (0,1) &\to \mathbb{R}, \\ \bar{\varrho} &\mapsto v_0(\bar{\varrho}) := \inf\{\omega \in \mathbb{R} \text{ such that } \varrho_0(\omega) = \bar{\varrho}\}. \end{aligned} \tag{2.3}$$

- For any given value $\bar{\varrho} \in (0,1)$, we can define the [illegible]

$$\begin{aligned} V : \quad \mathbb{R}^+ &\to \mathbb{R}, \\ t &\mapsto V(t,\bar{\varrho}), \end{aligned}$$

by means of the implicit function theorem in case $\partial_v \varrho \neq 0$, such that

$$V(t=0,\bar{\varrho}) = v_0(\bar{\varrho}), \qquad \varrho(t, V(t,\bar{\varrho})) = \bar{\varrho}. \tag{2.4}$$

Integrating (2.1) in v,

$$\partial_t \varrho(t,v) = \partial_v \varrho(t,v)\int_{\mathbb{R}} \mathcal{I}'(v-\omega)\partial_v \varrho(t,\omega).d\omega, \qquad \partial_v \varrho(t,v) = f(t,v),$$

and differentiating the second equality of (2.4) with respect to time, one deduces easily the time evolution of $V(.,\varrho)$ (we drop the $\bar{}$ for ease of reading); see also [15, 24]:

$$\partial_t V(t,\varrho) + \int_0^1 \mathcal{I}'\big(V(t,\varrho) - V(t,\varrho')\big).d\varrho' = 0, \qquad t > 0, \tag{2.5}$$

and $V(t=0,\varrho) = v_0(\varrho)$ for any $\varrho \in (0,1)$.

We define a Cartesian computational grid by means of the two positive parameters $\Delta t, \Delta v$. From now on, let v_k and t^n stand for $k\Delta v$ and $n\Delta t$, respectively. Therefore our numerical approach to (2.1) with convenient (unit mass, centered) initial data is built on computing the pseudoinverse of ϱ_0, $V(t=0,.)$, evolving it in time by means

of an [illegible] marching scheme for (2.5) in order to deduce the values of $\varrho(t, X(t,.)) \in [0,1]$ thanks to (2.4). Working on this pseudoinverse $V(t,.)$ allows us to bypass the usual problems in approximating Dirac measures [9, 17]. This trick can be used mainly because (2.1) is [illegible]. It will prove useful to introduce the [illegible] definition of weak solution to (2.1) (see [24, p. 412]).

DEFINITION 1. [illegible] $f_0 \in \mathcal{M}_p(\mathbb{R})$. $f(t,v) \in C^1(\mathbb{R}_*^+; \mathcal{M}_p(\mathbb{R}))$ [illegible] (2.1), (2.5) [illegible] $\varphi \in C^1(\mathbb{R})$ [illegible] $t > 0$

$$\frac{d}{dt}\int_0^1 \varphi(V(t,\varrho)).d\varrho = -\int_{(0,1)^2} \mathcal{I}'(V(t,\varrho) - V(t,\varrho'))\varphi'(V(t,\varrho)).d\varrho.d\varrho'$$
$$= -\int_{(0,1)^2} \mathcal{I}'(V(t,\varrho) - V(t,\varrho'))\frac{\varphi'(V(t,\varrho) - \varphi'(V(t,\varrho')}{2}.d\varrho.d\varrho'.$$

[illegible] $f(t,.) \to f_0$ [illegible] $t \to 0$ [illegible]

We now discretize the ϱ and t axes and define

$$V_k^n \simeq V(t^n, \varrho_k); \qquad \varrho_k := \varrho_0(v_k),\ t^n = n\Delta t \quad \text{for } k \in \mathcal{K} \subset \mathbb{N},\ n \in \mathbb{N}. \tag{2.6}$$

A numerical scheme for (2.5) reads as

$$V_k^{n+1} = V_k^n - \Delta t \sum_{k' \in \mathcal{K}} |C_{k'}| \mathcal{I}'(V_k^n - V_{k'}^n), \tag{2.7}$$

where $|C_k| = \varrho_{k+\frac{1}{2}} - \varrho_{k-\frac{1}{2}}$ stands for the width of the control cell centered on ϱ_k with $\varrho_{k+\frac{1}{2}} = \varrho_0(x_{k+\frac{1}{2}})$. As ϱ_0 is at least of bounded variation (hence it makes sense to speak about left/right values at a given point), a convenient choice is given by linear interpolation, $\varrho_{k+\frac{1}{2}} = \frac{1}{2}(\varrho_k + \varrho_{k+1})$, which yields $|C_k| = \frac{1}{2}\big(\varrho_{k+1} - \varrho_{k-1}\big)$. Since no divided differences show up in (2.7), there is no need for boundary conditions. The scheme (2.7) is conservative [illegible]. This furthermore yields, since $\mathcal{I}$ is an even function,

$$\forall n \in \mathbb{N}, \qquad \sum_k |C_k| V_k^n = \sum_k |C_k| V_k^0 \simeq \int_0^1 V(t=0,\varrho).d\varrho = \int_{\mathbb{R}} v.f_0(x).dv.$$

We stress that the ϱ_k's do [illegible] depend on time. In order to reconstruct $\tilde{\varrho}(t^n,.)$, an approximation of $\varrho(t,.)$ at a given time $t \simeq t^n$, one has to interpolate the family of numerical values ϱ_k, X_k^n, t^n since

$$\tilde{\varrho}(t^n, V_k^n) \overset{def}{=} \varrho_k \simeq \varrho(t^n, V_k^n),$$

$\partial_v \varrho$ being the weak solution to (2.1), up to the numerical truncation errors on V_k^n coming from the discretization (2.7). Then one deduces $f(t^n,.)$ by, e.g., centered divided differences.

2.2. Wasserstein contraction estimate. The fundamental lemma reads as follows.

LEMMA 2.2. [illegible] f_0, g_0 [illegible] $\mathcal{M}_p(\mathbb{R})$ [illegible] (2.1) [illegible] V, W [illegible]

$$\Delta t \sup_{V_k^0, W_{k'}^0} \big(\mathcal{I}''\big) \le 1, \qquad k, k' \in \mathcal{K}^2, \tag{2.8}$$

[illegible] (2.7) [illegible] d_p [illegible]

$$\forall n \in \mathbb{N}, \qquad \sum_k |C_k| |V_k^{n+1} - W_k^{n+1}|^p \le \sum_k |C_k| |V_k^n - W_k^n|^p, \qquad p \ge 1. \tag{2.9}$$

In practice, for (2.8), the supremum can be taken on the convex hull of the given set of data.

. We readily substract the schemes (2.7) on each variable to derive

$$\forall n \in \mathbb{N}, \qquad V_k^{n+1} - W_k^{n+1} = V_k^n - W_k^n - \Delta t \sum_{k' \in \mathcal{K}} |C_{k'}| \Big(\mathcal{I}'(V_k^n - V_{k'}^n) - \mathcal{I}'(W_k^n - W_{k'}^n) \Big).$$

We apply the mean-value theorem for each k, k' to obtain

$$\mathcal{I}'(V_k^n - V_{k'}^n) - \mathcal{I}'(W_k^n - W_{k'}^n) = (\mathcal{I}'')_{k,k'}^n \Big((V_k^n - W_k^n) - (V_{k'}^n - W_{k'}^n) \Big),$$

where convexity manifests itself through

$$0 \leq (\mathcal{I}'')_{k,k'}^n := \mathcal{I}'' \Big(\lambda (V_k^n - V_{k'}^n) + (1-\lambda)(W_k^n - W_{k'}^n) \Big),$$

for some $\lambda \in [0,1]$. Hence rearranging terms, one finds out that $V_k^{n+1} - W_k^{n+1}$ is a convex combination of each $V_{k'}^n - W_{k'}^n$, $k' \in \mathcal{K}$, under the CFL condition (2.8) since $\mathcal{I}$ is convex:

$$V_k^{n+1} - W_k^{n+1} = (V_k^n - W_k^n) \Big(1 - \Delta t \sum_{k' \in \mathcal{K}} |C_{k'}| (\mathcal{I}'')_{k,k'}^n \Big) + \Delta t \sum_{k' \in \mathcal{K}} |C_{k'}| (\mathcal{I}'')_{k,k'}^n (V_{k'}^n - W_{k'}^n).$$

The result (2.9) follows from summing on $k \in \mathcal{K}$ and invoking Jensen's inequality. ∎

This result is but a numerical equivalent of Theorem 1 in [24]. As an immediate consequence of Lemma 2.2, one obtains the L^p stability of the method, just taking $W_k^0 \equiv 0$ (which implies $W_k^n \equiv 0$ since $\mathcal{I}'(0) = 0$).

PROPOSITION 2.3. 2.2,

$$\forall n \in \mathbb{N}, \qquad \sum_k |C_k| \Big(|V_k^{n+1}|^p - |V_k^n|^p \Big) \leq 0, \qquad p \geq 1.$$

This somehow highlights the solution's support decay proved in [24]. It shows in particular the constant decrease of the numerical solution's temperature since $\theta(t^n) \simeq \sum_k |C_k| |V_k^n|^2$. Clearly, more involved treatments of the integral term of (2.5) would yield an estimate of the same flavor. It would also be possible to consider higher-order Runge–Kutta methods in time; see, e.g., [20].

COROLLARY 2.4. 2.2, f_0

$$\forall k, n \in \mathcal{K} \times \mathbb{N}, \qquad |V_k^{n+1} - V_k^n| \leq \Delta t \sup_{V_k^0} |\mathcal{I}'|. \tag{2.10}$$

. From Proposition 2.3, the sequence V_k^n is bounded in $L^\infty(0,1)$. The estimate (2.10) follows from $\sum_{k \in \mathcal{K}} |C_k| = \|f_0\|_{L^1(\mathbb{R})} = 1$. ∎

2.3. Compactness and entropy consistency of the scheme. By analogy with (1.2), we can define an "entropy" functional for (2.1),

$$\forall t \geq 0, \qquad \mathcal{E}(t,f) = \int_{\mathbb{R}^2} \mathcal{I}(v - v') f(t,v) f(t,v') . dv . dv',$$

which is known to decrease as times grow at least for certain potentials. It can be easily checked that (2.7) keeps this property; this is the purpose of the next result.

PROPOSITION 2.5. $\mathcal{E}^n = \sum_{k,k'} |C_k||C_{k'}|\mathcal{I}(V_k^n - V_{k'}^n)|$, $\mathcal{I}(\xi) = |\xi|^{2+\gamma}/\gamma+2$ 2.2,

$$\forall n \in \mathbb{N}, \qquad \mathcal{E}^n \leq \Big(1 - \Delta t \inf_{V_k^0}(\mathcal{I}'')\Big)^{n(\gamma+2)} \mathcal{E}^0. \tag{2.11}$$

. The proof proceeds as in Lemma 2.2; substracting and linearizing (2.7) with $W_k^n = V_\ell^n$ for some $\ell \in \mathcal{K}$ gives

$$\begin{aligned}
|V_k^{n+1} - V_\ell^{n+1}| &= \Big|V_k^n - V_\ell^n - \Delta t \sum_{k' \in \mathcal{K}} \Big|C_{k'}\big|\big(\mathcal{I}'(V_k^n - V_{k'}^n) - \mathcal{I}'(V_\ell^n - V_{k'}^n)\big)\big| \\
&= \Big|V_k^n - V_\ell^n - \Delta t \sum_{k' \in \mathcal{K}} \Big|C_{k'}\big|(\mathcal{I}'')_{k,\ell,k'}^n\big((V_k^n - V_\ell^n) - (V_{k'}^n - V_{k'}^n)\big)\big| \\
&= |V_k^n - V_\ell^n|\Big(1 - \Delta t \sum_{k' \in \mathcal{K}} |C_{k'}|(\mathcal{I}'')_{k,\ell,k'}^n\Big),
\end{aligned}$$

where $(\mathcal{I}'')_{k,\ell,k'}^n = \mathcal{I}''(\lambda V_k^n + (1-\lambda)V_\ell^n - V_{k'}^n)$ and from which we deduce that $|V_k^{n+1} - V_\ell^{n+1}| \leq |V_k^n - V_\ell^n|(1 - \Delta t \inf_{V_k^n}(\mathcal{I}''))$ under the CFL condition (2.8). Summing leads to (2.11). Thus the proof is complete. □

Of course, if one lets $\Delta t \to 0$, (2.11) provides the exponential decay of the entropy; see [24]. Lemma 2.2 also implies a bound on the total variation of the scheme, as we show now.

COROLLARY 2.6. 2.2,

$$\forall n \in \mathbb{N}, \qquad \sum_{k \geq 1} |V_k^n - V_{k-1}^n| \leq \Big(1 - \Delta t \inf_{V_k^0}(\mathcal{I}'')\Big)^n \sum_{k \geq 1} |V_k^0 - V_{k-1}^0|. \tag{2.12}$$

(2.7)

. The proof is essentially the same as in Proposition 2.5; substracting and linearizing (2.7) with $\ell = k-1$ gives again $|V_k^{n+1} - V_{k-1}^{n+1}| \leq |V_k^n - V_{k-1}^n|\big(1 - \Delta t \inf_{V_k^n}(\mathcal{I}'')\big)$ using the CFL condition (2.8). This also implies monotonicity preservation. By summing for $k, k' \geq 1$, one obtains the decrease of the total variation which is initially bounded if f_0 has compact support. □

. 1. One clearly sees that the CFL (2.8) condition is indeed very mild in this case, since the values of Δt and $|C_k|$ are completely decorrelated. In particular, no implicit time integrator is needed to ensure stability, in sharp contrast with [9].

The decrease of the total variation was to be expected; one can imagine considering $V(t=0,\varrho)$ as an initial value and $W(t=0,\varrho) := V(t=0,\varrho-h)$, $0 < h \ll 1$. Thus the Wasserstein contraction (2.9) ensures that the corresponding numerical approximations (2.7) will get closer to each other as times grow. For $p = 1$, this is exactly what is meant by (2.12). Eventually, and at least for $\mathcal{I}(\xi) = |\xi|^{2+\gamma}/\gamma+2$, both V and W will get stuck onto a self-similar profile which depends only on the value of $\gamma \geq 0$. We shall return to this in section 3.

At this level, it is convenient to define a sequence of approximate "Lagrangian solutions" $V^{\Delta t} \in BV_{loc}(\mathbb{R}_*^+ \times (0,1))$ as for all $t, \varrho \in \mathbb{R}_*^+ \times (0,1)$,

$$V^{\Delta t}(t,\varrho) = V_k^n \text{ if } |t - t^n| \leq \frac{\Delta t}{2} \text{ and } |\varrho - \varrho_k| \leq \frac{|C_k|}{2}. \tag{2.13}$$

Obviously, this implies the existence of another sequence $\varrho^{\Delta t}(t,v)$ defined as follows: for all $t, v \in \mathbb{R}_*^+ \times \mathbb{R}$,

$$\varrho^{\Delta t}(t,v) = \varrho_k \text{ if } |t - t^n| \leq \frac{\Delta t}{2} \text{ and } \frac{V_{k-1}^n - V_k^n}{2} \leq v - V_k^n \leq \frac{V_{k+1}^n - V_k^n}{2}, \tag{2.14}$$

from which one can recover an approximation $f^{\Delta t}(t,v)$ to (2.1) by divided differences. As a consequence of the monotonicity of the sequence $(V_k^n)_{k\in\mathcal{K}}$ for all $n \in \mathbb{N}$, $f^{\Delta t} \geq 0$ as soon as $f_0 \geq 0$. We note in passing that the total variation of $\varrho^{\Delta t}$ in the v variable is constant by construction.

Thus it just remains to extract a subsequence still denoted $V^{\Delta t}$ converging strongly in $L^1_{loc}(\mathbb{R}_*^+ \times (0,1))$ by Helly's compactness principle. From (2.7), we can compute $\varphi(V_k^{n+1})$ for a smooth function φ; then applying the mean-value theorem leads to

$$\forall k, n, \qquad \frac{1}{\Delta t}\Big(\varphi(V_k^{n+1}) - \varphi(V_k^n)\Big) = -\sum_{k'\in\mathcal{K}} |C_{k'}| \mathcal{I}'(V_k^n - V_{k'}^n)\varphi'(V_k^n) + o(1).$$

It is now easy to deduce that $V^{\Delta t}$, (2.13) will satisfy the requirements of Definition 1 as $\Delta t, \Delta v \to 0$. Hence we deduce the following result.

THEOREM 2.7. *Let $f_0 \in \mathcal{M}_p(\mathbb{R})$ [illegible] $\mathcal{I}$ [illegible] (2.8), [illegible] $V^{\Delta t}$ [illegible] (2.13) [illegible] (2.7) [illegible] $L^1_{loc}(\mathbb{R}_*^+ \times (0,1))$ [illegible] $\Delta t, \Delta v \to 0$ [illegible] (2.1) [illegible] 1. [illegible] $\mathcal{E}$ [illegible] $\mathcal{I}(\xi) = |\xi|^{2+\gamma}/(\gamma+2)$*

3. Study of the solution's large-time behavior. From [10], we know that the so-called homogeneous cooling states cannot hope to play the role of paramount importance, as is the case for Barenblatt–Pattle similarity solutions for nonlinear diffusion equations. However, we are about to show that they still can be evidenced numerically by means of our approach.

3.1. Entropy dissipation properties and similarity variable. In this section, we must restrict ourselves to usual interaction potentials $\mathcal{I}(\xi) = |\xi|^{\gamma+2}/\gamma + 2$, $\gamma \geq 0$, because we shall use homogeneity properties. The plan is to change variables in order to make potential self-similar profiles stationary; this opens the way to the study of long-time asymptotics for (2.1). We proceed by considering the associated problem with an antidrift term (it corresponds to (1.1) with $\mu = -1$ and $\sigma = 0$):

$$\partial_t g(t,v) = \partial_v \left\{ g(t,v) \int_{\mathbb{R}} \mathcal{I}'(v-\omega) g(t,\omega).d\omega - v g(t,v) \right\}; \qquad t, v \in \mathbb{R}_*^+ \times \mathbb{R}. \tag{3.1}$$

Indeed, if $g_\infty(v)$ is a steady-state solution for (3.1), then we can deduce a similarity solution of (2.1) quite easily (see [31]):

$$f_s(t,v) = \frac{1}{\alpha(t)} g_\infty\left(\frac{v}{\alpha(t)}\right); \qquad \alpha'(t) + \alpha^{1+\gamma}(t) = 0,\ \alpha(0) = 1.$$

Let us now consider $g(\tau, v)$ solution of (3.1) for $\tau(t) = -\log(\alpha(t))$ in order to state one result from [1, 10, 24].

THEOREM 3.1. *Let $f_0 \in \mathcal{M}_p(\mathbb{R})$ [illegible] $p = 2$ [illegible] f [illegible] (2.1). [illegible] $g(\tau, v)$ [illegible] (3.1). [illegible]*

$$g \overset{\tau\to+\infty}{\longrightarrow} \frac{1}{2}\left(\delta\left(v - \frac{1}{2}\right) + \delta\left(v + \frac{1}{2}\right)\right)$$

(δ [illegible])

Notice that $\alpha(0) = 1$ implies that $\theta_s(0) = \int_{\mathbb{R}} f_s(t=0,v)v^2.dv = \int_{\mathbb{R}} v^2 g_\infty(v).dv = 1/4$. Following [1, 24], we recall the free energy functional (1.2) defined in the introduction. However, we rewrite it in terms of the reciprocal mapping,

$$\mathcal{J}(g,\tau) = \frac{1}{2+\gamma}\int_{(0,1)^2} |\tilde{V}(\tau,\varrho) - \tilde{V}(\tau,\varrho')|^{2+\gamma}.d\varrho.d\varrho' - \int_{\mathbb{R}} \tilde{V}(\tau,\varrho)^2.d\varrho,$$

where $\tilde{V}(\tau,\varrho)$ is the reciprocal mapping of $g(\tau,v)$. We can give at once the value of $\mathcal{J}(g_\infty)$:

$$\begin{aligned}\mathcal{J}(g_\infty) &= \frac{1}{2+\gamma}\int_{(0,1)^2} |\tilde{V}_\infty(\varrho) - \tilde{V}_\infty(\varrho')|^{\gamma+2}.d\varrho.d\varrho' - \frac{1}{4}\\ &= \frac{2}{2+\gamma}\int_0^{\frac{1}{2}}\int_{\frac{1}{2}}^1 |\tilde{V}_\infty(\varrho) - \tilde{V}_\infty(\varrho')|^{2+\gamma}.d\varrho.d\varrho' - \frac{1}{4}\\ &= \frac{1}{2(2+\gamma)} - \frac{1}{4} = -\frac{\gamma}{4(2+\gamma)} \le 0.\end{aligned}$$

In particular, there holds that

$$\tilde{V}(\tau,\varrho) = \frac{V(\tau,\varrho)}{\alpha(t)},$$

where V still stands for the reciprocal mapping of $f(t,v)$ solution of (2.1). Hence $\mathcal{J}(g,\tau)$ rewrites (at this level we use that $\mathcal{I}(V/\alpha) = \alpha^{-(2+\gamma)}\mathcal{I}(V)$) as

$$\mathcal{J}(f,t) = \frac{1}{2+\gamma}.\frac{1}{\alpha(t)^{(2+\gamma)}}\int_{(0,1)^2} |V(t,\varrho) - V(t,\varrho')|^{2+\gamma}.d\varrho.d\varrho' - \frac{\theta(t)}{\alpha(t)^2};$$

with the temperature $\theta(t)$ defined in section 2, $\theta(t) = \int_0^1 V(t,\varrho)^2.d\varrho$ can be shown to satisfy the following equation:

$$\begin{aligned}\frac{d\theta(t)}{dt} &= \int_0^1 \partial_t V(t,\varrho)V(t,\varrho).d\varrho\\ &= -2\int_{(0,1)^2} V(t,\varrho)\mathcal{I}'(V(t,\varrho) - V(t,\varrho')).d\varrho.d\varrho'\\ &= -\int_{(0,1)^2} \mathcal{I}(V(t,\varrho) - V(t,\varrho')).d\varrho.d\varrho' \le 0.\end{aligned}$$

In the last step we use explicitly that $\mathcal{I}(\xi)$ is a power function. It is now necessary to make an [illegible] concerning the speed of decay for $\mathcal{J}$; namely, we assume that

$$\exists z \text{ such that } 0 \le \mathcal{J}(g,\tau) - \mathcal{J}(g_\infty) \le \underbrace{[\mathcal{J}(g,\tau=0) - \mathcal{J}(g_\infty)]z(\tau)}_{Z(\tau(t))}, \tag{3.2}$$

upon which relies the following result.

PROPOSITION 3.2. [illegible] $V(t=0,\varrho)$ [illegible] $f_0(v) \in \mathcal{M}_p(\mathbb{R})$. [illegible] (3.2)

$$\lim_{t\to+\infty}\frac{1}{t}\int_0^t Z(\tau(s)).ds = 0 \Rightarrow \lim_{t\to+\infty}\frac{\theta(t)}{\alpha(t)^2} = \frac{1}{4}.$$

As an immediate consequence, this statement allows us to use the temperature as a way of rescaling the solution, as already done for nonlinear diffusion equations in [19] (except that here the task is to zoom out a Dirac measure); this might be of interest if considering more general interaction potentials $\mathcal{I}(\xi)$, in the spirit of [31], for which no similarity variable α is yet available.

. From the evolution of θ, we deduce that

$$\mathcal{J}(g,\tau) = -\frac{\alpha^{-(2+\gamma)}(t)}{\gamma(2+\gamma)}\frac{d\theta(t)}{dt} - \frac{\theta(t)}{\alpha^2(t)}.$$

Hence we compute

$$\begin{aligned}\frac{d}{dt}\big(\theta(t)\alpha^{-(2+\gamma)}(t)\big) &= \frac{d\theta(t)}{dt}\alpha^{-(2+\gamma)}(t) - (2+\gamma)\alpha^{-(3+\gamma)}(t)\alpha'(t)\theta(t)\\ &= \frac{d\theta(t)}{dt}\alpha^{-(2+\gamma)}(t) + (2+\gamma)\alpha^{-2}(t)\theta(t)\\ &= -(2+\gamma)\mathcal{J}(g,\tau).\end{aligned}$$

If (3.2) holds, then

$$-\mathcal{J}(g_\infty) \geq \frac{1}{2+\gamma}\frac{d\theta(t)}{dt}\big(\theta(t)\alpha^{-(2+\gamma)}(t)\big) \geq -\mathcal{J}(g_\infty) - Z(\tau(t)).$$

Thus we integrate in time and multiply by $\alpha^\gamma(t)$ to derive

$$-\mathcal{J}(g_\infty).t.\alpha^\gamma(t) \geq \frac{1}{2+\gamma}\frac{\theta(t)}{\alpha^2(t)} - \theta(0)\alpha^\gamma(t) \geq \alpha^\gamma(t)\left(-\mathcal{J}(g_\infty).t - \int_0^t Z(s).ds\right).$$

The final step is to notice that $\alpha^\gamma(t) = (1+\gamma t)^{-1}$ and that $-\mathcal{J}(g_\infty)(2+\gamma)/\gamma = \frac{1}{4}$. □

3.2. Numerical validation: Homogeneous cooling states. Proposition 3.2 provides us with a simple and systematic way to visualize approximate similarity solutions of (2.1) just computing

$$g_\infty^{\Delta t}(v) = f^{\Delta t}\Big(t, 2v\sqrt{\theta(t)}\Big)2\sqrt{\theta(t)}, \tag{3.3}$$

which can be easily obtained from (2.7). Indeed, Proposition 3.2 (relying on assumption (3.2)) ensures that (3.3) will behave asymptotically well. This is illustrated in Figure 1 for various values of $\gamma \geq 0$. To the authors' knowledge, this is the first time such intermediate asymptotics are displayed numerically. Clearly, as γ decreases, θ gets closer to the zero machine, and the accuracy in computing (3.3) is reduced. However, the correct shape is kept even with $\theta \simeq 10^{-10}$. The parameters we use are $\Delta v = 0.05$, and the time step is chosen adaptively. The initial data is a Gaussian distribution, $f_0(v) = \frac{1}{2\pi}\exp(-x^2/2)$.

4. Inclusion in a "thermal bath": the 1D friction-diffusion equation $\sigma > 0$. We now deal with the Cauchy problem for the equation:

$$\partial_t f(t,v) = \partial_v\left\{f(t,v)\int_{\mathbb{R}}\mathcal{I}'(v-\omega)f(t,\omega).d\omega\right\} + \sigma\partial_{vv}f; \qquad t,v \in \mathbb{R}_*^+ \times \mathbb{R}. \tag{4.1}$$

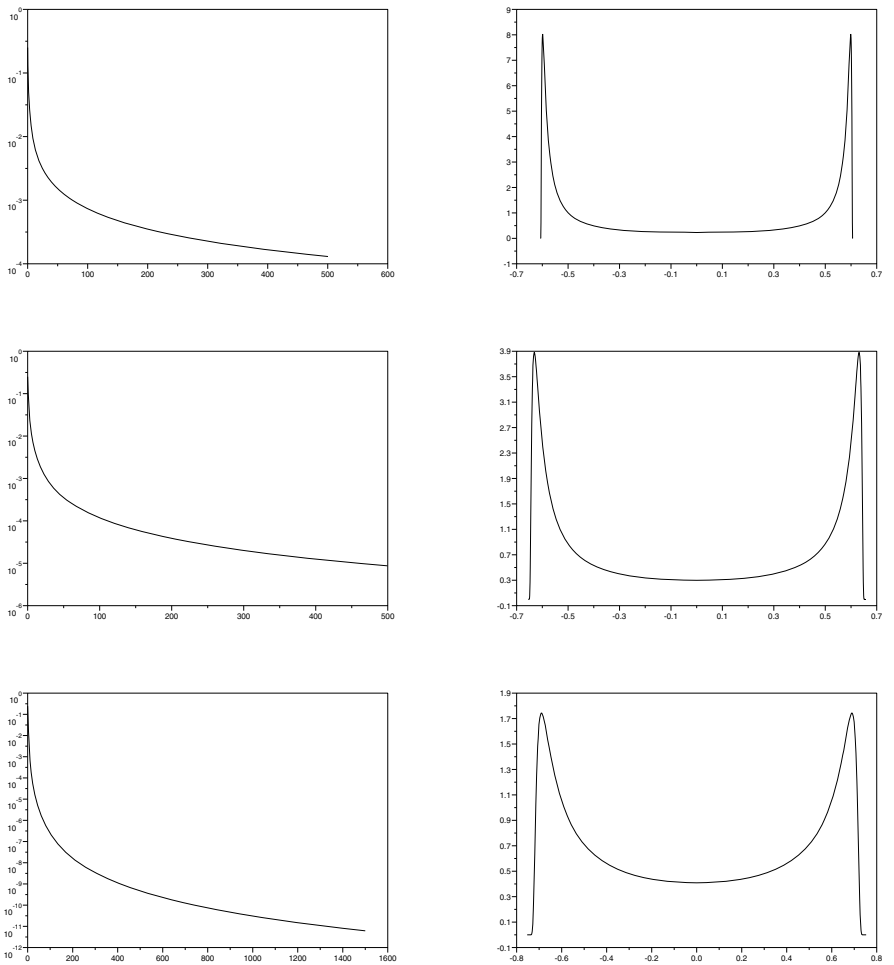


FIG. 1. *Decay of the temparatures* $t \mapsto \theta(t)$ (*left*) *and numerical similarity solutions* (*right*) *for* $\gamma = 1.8, 1.2, 0.5$ (*top to bottom*). *Notice the duration of stabilizing processes as* γ *is decreased.*

Existence and uniqueness have been shown in [12]. As in the preceding sections, we recall the weak solution to (4.1) as a distribution $f(t,v) \in C^1(\mathbb{R}_*^+; \mathcal{M}_p(\mathbb{R}))$ satisfying

$$\begin{aligned} \frac{d}{dt}\int_{\mathbb{R}} \varphi(v) f(t,v).dv &= -\int_{\mathbb{R}^2} \mathcal{I}'(v-w)\varphi'(v) f(t,v) f(t,w).dv.dw \\ &\quad + \sigma \int_{\mathbb{R}} \varphi''(v) f(t,v).dv \\ &= \frac{1}{2}\int_{\mathbb{R}^2} \mathcal{I}'(v-w)(\varphi'(w)-\varphi'(v)) f(t,v) f(t,w).dv.dw \\ &\quad + \sigma \int_{\mathbb{R}} \varphi''(v) f(t,v).dv, \end{aligned} \tag{4.2}$$

and $f(t,.) \to f(t=0,.) = f_0 \in \mathcal{M}_p(\mathbb{R})$ weakly as a measure as $t \to 0$. Clearly, from this definition, one deduces again the conservation of mass and momentum, which

corresponds to $\varphi(v) = 1, v$. However, the temperature is not meant to vanish as for the case $\sigma = 0$.

Uniqueness still holds for these measure solutions.

THEOREM 4.1. [illegible] $f_0 \in \mathcal{M}_p(\mathbb{R})$ [illegible] $p = 2$ [illegible] $\mathcal{I}$ [illegible] (4.1) [illegible]

The proof is given in Appendix B. The solution of (4.1) has been shown to converge weakly to the one of (2.1) as $\sigma \to 0$ in [26]. However, the results of [12] are more interesting at a computational level because they deal with smoother solutions. Let us recall the following result.

THEOREM 4.2. [illegible] $f_0 \in \mathcal{M}_p(\mathbb{R})$ [illegible] $\int_{\mathbb{R}}(1+v^4)f_0(v).dv$ [illegible] $\mathcal{I}$ [illegible] (4.1) [illegible] $(1+v^4)f(t,v) \in C^0(\mathbb{R}^+_*; L^1(\mathbb{R}))$ [illegible] $f_0 \in C^2(\mathbb{R})$ [illegible] f [illegible]

From this result, one sees immediately that the behavior of solutions is very different because of the diffusive term: global smooth solutions do exist.

4.1. Numerical scheme and Wasserstein contraction estimate. We do not recall the whole derivation of the equation satisfied by the reciprocal mapping (which can be easily derived from the considerations in section 2.1 and [19]): for all $t, \varrho \in \mathbb{R}^+_* \times (0,1)$,

$$\partial_t V(t,\varrho) + \sigma \partial_\varrho \left(\frac{1}{\partial_\varrho V(t,\varrho)} \right) + \int_0^1 \mathcal{I}'\Big(V(t,\varrho) - V(t,\varrho')\Big).d\varrho' = 0. \tag{4.3}$$

We shall also rely on the same type of discretization; hence we give at once the expression of the resulting scheme:

(4.4)

$$V_k^{n+1} = V_k^n - \Delta t \sum_{k' \in \mathcal{K}} |C_{k'}| \mathcal{I}'(V_k^n - V_{k'}^n) - \frac{\sigma \Delta t}{|C_k|} \left\{ \left(\frac{\varrho_{k+1} - \varrho_k}{V_{k+1}^n - V_k^n} \right) - \left(\frac{\varrho_k - \varrho_{k-1}}{V_k^n - V_{k-1}^n} \right) \right\}.$$

This scheme can be easily proven to be conservative and preserves momentum. It is also clearly asymptotic-preserving in the sense of Jin [22]. In contrast with section 2.1, (4.4) has to be completed by boundary conditions in $\varrho = 0, 1$; we observe that

$$\frac{1}{\partial_\varrho V(t,\varrho)} = f(t, V(t,\varrho)) = 0 \quad \text{for } \varrho = 0, 1,$$

at least for smooth functions vanishing at infinity. Hence in $\varrho = 0$, we have

$$V_0^{n+1} = V_0^n - \Delta t \sum_{k' \in \mathcal{K}} |C_{k'}| \mathcal{I}'(V_0^n - V_{k'}^n) - \frac{\Delta t \sigma}{|C_k|} \left(\frac{\varrho_1 - \varrho_0}{V_1^n - V_0^n} \right)$$

and a similar expression in $\varrho = 1$. In the spirit of Lemma 2.2, we can prove the following lemma.

LEMMA 4.3. [illegible] f_0, g_0 [illegible] $\mathcal{M}_p(\mathbb{R})$ [illegible] (4.1) [illegible] V, W [illegible]

$$\Delta t \sup_{V_k^n, W_{k'}^n} \left\{ \mathcal{I}'' + \sigma \left(\frac{\varrho_{k+1} - \varrho_k}{(V_{k+1}^n - V_k^n)^2} + \frac{\varrho_{k+1} - \varrho_k}{(W_{k'+1}^n - W_{k'}^n)^2} \right) \right\} \leq 1, \qquad k, k' \in \mathcal{K}^2, \tag{4.5}$$

[illegible] (4.4) [illegible] d_p [illegible]

$$\forall n \in \mathbb{N}, \qquad \sum_k |C_k| |V_k^{n+1} - W_k^{n+1}|^p \leq \sum_k |C_k| |V_k^n - W_k^n|^p, \qquad p \geq 1. \tag{4.6}$$

. We substract the schemes (4.4) and rearrange the terms corresponding to each variable. The integral term is linearized the same way and with the notation of Lemma 2.2; the diffusive part gives

$$\begin{aligned} V_k^{n+1} - W_k^{n+1} = (V_k^n - W_k^n)\Big(1 - \Delta t \sum_{k' \in \mathcal{K}} |C_{k'}| (\mathcal{I}'')_{k,k'}^n + \frac{\Delta t}{|C_k|}(\Phi'_{k+\frac{1}{2}} + \Phi'_{k-\frac{1}{2}})\Big) \\ + \Delta t \sum_{k' \in \mathcal{K}} |C_{k'}| (\mathcal{I}'')_{k,k'}^n (V_{k'}^n - W_{k'}^n) \\ - \frac{\Delta t}{|C_k|}\Phi'_{k+\frac{1}{2}}(V_{k+1}^n - W_{k+1}^n) - \frac{\Delta t}{|C_k|}\Phi'_{k-\frac{1}{2}}(V_{k-1}^n - W_{k-1}^n), \end{aligned}$$

with the following function:

$$\Phi_{k+\frac{1}{2}}(V) = \frac{\sigma(\varrho_{k+1} - \varrho_k)}{V}, \qquad \Phi'_{k+\frac{1}{2}}(V) = -\frac{\sigma(\varrho_{k+1} - \varrho_k)}{V^2} \leq 0.$$

Then $\Phi'_{k+\frac{1}{2}}$ stands for a midpoint value obtained through the mean-value theorem. From the CFL condition (4.5), the combination is convex; thus Jensen's inequality yields the announced result. □

We stress that Lemma 4.3 does not imply any L^p decay no more simply because initializing with $W_k^0 \equiv C^t \in (0,1)$ is forbidden by the CFL condition (4.5). Nevertheless, the order-preserving property still holds.

However, as said before, this result allows one to restrict attention to smooth C^2 initial data (which give rise to classical solutions as well) in order to take advantage of classical techniques in numerical analysis. Indeed, even in case the initial data is singular, one can argue that the error between the solutions emerging from its regularization and the genuine solution decay with time in any Wasserstein metric d_p.

So, going down this track, we first observe (by means of Taylor expansions) that the scheme (4.4) is clearly a first-order discretization of (4.1); hence its , is $e_k^n = O(\Delta t + |C_k|)$. Hence in order to establish convergence, we must study the evolution in time of its , $e^{\Delta t}(t,\varrho) \overset{def}{=} V^{\Delta t}(t,\varrho) - V(t,\varrho)$ for all $\varrho \in (0,1)$. It satisfies

$$\begin{aligned} \partial_t e^{\Delta t}(t,\varrho) + \sigma \partial_\varrho \left(\frac{1}{\partial_\varrho V^{\Delta t}(t,\varrho)} - \frac{1}{\partial_\varrho V(t,\varrho)} \right) \\ + \int_0^1 \mathcal{I}'\Big(V^{\Delta t}(t,\varrho) - V^{\Delta t}(t,p)\Big) - \mathcal{I}'\Big(V(t,\varrho) - V(t,p)\Big).dp = e_k^n \end{aligned}$$

for $t = t^n$ and $\varrho = \varrho_k$. At this point, one multiplies by $e^{\Delta t}(t,\varrho)$ and integrates in $\varrho \in (0,1)$. The integral term is positive by the uniqueness proof of Theorem 2.1 given in Appendix A. The diffusive term is handled thanks to a monotonicity argument, as in the proof of Theorem 4.1 given in Appendix B; indeed, an integration by parts yields

$$-\sigma \int_0^1 (\partial_\varrho V^{\Delta t} - \partial_\varrho V) \left(\frac{1}{\partial_\varrho V^{\Delta t}} - \frac{1}{\partial_\varrho V} \right)(t,\varrho).d\varrho \geq 0, \qquad \sigma \geq 0.$$

Thus it remains only that

$$\frac{1}{2}\partial_t \int_0^1 e^{\Delta t}(t,\varrho)^2.d\varrho \leq O(\Delta t + \sup_k |C_k|) \int_0^1 |e^{\Delta t}(t,\varrho)|.d\varrho,$$

and since $(0,1)$ is bounded, the L^2 norm dominates the L^1. Hence the error grows at most exponentially in time depending on the local truncation error of (4.4). We deduce the following result.

THEOREM 4.4. [illegible] $f_0 \in C^2(\mathbb{R})$ [illegible] $\mathcal{I}$ [illegible] (4.5), [illegible] $V^{\Delta t}$ [illegible] (2.13) [illegible] (4.4) [illegible] $\Delta t, \Delta v \to 0$ [illegible] [illegible] (4.3) [illegible] (4.1).

The equivalence of the two notions of solution to (4.1) has been used previously in [26]. It is easy to obtain a formal notion of consistency for the scheme (4.4) following the ideas presented in [19]; namely, one writes down the basic equality coming directly from (2.4),

$$\forall k, n \in \mathcal{K} \times \mathbb{N}, \qquad \varrho^{\Delta t}(t^n, V_k^n) = \varrho^{\Delta t}(t^{n+1}, V_k^{n+1}),$$

and then plugs into (4.4). A partly implicit finite-volume discretization of (4.1) is derived by means of Taylor expansions, assuming that $\varrho^{\Delta t}$ is at least C^1 in the v variable. We close this section mentioning that (4.4) is not the unique choice leading to the contraction property (4.6); a time-splitting strategy would behave equally well. In this last case, the proof would be the concatenation of the proof of Lemma 2.2 and [19] (the CFL would be slightly lighter though). The last situation emanating from (2.1) which has not yet been treated is $\sigma > 0$ and $\mu \neq 0$ and corresponds to a so-called [illegible] equation, as studied in [12]. However, when expressed by means of the reciprocal mapping, the confining term rewrites simply as $\mu V(t, \varrho)$, thus leading to a simple exponential term which can be integrated explicitly. Indeed, upon multiplying (1.5) by $\exp(\mu t)$, one derives the following equation:

$$\partial_t \tilde{V}(t,\varrho) + \exp(\mu t)\left\{\sigma \partial_\varrho\left(\frac{1}{\exp(-\mu t)\partial_\varrho \tilde{V}(t,\varrho)}\right) + \int_0^1 \mathcal{I}'\Big(\exp(-\mu t)(\tilde{V}(t,\varrho) - \tilde{V}(t,\varrho'))\Big).d\varrho'\right\} = 0, \tag{4.7}$$

where $\tilde{V}(t,.) = \exp(\mu t)V(t,.)$. The aforementioned techniques allow us to cover this case with minor changes.

4.2. Numerical results; study of the solution's "tails". Our first validation for (4.4) has been to look at the behavior of rescaled numerical solutions (3.3) as σ is decreased by several orders of magnitude. This has to do with checking the so-called asymptotic-preserving property in the sense of [22]. In this case, the decay of the approximate solution's temperature is interesting to look at since it should strengthen as $\sigma \to 0$. This is indeed the case as seen in Figure 2; the parameters used are $\Delta v = 0.27$ and $\gamma = 1.2$. The time step Δt is chosen adaptively according to (4.5).

It is a well-known fact that for great values of $|v|$, the solution of (4.1) with $\gamma = 1$ behaves like $\exp(-|v|^3)$; see [7, 8, 9]. We tried to check it out by iterating (4.4) with $\sigma = 1$ for quite a long time starting from a Gaussian initial datum. Results are shown in Figure 3. However, as the values of $f^{\Delta t}$ become very small (less than 10^{-50}), results could be slightly corrupted by spurious noise; it is nevertheless seeable that the behavior is qualitatively correct.

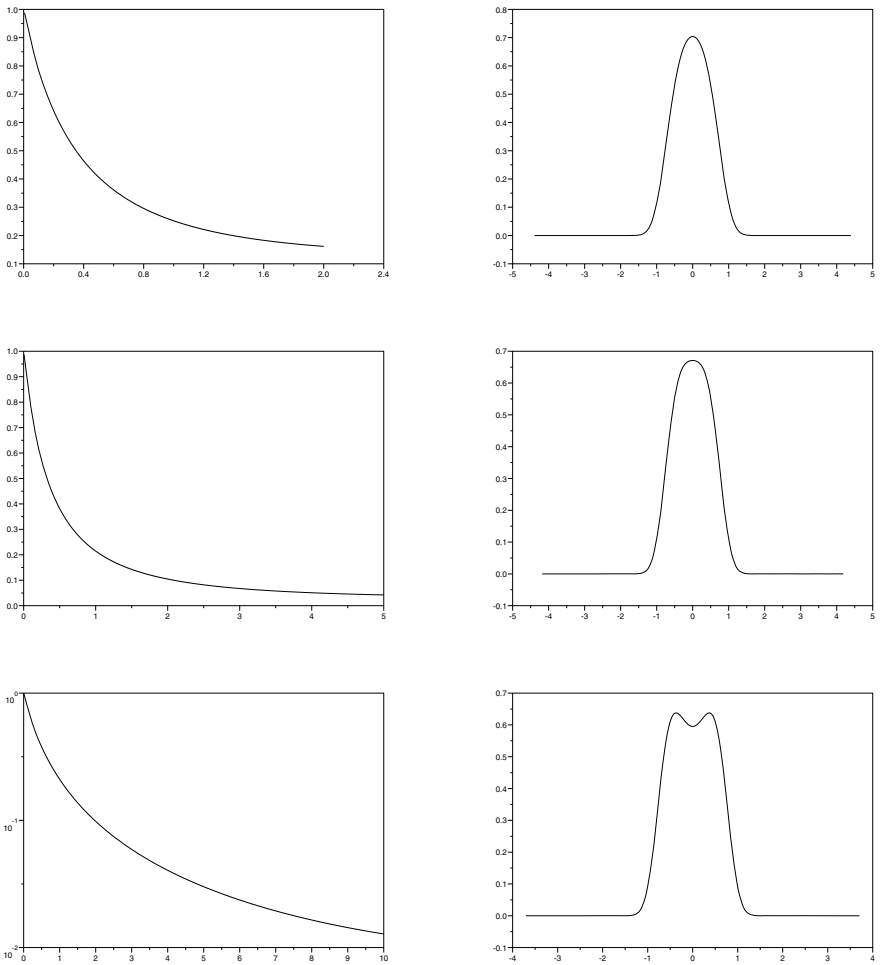


FIG. 2. *Decay of the temparatures* $t \mapsto \theta(t)$ (*left*) *and rescaled numerical solutions* (3.3) (*right*) *for* $\sigma = 0.1, 0.01, 0.001$ (*top to bottom*).

5. Numerical approximations of the Hele–Shaw cell.

5.1. Derivation of the numerical process. As announced in the introduction, we are about to concentrate mainly on a particular case of (1.4) corresponding to $a(u) = u$ (we drop the superscripts ε, as there is no ambiguity),

$$\partial_t u + \partial_x\big(u\partial_x(u *_x \Gamma_\varepsilon''(x))\big) = 0, \qquad 0 \le u_0 \in L^1 \cap H^1(\mathbb{R}).$$

The general case with mobility $a(u) \neq u$ is to be tackled in section 6 by means of quite different techniques. It is well known that (1.3) is mass-preserving; hence there is no loss of generality in considering only nonnegative initial data satisfying the additional requirement $\int_{\mathbb{R}} u_0(x).dx = 1$. One easily shows that it also dissipates energy, i.e.,

$$\frac{d}{dt}\int_{\mathbb{R}} |\partial_x u(t,x)|^2.dx \le 0, \qquad t \in \mathbb{R}^+.$$

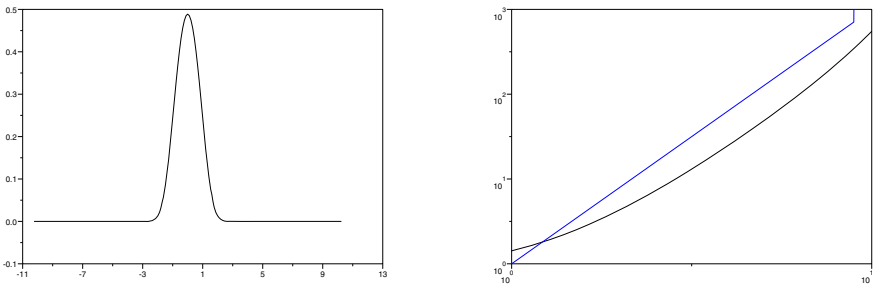


FIG. 3. *Large-time behavior for* (4.1) *at time* $t \simeq 33$ (*left*) *and comparison of its solution's tails with* $v \mapsto \exp(-|v|^3)$ (*right, where we plotted* $-\ln(f^{\Delta t})$ *and* $|v^3|$ *in logscale*).

Nonnegativity preservation has been proved to hold as a consequence of entropy dissipation [4, 5, 6]; however, in one space dimension, positive solutions to (1.3) are classical for $a(u) = u^p$, p big enough. As the solution vanishes, it typically suffers a loss of regularity and propagates at finite speed (it admits in particular source-type solutions [16]). This is strongly reminiscent of second-order degenerate parabolic equations already treated numerically in [19]. In this context, we first recall that approximations of the type (1.4) have been shown to be consistent with the original problem (1.3) for the special choice of Γ_ε being a Gaussian kernel,

$$\Gamma_\varepsilon(x) = \frac{1}{\sqrt{2\pi\varepsilon}} \exp\left(-\frac{x^2}{2\varepsilon}\right);$$

see Theorem 3.1 in [29]. Relying on this result, we shall pursue with this choice even if (many) other choices could be admissible too, as, for instance, the compactly supported Landau kernel given by ($m \in \mathbb{N}$)

$$\Gamma_{\varepsilon,m}(x) = \begin{cases} \frac{(\varepsilon^2 - x^2)^m}{\int_{-\varepsilon}^{\varepsilon} (\varepsilon^2 - y^2)^m . dy}, & |x| \leq \varepsilon, \\ 0, & |x| > \varepsilon. \end{cases}$$

For our purpose, one would get

$$\Gamma_{\varepsilon,4}(x) = \frac{315(\varepsilon^2 - x^2)^4}{128\varepsilon^9}, \qquad \Gamma'''_{\varepsilon,4}(x) = \frac{105}{2\varepsilon^6} x(9\varepsilon^2 - 21x^2)\Gamma_{\varepsilon,1}(x).$$

Other kernels can be found in, e.g., [33]. Anyway, following ideas from the preceding sections, one can derive a numerical scheme for (1.4): by integration,

$$\partial_t \varrho + (\partial_x \varrho)(\partial_x \varrho *_x \Gamma'''_\varepsilon) = 0, \qquad \varrho(t,x) = \int_{-\infty}^{x} u(t,y).dy,$$

and introducing $X(t,\varrho)$ as the reciprocal mapping of $u(t,x)$, ($\|u_0\|_{L^1(\mathbb{R})} = 1$)

$$\partial_t X(t,\varrho) - \int_0^1 \Gamma'''_\varepsilon(X(t,\varrho) - X(t,\varrho')).d\varrho' = 0, \qquad (t,\varrho) \in \mathbb{R}^+ \times (0,1). \tag{5.1}$$

For Γ_ε being Gaussian, the scheme is completely determined with

$$\Gamma_\varepsilon(x) = \frac{1}{\sqrt{2\pi\varepsilon}} \exp\left(-\frac{x^2}{2\varepsilon}\right), \qquad \Gamma'''_\varepsilon(x) = \left(\frac{3x}{\varepsilon^2} - \frac{x^3}{\varepsilon^3}\right)\Gamma_\varepsilon(x).$$

Hence, defining a Cartesian grid by means of two positive parameters $\Delta t, \Delta x$ and denoting $t^n = n\Delta t$, $x_k = k\Delta x$, $\varrho_k = \varrho_0(x_k)$, one obtains finally

$$X_k^{n+1} = X_k^n + \Delta t \sum_{k' \in \mathcal{K}} |C_{k'}| \Gamma_\varepsilon'''(X_k^n - X_{k'}^n). \tag{5.2}$$

Now the novelty lies in the fact that Γ_ε''' has no sign; this somehow reflects the high-order space derivative in (1.3). As in section 2, no boundary conditions are needed for (5.2).

5.2. Stability estimates for $\varepsilon > 0$. As in former papers [34, 21], we aim at establishing a nonnegativity-preserving property; however, our scheme is not contractive in any Wasserstein distance. What remains is only the monotonicity-preserving property.

PROPOSITION 5.1. [illegible] $u_0 \geq 0$ [illegible] $X(t, \varrho)$ [illegible] (5.1) [illegible] $X(t = 0, .)$ [illegible] $\varrho(t = 0, .)$ [illegible]

$$\Delta t \leq \frac{1}{6} \varepsilon^2 \sqrt{2\pi\varepsilon \exp(3)}, \tag{5.3}$$

[illegible] (5.2) [illegible]

[illegible]. The proof proceeds by substracting and linearizing (5.2); one gets

$$\begin{aligned} X_k^{n+1} - X_{k-1}^{n+1} &= X_k^n - X_{k-1}^n + \Delta t \sum_{k' \in \mathcal{K}} |C_{k'}| \Big(\Gamma_\varepsilon'''(X_k^n - X_{k'}^n) - \Gamma_\varepsilon'''(X_{k-1}^n - X_{k'}^n) \Big) \\ &= (X_k^n - X_{k-1}^n) \Big(1 + \Delta t \sum_{k' \in \mathcal{K}} |C_{k'}| (\Gamma_\varepsilon'''')_{k,k-1,k'}^n \Big), \end{aligned}$$

where $(\Gamma_\varepsilon'''')_{k,k-1,k'}^n = \Gamma_\varepsilon''''(\lambda X_k^n + (1-\lambda) X_{k-1}^n - X_{k'}^n)$. Now, for Γ_ε being Gaussian, its fourth derivative reads as

$$\Gamma_\varepsilon''''(x) = \frac{\Gamma_\varepsilon(x)}{\varepsilon^2} \big(y^2 - 6y + 3 \big), \qquad y = \frac{x^2}{\varepsilon}.$$

By elementary calculus, this polynomial has a minimum for $y = 3$, and the corresponding value is -6. It remains to observe that (5.3) ensures that $1 + \Delta t \inf \Gamma_\varepsilon'''' \geq 0$. □

As an immediate consequence, nonnegativity for the u variable holds under the same condition (5.3). At this point, we stress that our scheme (5.2) is endowed with a very light CFL restriction $\Delta t = O(\varepsilon^{5/2})$; indeed, the parameters Δt and Δx are completely decorrelated as in section 2.2. Clearly, the smaller the ε, the more precise the numerical approximation but the higher the computational cost. The choice of the Landau kernel would impose $\Delta t = O(\varepsilon^5)$.

We now move on to an energy estimate for the scheme (5.2); it is convenient to recall some shorthand notation from [19], namely,

$$\delta X_{k-\frac{1}{2}}^n := X_k^n - X_{k-1}^n, \qquad \delta \varrho_{k-\frac{1}{2}} := \varrho_k - \varrho_{k-1}.$$

PROPOSITION 5.2. [illegible] $\alpha = \frac{6}{\varepsilon^2 \sqrt{2\pi\varepsilon \exp(3)}}$ [illegible] 5.1, [illegible] (5.2):

$$\forall n \in \mathbb{N}, \quad \left| \frac{\delta \varrho_{k+\frac{1}{2}}}{\delta X_{k+\frac{1}{2}}^n} - \frac{\delta \varrho_{k-\frac{1}{2}}}{\delta X_{k-\frac{1}{2}}^n} \right| \leq \exp(\alpha t^n) \left\{ \left| \frac{\delta \varrho_{k+\frac{1}{2}}}{\delta X_{k+\frac{1}{2}}^0} - \frac{\delta \varrho_{k-\frac{1}{2}}}{\delta X_{k-\frac{1}{2}}^0} \right| + O(\varepsilon^2) \right\}. \tag{5.4}$$

. We observe that from the proof of Proposition 5.1, the divided difference satisfies

$$\left|\frac{\delta\varrho_{k-\frac12}}{\delta X^n_{k-\frac12}}\right| \le \left|\frac{\delta\varrho_{k-\frac12}}{\delta X^{n-1}_{k-\frac12}}\right| \frac{1}{1+\Delta t\inf\Gamma''''_\varepsilon} \le \left|\frac{\delta\varrho_{k-\frac12}}{\delta X^0_{k-\frac12}}\right| \exp(-t^n\inf(\Gamma''''_\varepsilon)),$$

since the CFL condition (5.3) ensures monotonicity preservation and $\inf(\Gamma''''_\varepsilon)\le 0$. For any choice of a,b,c,d in $\mathbb{R}^+$, there holds that $\left|\frac{a}{b}-\frac{c}{d}\right| \le \frac{|a-c|}{b} + c\left|\frac1b-\frac1d\right|$. Hence we get that

$$\left|\frac{\delta\varrho_{k+\frac12}}{\delta X^{n+1}_{k+\frac12}} - \frac{\delta\varrho_{k-\frac12}}{\delta X^{n+1}_{k-\frac12}}\right| \le \frac{\left|\frac{\delta\varrho_{k+\frac12}}{\delta X^{n}_{k+\frac12}} - \frac{\delta\varrho_{k-\frac12}}{\delta X^{n}_{k-\frac12}}\right|}{1+\Delta t\inf(\Gamma''''_\varepsilon)} + O(\Delta t)\left|\frac{\delta\varrho_{k-\frac12}}{\delta X^0_{k-\frac12}}\right| \exp(-t^n\inf(\Gamma''''_\varepsilon)).$$

By induction, this yields

$$\begin{aligned}\left|\frac{\delta\varrho_{k+\frac12}}{\delta X^{n+1}_{k+\frac12}} - \frac{\delta\varrho_{k-\frac12}}{\delta X^{n+1}_{k-\frac12}}\right| &\le \exp(-(n+1)\Delta t\inf(\Gamma''''_\varepsilon))\\ &\quad\times\left\{\left|\frac{\delta\varrho_{k+\frac12}}{\delta X^{0}_{k+\frac12}} - \frac{\delta\varrho_{k-\frac12}}{\delta X^{0}_{k-\frac12}}\right| + O(\Delta t)\left|\frac{\delta\varrho_{k-\frac12}}{\delta X^0_{k-\frac12}}\right| \frac{1+\Delta t\inf(\Gamma''''_\varepsilon)}{-\Delta t\inf(\Gamma''''_\varepsilon)}\right\}.\end{aligned}$$

It remains to plug into the value of $\inf(\Gamma''''_\varepsilon)\le 0$ and to observe that the last term is at least $O(\varepsilon^2)$. Thus (5.4) follows. □

Unfortunately, it does not seem easy to establish the equivalent of the energy decay which holds for the continuous problem (1.3) as a consequence of the nonnegativity of $a(u)$. However, (5.4) can be considered as a "cheap" estimate which ensures that approximate solutions generated by (5.2) will not blow up in finite time.

5.3. Numerical results: Dead core phenomenon and similarity solution. We finally aim at illustrating our approach's outcome on two test cases from the literature, each describing a typical phenomenon for (1.3): the first one is the appearance of the so-called dead core, which corresponds to film rupture (see section 9 in [34] and [21]); the second is the spreading of the source-type solution already studied in [16, 29].

We first consider the following initial data, taken from [6, 34]:

$$u_0(x) = 0.8 - \cos(\pi x) + 0.25\cos(2\pi x), \qquad x\in(-2,2). \tag{5.5}$$

Despite the fact that the authors of [34] consider $a(u)=\sqrt{u}$, we obtained dead core evidence around time $t\simeq 0.003$ as shown in Figure 4; initial data are plotted for comparison. The parameters used for this run are $\Delta x = 2^{-7}$ and $\varepsilon = 0.002$, and Δt is computed from (5.3). Observe in particular the loss of regularity when passing from $t=0.002$ to $t=0.0035$. There is no clear touchdown onto the x-axis, as our scheme is endowed with an error in $\varepsilon>0$ from the very beginning; left apart are the errors coming from the discretization in $\Delta t, \Delta x$.

The second case of interest is the simulation of a droplet's spreading, rendered through a similarity solution which is only $C^1(\mathbb{R})$; the following formula is (partially) taken from [29]:

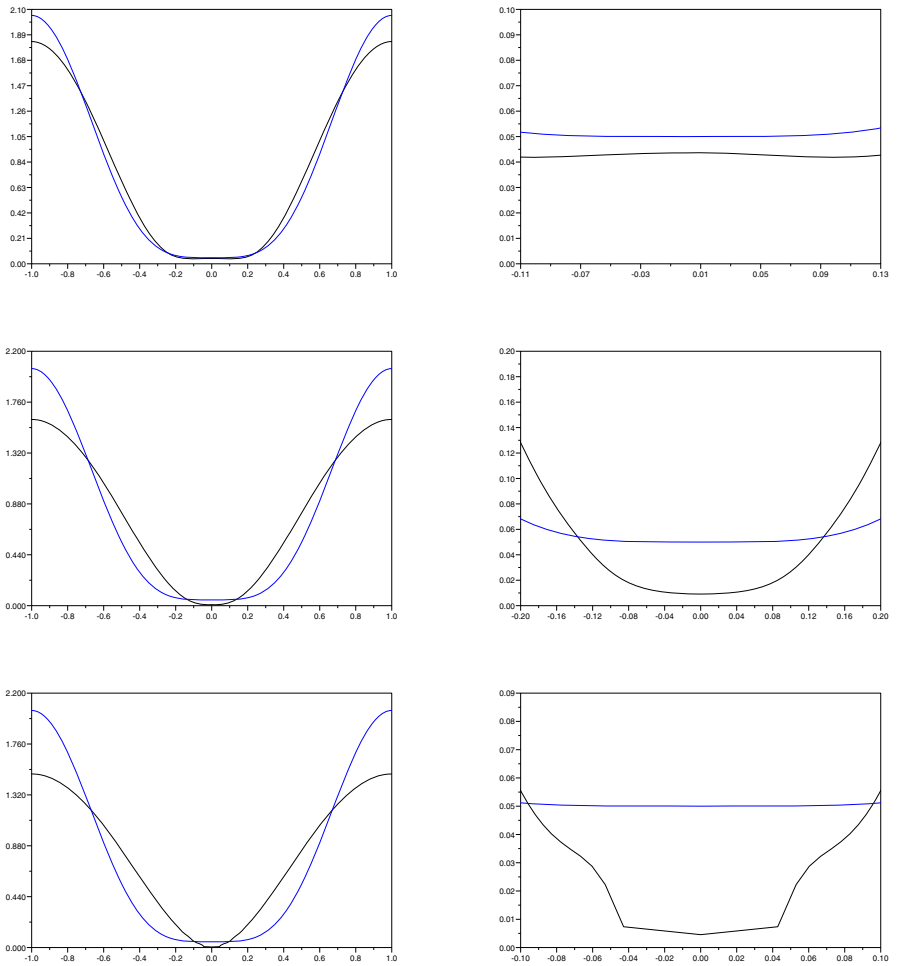


FIG. 4. *Evolution of the dead core for initial data* (5.5) *at times* $t = 0.001, 0.002, 0.0035$ (*top to bottom*).

(5.6)

$$S(t,x) = \frac{1}{24(5t+\tau)^{\frac{1}{5}}} \max\left[0, \left(\omega^2 - \frac{x^2}{(5t+\tau)^{\frac{2}{5}}}\right)^2\right], \qquad u_0(x) = \frac{S(0,x)}{\|S(0,.)\|_{L^1(\mathbb{R})}}.$$

The parameters used here are $\Delta x = 2^{-8}$, $\varepsilon = 0.03$, $\omega = 2$, and $\tau = 0.5$, and we iterate up to time $t = 0.11$; see Figure 5. The main difference with the preceding case is that the rendering of our scheme (5.2) shows oscillations (which decay with ε): this comes from the loss of regularity at the vanishing points of the approximate solution. We note in passing that oscillations occurred also for [29] and [34], where a postprocessing has been implemented to plot only selected points (see [34, p. 550]). From our CFL condition (5.3), there is clearly a tradeoff between quality of rendering and CPU time for such a simulation. We plotted the initial data and the exact value of the source-type profile for the sake of completeness; an evolution of the temperature is also shown.

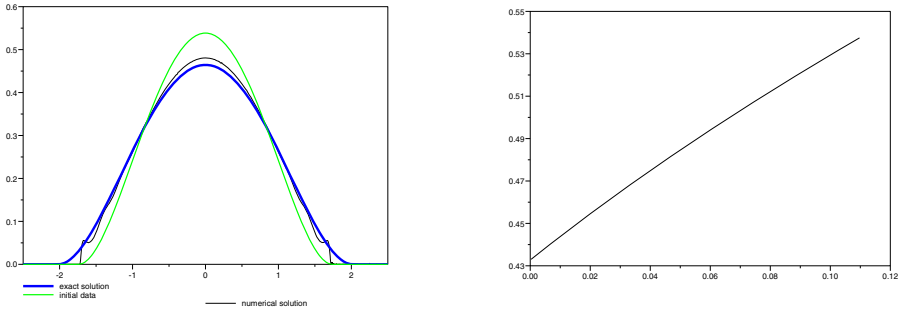


FIG. 5. *Spreading of the similarity solution* (5.6) *at time* $t = 0.11$ *with* $\tau = 0.5$ (*left*) *and evolution of its temperature* $t \mapsto \theta(t)$ (*right*).

. . . . 2. From the computations in section 5.1, one could formally extend the present approach to mobilities $a(u) \neq u$. In this direction, (5.1) would modify like

$$\partial_t X(t,\varrho) - a\left(\frac{1}{\partial_\varrho X}\right)\partial_\varrho X(t,\varrho)\int_0^1 \Gamma_\varepsilon'''(X(t,\varrho)-X(t,\varrho')).d\varrho' = 0$$

and the numerical scheme (5.2), with obvious notation, like

$$X_k^{n+1} = X_k^n + \tilde{a}\left(\frac{\delta\varrho_{k+\frac{1}{2}}}{\delta X^n_{k+\frac{1}{2}}}, \frac{\delta\varrho_{k-\frac{1}{2}}}{\delta X^n_{k-\frac{1}{2}}}\right)\Delta t \sum_{k'\in\mathcal{K}} |C_{k'}|\Gamma_\varepsilon'''(X_k^n - X_{k'}^n).$$

Hence the design of a discrete mobility $\tilde{a} \geq 0$ which could maintain monotonicity and energy estimates has to be tackled (as in [34]).

6. An efficient Fourier scheme for general thin-film equations. As pointed out in Remark 2, the preceding approach cannot be extended to general mobilities $a(u) = u^p$ in a straightforward manner. To partially fix this drawback, we wish to propose here a general way to build schemes for (1.3), based on repeated use of `fft` routines. However, we have been unable to establish any rigorous bound on this class of approximate solutions; thus we just mention it here. Future work might be dedicated to these matters.

We assume solutions of (1.3) endowed with a $H^1(\mathbb{R})$ regularity in the space variable (see [16]); hence we introduce their Fourier transform which we shall denote $\hat{u}(t,\xi)$ for $\xi \in \mathbb{R}$. Obviously, there holds that $\partial_t\hat{u}(t,\xi) = -i\xi\widehat{a(u)\partial_{xxx}u}$. Our strategy is based on the idea of computing correctly the nonlinear term relying on the unknown's smoothness. By repeated integrations by parts, one derives the following equality:

$$\widehat{a(u)\partial_{xxx}u}(t,\xi) = -i\xi\int_{\mathbb{R}}\left(A(\partial_x u) - 2\partial_x u\partial_x a(u) - \xi^2 A(u)\right)\exp(-ix\xi).dx,$$

where, for $a(u) = u^p$, $A(u) = \frac{u^{p+1}}{p+1}$. This leads to the general expression:

(6.1)
$$\partial_t\hat{u}(t,\xi) + \xi^2\left\{B\widehat{(u,\partial_x u)}(t,\xi) + \xi^2\widehat{A(u)}(t,\xi)\right\} = 0, \quad B(u,\partial_x u) = p(\partial_x u)^2u^{p-1} + A(\partial_x u).$$

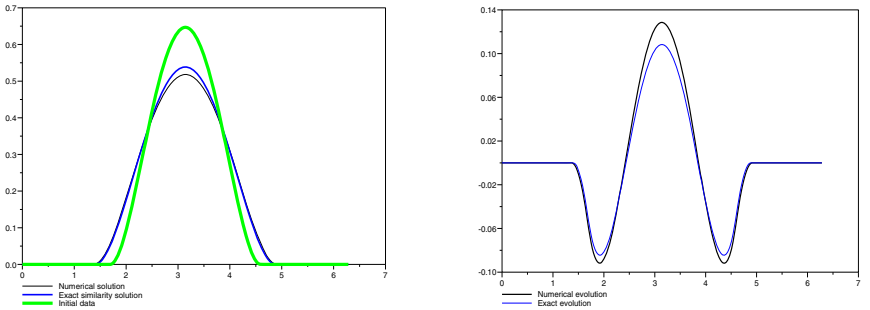


FIG. 6. *Similarity solution* (5.6) *at time* $t = 0.06$ *with* (6.2), $\tau = 0.2$ (*left*); *difference with initial data for exact solution and numerical* (*right*).

The Hele–Shaw cell corresponding to $p = 1$ is actually a quite particular case, as (6.1) simplifies and boils down to[1]

$$\partial_t \hat{u}(t,\xi) + \xi^2 \left\{ \frac{3}{2} \widehat{|\partial_x u|^2}(t,\xi) + \frac{\xi^2}{2} \widehat{|u|^2}(t,\xi) \right\} = 0. \tag{6.2}$$

It is easy to implement both (6.1) and (6.2) using, e.g., the SCILAB package. The derivative $|\partial_x u|^2$ is computed via Fourier transform, too; this has the advantage of suppressing the wide-stencil problem inherent to (1.3). Indeed, both approaches in [21, 34] suffer from it. Moreover, we observed that (6.2) does not ask for an implicit time integrator; like all the schemes proposed in this paper, it can be treated explicitly, thus keeping the computational cost very low. A last observation is that [illegible] that the CFL condition for (6.2) is linear, similar to $\Delta t \leq \Delta x/20^3$ instead of $\Delta t = O(\Delta x^4)$ for finite differences; the factor 20 has been found experimentally, and we cannot give any rigorous explanation of it. It is necessary to filter spurious oscillations when computing derivatives; a standard Gaussian filter has been used there.

As a numerical validation, we selected again the similarity solution (5.6) with the same parameters, except a lower value of τ, and 256 discretization points in space (this asks for $\Delta t = 2.10^{-6}$). The results are displayed in Figure 6. One observes that despite a lower space resolution, the approximate solution is better than the one in Figure 5, as it does not show any oscillations at its tails. The speed of propagation is also correct. There is a particular structure associated with the differential equation (6.2); we present some computations on this in Appendix C. For the sake of completeness, we also include numerical results on a test case proposed in [21]: it is a Lipschitz initial datum, $u_0(x) = 0.1 \max(0, \pi - 2|\pi - x|)$, for $x \in [0, 2\pi]$. This highlights the great importance of the exponent p in the dynamics of (1.3) since, for $p = 1$, it develops instantaneously a zero-angle contact on each side, whereas for $p = 4$, the support does not spread [illegible], and convergence onto a parabolic profile is observed. Both features are successfully rendered by our Fourier approach (6.1), as can be seen in Figure 7. Similar parameters have been used, and the scheme is fully explicit in time.

7. Conclusion and outlook. We have proposed in this paper to work out a rather wide class of 1D nonlinear diffusion equations arising from physics modeling by using numerical techniques borrowed from mass transportation theory [32]. In cases it applies to, this approach generally allows for the derivation of simple schemes and

[1] This has to do with Theorem 1, Case 2 in [6].

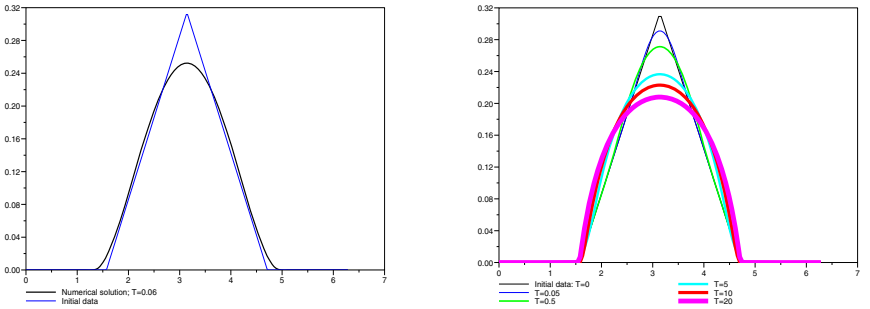


FIG. 7. *Evolution of Lipschitz initial data for $a(u) = |u|$ (left) and $a(u) = |u|^4$ (right).*

short proofs. Similar techniques could be applied in other areas of application, e.g., the so-called flashing ratchet [23]. However, several questions remain open; among them lies the design of an efficient strategy for the simulation of the full dissipative Boltzmann equation in the space t, x, v (see, however, [26]). The situation for the general thin-film equation is even more challenging; especially, the case $a(u) = u^3$ is very much of interest from the physics perspective (see again [25]). For this case, $p = 3$ and (6.1) reads as

$$\partial_t \hat{u}(t,\xi) + \xi^2 \left\{ \frac{3}{2} \widehat{|\partial_x u^2|^2}(t,\xi) + \frac{1}{4} \widehat{|\partial_x u|^4}(t,\xi) + \frac{\xi^2}{4} \widehat{|u|^4}(t,\xi) \right\} = 0.$$

Appendix A. Proof of Theorem 2.1. We consider two possible solutions f, g and assume that as densities they satisfy $\int_{\mathbb{R}} f(t,v).dv \equiv 1$ and $\int_{\mathbb{R}} vf(t,v).dv \equiv 0$. Then we insert (2.5) in the following expression to derive

$$\begin{aligned} \frac{d}{dt}\int_0^1 |V(t,\varrho)-W(t,\varrho)|^2.d\varrho &= 2\int_0^1 \partial_t (V(t,\varrho)-W(t,\varrho))(V(t,\varrho)-W(t,\varrho)).d\varrho \\ &= -2\int_{(0,1)^2} \big(\mathcal{I}'(V(t,\varrho)-V(t,p))-\mathcal{I}'(W(t,\varrho)-W(t,p))\big) \\ &\quad \times (V(t,\varrho)-W(t,\varrho)).dp.d\varrho. \end{aligned}$$

At this point, it is useful to recall from [24] that $\int_{\mathbb{R}} (v-w) d\mu(v,w) = 0$ for all density μ admitting f and g as marginals. This implies that $\int_0^1 V(t,\varrho) - W(t,\varrho).d\varrho = 0$; hence,

$$V(t,\varrho) - W(t,\varrho) = \int_0^1 (V(t,\varrho) - V(t,q)) - (W(t,\varrho) - W(t,q)).dq.$$

This is to be plugged into the previous equality in order to derive

$$\begin{aligned} &\frac{d}{dt}\int_0^1 |V(t,\varrho) - W(t,\varrho)|^2.d\varrho \\ &\quad = -2\int_0^1 \left(\int_0^1 \mathcal{I}'(V(t,\varrho) - V(t,p)) - \mathcal{I}'(W(t,\varrho) - W(t,p)).dp \right) \\ &\qquad \times \left(\int_0^1 (V(t,\varrho) - V(t,q)) - (W(t,\varrho) - W(t,q)).dq \right).d\varrho. \end{aligned}$$

Now the trick is to observe that the roles of p and ϱ are symmetric; thus we can exchange them in the second integral part. The integral rewrites as

$$\frac{d}{dt}\int_0^1 |V(t,\varrho)-W(t,\varrho)|^2.d\varrho = 2\int_{(0,1)^3} \big(\mathcal{I}'(V(t,p)-V(t,\varrho))-\mathcal{I}'(W(t,p)-W(t,\varrho))\big) \times\big((V(t,p)-V(t,q))-(W(t,p)-W(t,q))\big).d\varrho.dp.dq.$$

The sign changes because $\mathcal{I}'$ is odd. At this point, one computes half the difference of the preceding expressions in order to make the q terms disappear. All this boils down to

$$\frac{d}{dt}\int_0^1 |V(t,\varrho)-W(t,\varrho)|^2.d\varrho = -\int_{(0,1)^2} \big((V(t,\varrho)-V(t,p))-(W(t,\varrho)-W(t,p))\big) \times\big(\mathcal{I}'(V(t,\varrho)-V(t,p))-\mathcal{I}'(W(t,\varrho)-W(t,p))\big).dp.d\varrho \le 0,$$

since $\mathcal{I}'$ is nondecreasing. Finally, we reach the contraction estimate which has been first proved in [24] for special potentials.

Appendix B. Proof of Theorem 4.1. We follow the same idea while aiming at computing $\frac{d}{dt}\int_0^1 |V(t,\varrho)-W(t,\varrho)|^2.d\varrho$ as in Appendix A. Two terms appear on the right-hand side; the first one corresponds to the friction part and is to be treated exactly as in Appendix A. For the second one, it is enough to observe that

$$\begin{aligned} &-\int_0^1 (V(t,\varrho)-W(t,\varrho))\partial_\varrho\left(\frac{1}{\partial_\varrho V(t,\varrho}-\frac{1}{\partial_\varrho W(t,\varrho)}\right).d\varrho \\ &=\int_0^1 (\partial_\varrho V(t,\varrho)-\partial_\varrho W(t,\varrho))\left(\frac{1}{\partial_\varrho V(t,\varrho)}-\frac{1}{\partial_\varrho W(t,\varrho)}\right).d\varrho \le 0 \end{aligned}$$

by monotonicity of the function $v\mapsto 1/v$. The boundary terms are zero thanks to

$$\frac{1}{\partial_\varrho V(t,\varrho)} = f(t,V(t,\varrho)) = 0 \quad \text{for } \varrho = 0,1.$$

Appendix C. The structure of (6.2). It is interesting to rewrite in full detail (6.2):

$$\partial_t\hat{u}(t,\xi) = -\frac{\xi^2}{8\pi^2}\int_{\mathbb{R}} \exp(-ix\xi)\Big\{3\Big(\int_{\mathbb{R}} i\eta\hat{u}(t,\eta)\exp(ix\eta).d\eta\Big)^2 + \xi^2\Big(\int_{\mathbb{R}}\hat{u}(t,\eta)\exp(ix\eta).d\eta\Big)^2\Big\}.dx.$$

It trivially yields the conservation of mass since $\partial_t\hat{u}(t,0)=0$. The quantities between parentheses belong to $\mathbb{R}$. We use the fact that $a^2+b^2=(a+ib)(a-ib)$ and derive, after a simple change of variables,

$$\partial_t\hat{u}(t,\xi) = -\frac{3\xi^2}{8\pi^2}\int_{\mathbb{R}} \exp(-ix\xi)\Big(\int_{\mathbb{R}} i\eta\hat{u}\left(t,\eta-\frac{\xi}{\sqrt{3}}\right)\exp(ix\eta).d\eta\Big) \times\Big(\int_{\mathbb{R}} i\eta\hat{u}\left(t,\eta+\frac{\xi}{\sqrt{3}}\right)\exp(ix\eta).d\eta\Big).dx.$$

We can easily see that the terms between parentheses are now each other's complex conjugate since they read, e.g., as

$$M(x,\xi) \stackrel{def}{=} \int_{\mathbb{R}} i\eta\hat{u}\left(t,\eta+\frac{\xi}{\sqrt{3}}\right)\exp(ix\eta).d\eta$$
$$= \int_{\mathbb{R}^+} i\eta\left[\hat{u}\left(t,\eta+\frac{\xi}{\sqrt{3}}\right)\exp(ix\eta) - \overline{\hat{u}\left(t,\eta-\frac{\xi}{\sqrt{3}}\right)\exp(ix\eta)}\right].d\eta,$$

where we have used that u being real, $\hat{u}(t,-\xi) = \overline{\hat{u}(t,\xi)}$ for all ξ. Moreover, the quantity appearing in the expression of $\partial_t\hat{u}(t,\xi)$, $\xi \mapsto M(x,\xi)\overline{M(x,\xi)}$, is an even function for any value of x. Using this while performing a backward Fourier transform leads us to a simple expression for $\partial_t u(t,x)$,

$$\partial_t u(t,x) + \frac{1}{32\pi^3}\int_{\mathbb{R}^2}\xi^2\cos((y-x)\xi)M(x,\xi)\overline{M(x,\xi)}.dy.d\xi = 0,$$

which turns out to belong to $\mathbb{R}$. Hence the whole computation is coherent. This highlights a convolution structure inside the Hele–Shaw equation since one could rewrite it under the form

$$\partial_t u(t,x) + \frac{1}{32\pi^3}\int_{\mathbb{R}^2}\xi^2\big(M(.,\xi)\overline{M(.,\xi)} * \cos(\xi x)\big).d\xi = 0.$$

Acknowledgments. Part of this research was conducted in the Mathematics Department of the Università di Pavia, whose hospitality is acknowledged. The authors wish to thank Prof. J. A. Carrillo for having drawn their attention to the present problem.

REFERENCES

[1] D. Benedetto, E. Cagliotti, and M. Pulvirenti, *A kinetic model for granular media*, RAIRO Modél. Math. Anal. Numer., 31 (1997), pp. 615–641.
[2] D. Benedetto, E. Cagliotti, J. A. Carrillo, and M. Pulvirenti, *A non-Maxwellian distribution for one-dimensional granular media*, J. Statist. Phys., 91 (1998), pp. 979–990.
[3] E. Beretta, M. Bertsch, and R. Dal Passo, *Nonnegative solutions of a fourth-order nonlinear degenerate parabolic equation*, Arch. Rational Mech. Anal., 129 (1995), pp. 175–200.
[4] F. Bernis, *Finite speed of propagation and continuity of the interface for slow viscous flows*, Adv. Differential Equations, 1 (1996), pp. 337–368.
[5] F. Bernis and A. Friedman, *Higher order nonlinear degenerate parabolic equations*, J. Differential Equations, 83 (1990), pp. 179–206.
[6] A. Bertozzi and M. Pugh, *The lubrication approximation for thin viscous films: Regularity and long-time behaviour of weak solutions*, Comm. Pure Appl. Math., 49 (1996), pp. 85–123.
[7] A. V. Bobylev, J. A. Carrillo, and I. Gamba, *On some properties of kinetic and hydrodynamic equations for inelastic interactions*, J. Statist. Phys., 98 (2000), pp. 743–773.
[8] A. V. Bobylev, I. M. Gamba, and V. Panferov, *Moment inequalities and high-energy tails for Boltzmann equations with inelastic interactions*, J. Statist. Phys., 116 (2004), pp. 1651–1682.
[9] C. Buet, S. Cordier, and V. Dos Santos, *A conservative and entropy scheme for a simplified model of granular media*, Transport Theory Statist. Phys., 33 (2004), pp. 125–155.
[10] E. Caglioti and C. Villani, *Homogeneous cooling states aren't always good approximations to granular flows*, Arch. Ration. Mech. Anal., 163 (2002), pp. 329–343.
[11] S. Campbell, *Rapid granular flows*, Ann. Rev. of Fluid Mech., 22 (1990), pp. 57–92.
[12] J. A. Carrillo, *On a 1-D granular media immersed in a fluid*, Fields Inst. Commun., 27 (2000), pp. 43–56.

[13] J. A. Carrillo, C. Cercignani, and I. M. Gamba, *Steady states of a Boltzmann equation for driven granular media*, Phys. Rev. E, 62 (2000), pp. 7700–7707.
[14] J. A. Carrillo, R. J. McCann, and C. Villani, *Kinetic equilibration rates for granular media and related equations: Entropy dissipation and mass transportation estimates*, Rev. Mat. Iberoamericana, 19 (2003), pp. 971–1018.
[15] J. A. Carrillo and G. Toscani, *Wasserstein metric and large-time asymptotics of nonlinear diffusion equations*, in New Trends in Mathematical Physics, World Scientific, Hackensack, NJ, 2004, pp. 234–244.
[16] J. A. Carrillo and G. Toscani, *Long-time asymptotics for strong solutions of the thin film equation*, Comm. Math. Phys., 225 (2002), pp. 551–571.
[17] L. Gosse and F. James, *Numerical approximation of linear one-dimensional conservation equations with discontinuous coefficients*, Math. Comp., 69 (2000), pp. 987–1015.
[18] L. Gosse, *Time-splitting schemes and measure source terms for a quasilinear relaxing system*, Math. Models Methods Appl. Sci., 13 (2003), pp. 1081–1101.
[19] L. Gosse and G. Toscani, *Identification of asymptotic decay to self-similarity for one-dimensional filtration equations*, SIAM J. Numer. Anal., 43 (2006), pp. 2590–2606.
[20] S. Gottlieb, C.-W. Shu, and E. Tadmor, *Strong stability-preserving high-order time discretization methods*, SIAM Rev., 43 (2001), pp. 89–112.
[21] G. Grün and M. Rumpf, *Nonnegativity-preserving convergent schemes for the thin films equation*, Numer. Math., 87 (2000), pp. 113–152.
[22] S. Jin, *Efficient asymptotic-preserving (AP) schemes for some multiscale kinetic equations*, SIAM J. Sci. Comput., 21 (1999), pp. 441–454.
[23] D. Kinderlehrer and M. Kowalczyk, *Diffusion-mediated transport and the flashing ratchet*, Arch. Ration. Mech. Anal., 161 (2002), pp. 149–179.
[24] H. Li and G. Toscani, *Long-time asymptotics of kinetic models of granular flows*, Arch. Ration. Mech. Anal., 172 (2004), pp. 407–428.
[25] T. G. Myers, *Thin films with high surface tension*, SIAM Rev., 40 (1998), pp. 441–462.
[26] G. Naldi, L. Pareschi, and G. Toscani, *Spectral methods for a singular Boltzmann equation for granular flows and numerical quasi-elastic limit*, M2AN Math. Model. Numer. Anal., 37 (2003), pp. 73–90.
[27] S. McNamara and W. R. Young, *Kinetics of a one-dimensional granular medium in the quasi-elastic limit*, Phys. Fluids A, 74 (1993), pp. 34–45.
[28] L. Pareschi, *On the fast evaluation of kinetic equations for driven granular media*, in Numerical Analysis and Advanced Applications—Proceedings of ENUMATH 2001, The 4th Conference on Numerical Mathematics and Advanced Applications, Ischia, Italy, 2001, Springer-Italia, Milan, 2003.
[29] L. Pareschi, G. Russo, and G. Toscani, *A kinetic approximation of Hele–Shaw flow*, C. R. Math. Acad. Sci Paris, 338 (2004), pp. 177–182.
[30] G. Toscani, *One-dimensional kinetic models of granular flows*, M2AN Math. Model. Numer. Anal., 34 (2000), pp. 1277–1291.
[31] G. Toscani, *A central limit theorem for solutions of the porous medium equation*, J. Evol. Equ., 5 (2005), pp. 185–203.
[32] C. Villani, *Topics in Optimal Mass Transportation*, Grad. Stud. Math. 58. AMS, Providence, RI, 2003.
[33] G. W. Wei, *Wavelets generated by using discrete singular convolution kernels*, J. Phys. A, 33 (2000), pp. 8577–8596.
[34] L. Zhornitskaya and A. L. Bertozzi, *Positivity-preserving numerical schemes for lubrication-type equations*, SIAM J. Numer. Anal., 37 (2000), pp. 523–555.

SIAM J. SCI. COMPUT.
Vol. 28, No. 4, pp. 1228–1251
© 2006 Society for Industrial and Applied Mathematics

ACCURACY PROPERTIES OF THE WAVE-RAY MULTIGRID ALGORITHM FOR HELMHOLTZ EQUATIONS*

IRENE LIVSHITS† AND ACHI BRANDT‡

Abstract. Helmholtz equations with their highly oscillatory solutions play an important role in physics and engineering. These equations present the main computational difficulties typical to acoustic, electromagnetic, and other wave problems. They are often accompanied by radiation boundary conditions and are considered on infinite domains. Solving them numerically using standard procedures, including multigrid, is too expensive. The wave-ray multigrid algorithm efficiently solves the Helmholtz equations and naturally incorporates the radiation boundary conditions. Important accuracy properties of the wave-ray solver are discussed in this paper.

Using various mode analyses, we show that, with the right choice of parameters, this algorithm can obtain an approximation to the differential solution with accuracy that equals the accuracy of the target grid discretization. Moreover, the boundary conditions can be introduced with any desired accuracy. Our theoretical conclusions are confirmed by numerical experiments.

Key words. Helmholtz equations, accuracy, multigrid, wave-ray

AMS subject classifications. 65N15, 65N55, 35C99

DOI. 10.1137/040620461

1. Introduction.

These studies are aimed at constructing fast and accurate numerical algorithms for the Helmholtz equation

$$\Delta u(x,y) + k^2 u(x,y) = f(x,y), \qquad (x,y) \in \mathbf{R}^2, \tag{1.1}$$

with the radiation boundary conditions. Our goal is to approximate the solution of (1.1) on a finite domain Ω of diameter d. We primarily consider domains with a d that satisfies $kd \gg 1$. This makes the problem highly indefinite and hard to solve.

Although the solution process we describe can solve more general problems, our analysis here is focused on the model problem (1.1) when the wave number k is constant, $f(x,y)$ has compact support $\Omega_f \subset \mathbf{R}^2$, and employed is the second-order discretization of (1.1),

$$(L^h u^h)_{i,j} = \frac{u^h_{i-1,j} + u^h_{i+1,j} + u^h_{i,j-1} + u^h_{i,j+1} - 4u^h_{i,j}}{h^2} + k^2 u^h_{i,j} = f_{i,j}, \tag{1.2}$$

where $u^h_{i,j}$ approximates $u(x_i, y_j)$ and $f_{i,j} \approx f(x_i, y_j)$, $i = 0, \ldots, N_x$, $j = 0, \ldots, N_y$, $\Omega_f \subset \Omega$. (Helmholtz equations (1.1) with radiation boundary conditions are usually considered on infinite domains. Therefore we assume, without a loss of generality, that Ω is considerably wider than Ω_f.)

*Received by the editors December 8, 2004; accepted for publication (in revised form) February 2, 2006; published electronically July 31, 2006.

http://www.siam.org/journals/sisc/28-4/62046.html

†Department of Mathematical Sciences, Ball State University, Muncie, IN 47306 (ilivshits@bsu.edu). The work of this author was supported in part by National Science Foundation grant DMS-0521097.

‡Department of Mathematics & Computer Science, Weizmann Institute of Science, Rehovot 76100, Israel (aci.brandt@weizmann.ac.il). The work of this author was supported in part by Israel Science Foundation grant 295/01.

For simplicity, Ω is chosen to be the square

$$\Omega = \{(x, y), -d/2 \le x \le d/2, -d/2 \le y \le d/2\}, \tag{1.3}$$

and hence $N_x = N_y = N$, $x_i = -d/2 + ih$, and $y_j = -d/2 + jh$.

For $kd \gg 1$, the cost of solving (1.2) can be overwhelming. To provide an accurate approximation to the differential solutions, discretization of (1.1) must employ many, actually more than $O(k^3d^3)$, grid points. Moreover, to converge to the numerical solution, a conventional solver should employ many, actually more than $O(k^6d^6)$, sweeps of a slow (e.g., Kaczmarz) relaxation scheme, since faster schemes diverge for some components. Thus the total work grows at least proportionally to $O(k^9d^9)$.

The above provides motivation for the use of multigrid methods [11], [13], [15], [17]. However, standard multigrid procedures are not efficient for the highly indefinite Helmholtz equations. This is because there is a range of Fourier error components $e^{i(\omega_1 x+\omega_2 y)}$ that the standard multigrid cycle cannot reduce efficiently. Those are the components with $\omega_1^2 + \omega_2^2 \approx k^2$; we call them [illegible]. On fine grids their amplitudes are much larger than the amplitudes of the corresponding normalized residuals, making them almost invisible to any local processing. Hence, they are only very slowly reduced by relaxation. On the other hand, unlike regular smooth components, these have poor approximations on coarse grids, which, therefore, cannot serve to correct the finer-grid solutions.

As a result, the characteristic components need a special multigrid treatment, and this is the basis of the wave-ray algorithm. Its fast convergence was the subject of [9]. There it was shown that the algorithm has high multigrid efficiency at a cost comparable to the cost of solving a Poisson equation with the standard cycle.

In the present paper we study the accuracy aspects of the wave-ray algorithm. The goal of this algorithm, employing the second-order discretization, is to provide accurate $O(h^2)$ approximation to the differential solution while costing only $O(h^{-2})$ computer operations. This is not as straightforward as for definite elliptic systems, both due to the phase errors and because the radiation boundary conditions need to be brought accurately from the coarsest level of the algorithm to the target level.

Obviously, by choosing a smaller finest grid mesh-size h, we would like to obtain a more accurate solution, i.e., one that is closer to the solution of differential equation (1.1). Moreover, since the second-order discretization scheme is employed in (1.2), the accuracy is expected to improve as $O(h^2)$. In the wave-ray algorithm, however, the following scenario is possible: after choosing a smaller h, the approximation error, the difference between the differential and the wave-ray solution, is almost unreduced. This property, not typical for the standard multigrid methods, occurs because of the extended role of the coarse grids in the wave-ray algorithm. This role is different because here the coarse grids do not produce just a correction to smooth error components, but the principal components of the solution itself. Moreover, these components are almost invisible on other grids (due to their small residuals) and therefore are not updated there. In addition, the radiation boundary conditions are introduced on the coarsest ray grid, and they are not corrected on any finer grids (carrying on interpolation errors). Therefore, processing on coarse ray grids can cause significant errors that overshadow the discretization error on the finest grid, governed by the finest mesh-size.

In this paper we discuss different factors and the errors they cause that can lead to such [illegible]. We then suggest ways to deal with them. The ultimate goal is to minimize the effect of these errors on the total error to the level

of becoming negligible compared to the finest grid discretization error, which then regains its principal error term status (as in traditional multigrid solvers).

In this study we do not assume any restriction on the finest grid mesh-size, and therefore we avoid the difficult task of dealing with the effect of pollution there (the work on fine grid pollution can be found, for example, in [4], [12], [14]). The problem of pollution in our algorithm appears on the coarse ray grids, where it is treated by properly choosing different parameters of the algorithm. These are discussed in detail.

The outline of the paper is as follows: We start with a description of the algorithm. Its efficiency is achieved by using two types of solution representation: wave and ray. (This gave the algorithm its name `wave-ray`. The grids on which the wave and the ray equations are discretized are called the [illegible] and the [illegible], respectively.) This technique can cause inaccuracies, unless the algorithmic parameters are properly chosen.

In section 2 we discuss discretization errors and their influence on the choice of the finest mesh-sizes in the wave and in the ray discretization. Unlike in standard multigrid procedures, coarse ray grids often (away from Ω_f) provide more accurate approximation to the differential solution than the finer ray or wave representations.

In section 3 our subject is the approximation of the differential problem on an infinite domain by a solver which employs finite domains on many levels. In particular, we deal with the question of how to impose the radiation boundary conditions.

We conclude the paper with numerical results and discuss their agreement with our theoretical conclusions.

1.1. Wave-ray algorithm for Helmholtz equations. First, we define the circle of [illegible] Fourier components to be the subset of characteristics components with frequencies $\omega_1^2 + \omega_2^2 = k^2$. They satisfy (1.1) with $f \equiv 0$ and have a significant role in the algorithm. Among all principal components we consider a finite set of [illegible] principal components uniformly spaced on the principal circle. We denote them (k_1^ℓ, k_2^ℓ), $\ell = 1, \ldots, L$; L is the number of lattice components. We use the word `lattice` to describe the discretization of the principal circle and to distinguish it from `grids` in the physical space. For the principal lattice component $e^{i(k_1^\ell x + k_2^\ell y)}$, we define a rotated Cartesian coordinate system (ξ^ℓ, η^ℓ), with ξ^ℓ being parallel to (k_1^ℓ, k_2^ℓ), the [illegible] direction.

Each characteristic component can be represented as a product of a lattice principal component and a smooth envelope function. Hence, a general representation of the error that cannot be efficiently reduced by a regular multigrid cycle can be given by

$$v(x,y) = \sum_{\ell=1}^{L} \hat{v}^\ell(x,y) e^{i(k_1^\ell x + k_2^\ell y)}, \tag{1.4}$$

where $\hat{v}^\ell(x,y)$ are smooth functions. Note that the functions $\hat{v}^\ell(x,y)$ are not uniquely defined: Fourier components of v with frequencies between two lattice points can be assigned to these points with arbitrary weights, yet still yielding the desired smoothness. Considered in the corresponding rotated coordinates, the scale of smoothness of $\hat{v}^\ell(x,y)$ is $O(L^2/k)$ in the propagation direction ξ^ℓ, and $O(L/k)$ in the η^ℓ direction. (In other words, the function $v(x,y)$ essentially consists of Fourier components with frequencies in a ring of width $O(k/L^2)$ around the principal circle.)

We call the functions $\hat{v}^\ell(x,y)$ the [illegible]; the equations we derive for them, the [illegible]. The residual ray equation for $\hat{v}^\ell(x,y)$ is

$$\Delta \hat{v}^\ell(x,y) + 2ik_1^\ell \frac{\partial}{\partial x}\hat{v}^\ell(x,y) + 2ik_2^\ell \frac{\partial}{\partial y}\hat{v}^\ell(x,y) = \hat{r}^\ell(x,y). \tag{1.5}$$

This results directly from the Helmholtz equation, with $\hat{r}^\ell(x,y)$ being the smooth weights in the ray representation of the wave residual function:

$$Lv(x,y) = r(x,y) = \sum_{\ell=1}^{L} \hat{r}^\ell(x,y) e^{i(k_1^\ell x + k_2^\ell y)}. \tag{1.6}$$

Each principal component $e^{i(k_1^\ell x + k_2^\ell y)}$ in its rotated coordinates has the form $e^{ik\xi^\ell}$, and thus (1.5) can be simplified there to

$$\Delta \hat{v}^\ell(\xi^\ell,\eta^\ell) + 2ik\frac{\partial}{\partial_{\xi^\ell}}\hat{v}^\ell(\xi^\ell,\eta^\ell) = \hat{r}^\ell(\xi^\ell,\eta^\ell), \quad (\xi^\ell,\eta^\ell) \in \hat{\Omega}^\ell. \tag{1.7}$$

Since the problem is defined over the entire space $(x,y) \in \mathbf{R^2}$, we have to define a large enough finite computational domain $\hat{\Omega}^\ell$ over which $\hat{v}^\ell$ will be computed. For convenience we choose $\hat{\Omega}^\ell$ to be rectangular,

$$\hat{\Omega}^\ell = \{(\xi^\ell,\eta^\ell) : \xi_0^\ell \le \xi^\ell \le \xi_1^\ell, -\eta_0^\ell \le \eta^\ell \le \eta_0^\ell\}, \tag{1.8}$$

where appropriate values for ξ_0^ℓ, ξ_1^ℓ, and η_0^ℓ will be a subject of the analysis below. We call the set of points (ξ^ℓ,η^ℓ) on the lines $\xi^\ell = \xi_0^\ell$, $\xi^\ell = \xi_1^\ell$, and $\eta^\ell = \pm\eta_0^\ell$ the `entrance`, the `exit`, and the `side` boundaries, respectively.

(Each equation (1.7) in the rotated coordinates allows the most aggressive coarsening ratio: by the factor of four (instead of two) in the propagation direction. The use of the standard Cartesian (x,y) coordinates is also possible. Moreover, it too allows a special coarsening for at least some components: for instance, for $L = 8$, one half of the components propagate in either x or y direction, and therefore for them the same coarsening can be applied. For a higher value of L, for most of the components their propagation direction is closer to either x or y, and therefore some stronger coarsening can be applied there, with the factors of three or four, depending on the component in question and the accuracy requirements.)

The error for which the usual multigrid cycle is not effective can efficiently be reduced by a multigrid procedure which on coarse grids represents the smooth ray functions $\hat{v}^\ell(x,y)$ rather than the nonsmooth function $v(x,y)$ itself. The coarser the grid, the larger the scale of smoothness of $\hat{v}^\ell(x,y)$ represented on it. This requires a larger L, i.e., a finer lattice of principal components.

Let the ray levels be enumerated by $n = 1,\dots,N$ so that $n = N$ is the level with the coarsest grid and the finest lattice. For the nth level, the principal lattice is then composed of the frequencies

$$(k_1^{n,\ell}, k_2^{n,\ell}) = (k\cos\theta^{n,\ell}, k\sin\theta^{n,\ell}), \qquad \ell = 1,\dots,L^n, \tag{1.9}$$

where $\theta^{n,\ell} = 2\pi\ell/L^n$; the corresponding ray functions are denoted $\hat{v}^{n,\ell}$. At each level n, each ray function $\hat{v}^{n,\ell}(x,y)$ in its rotated coordinates $(\xi^{n,\ell},\eta^{n,\ell})$ is discretized on a uniform grid with mesh-sizes $h_\xi^n = O((L^n)^2/k)$ and $h_\eta^n = O(L^n/k)$. Our actual choice of parameters, based on a careful mode analysis, has been

$$L^n = 2^{n+2}, \qquad h_\xi^n = C(L^n)^2/(32k), \qquad h_\eta^n = C(L^n)/(8k), \tag{1.10}$$

where typically $1.25 \le C \le 2.5$.

The wave-ray cycle is composed of two parts. The first part, the wave subcycle, is a standard multigrid cycle applied to (1.2). It proceeds from the finest wave grid to

the coarsest; there is actually no need to go to grids with mesh-size coarser than $4/k$ (this is because for such h the system (1.2) becomes so strongly diagonal dominant that the Gauss–Seidel relaxation converges very fast). Few Kaczmarz or Gauss–Seidel relaxation sweeps are performed on each wave level except for the ones with mesh-sizes $h \approx 2/k$. Relaxation on the coarsest grid reduces the smoothest part of the error, while relaxation on the fine levels reduce its high-frequency components. The wave subcycle does not change the characteristics components effectively, and consequently it leaves an error of the type (1.4). Also it does not treat the radiation boundary conditions, which are associated only with principal components and, therefore, ray representation. In this part of the algorithm the sole role of the coarse grids is to produce smooth noncharacteristic corrections to the finer grid approximations in the interior of the finer grid computational domains.

The second part of the algorithm starts with a ray subcycle, which is needed to reduce characteristic errors and to impose the radiation boundary conditions. First, the wave residual $r(x, y)$ is calculated on some sufficiently fine wave level with mesh-size $h = O(d^{-1/2}k^{-3/2})$—still fine enough to produce small phase errors throughout the computational domain. From this grid, the wave residual is transferred directly to the wave grid with $h \approx 1/k$. There, for each $1 \le \ell \le L^1$ this wave residual is multiplied by $e^{-i(k_1^\ell x + k_2^\ell y)}$, resulting in

$$r(x,y)e^{-i(k_1^\ell x+k_2^\ell y)} = \hat{r}^\ell(x,y) + \sum_{m\neq\ell} \hat{r}^m(x,y)e^{i(\Delta k_1^m x+\Delta k_2^m y)}, \tag{1.11}$$

where $k_j^m = k_j^{1,m}$, $\Delta k_j^m = k_j^m - k_j^\ell$, $j = 1, 2$. Notice that the first term on the right-hand side (RHS) of (1.11) is very smooth compared to each term of the sum. The next step is to approximate each $\hat{r}^\ell$ in the rotated coordinates $(\xi^\ell, \eta^\ell) = (\xi^{1,\ell}, \eta^{1,\ell})$. For this purpose a procedure called [illegible] is applied, denoted

$$\hat{r}^\ell = \hat{r}^{1,\ell} = W_{1,\ell}(r(x,y)e^{-i(k_1^\ell x+k_2^\ell y)}), \quad \ell = 1, \dots, L^1. \tag{1.12}$$

Here $W_{1,\ell}$ is a product of two weighting operators that accurately transfer smooth components from the wave grid with mesh-size $h \approx 1/k$ first to an intermediate ray grid with mesh-size $(h_\xi, h_\eta) \approx (2/k, 2/k)$, and then to the ray grid $(\xi^{1,\ell}, \eta^{1,\ell})$ with mesh-size $(h_\xi^1, h_\eta^1) \approx (4/k, 2/k)$. The operators are designed to almost annihilate components that are highly oscillatory on the current scale.

It is important to note here that the entire solver employs the full approximation scheme (FAS; see, e.g., [8]). This means that, although the equations at each level are basically equations for the correction $\hat{v}^{n,\ell}$, driven by the next-finer-level residuals, as in (1.7), they are modified (by adding an appropriate term to the RHS) so that their solution always represents the full approximation $\hat{u}^{n,\ell} = \hat{v}^{n,\ell} + \tilde{u}^{n,\ell}$, where $\tilde{u}^{n,\ell}$ is an approximation to $\hat{u}^{n,\ell}$. The algorithm employs the value of $\hat{u}^{n,\ell}$ at the end of the previous cycle, as the first approximation to $\tilde{u}^{n,\ell}$. Such approximation satisfies the radiation boundary condition, and it does not require an update from the wave part of the cycle, since characteristic components are not changed there. At the first cycle, $\tilde{u}^{n,\ell}$ satisfies the radiation boundary conditions at the entrance and side boundaries, and it is zero elsewhere.

The FAS RHS is calculated by

$$\hat{f}^{n,\ell} = \hat{r}^{n,\ell} + \hat{L}^n \tilde{u}^{n,\ell}. \tag{1.13}$$

The FAS is chosen to enable the imposition of the radiation boundary conditions at the coarsest ray level.

As the model boundary conditions, we have chosen the Dirichlet radiation boundary conditions, which can be formulated as follows: for each propagation direction ξ, the amplitude $\hat{u}_0$ of the principal component $e^{ik\xi}$ as it propagates from infinity into Ω is given. This amplitude, of course, gives the initial values of the corresponding ray function at the entering boundaries. These boundary conditions are very suitable for our investigation, since they allow explicit analysis of different error types on the ray grids. Similar to the Sommerfeld radiation boundary conditions, these Dirichlet boundary conditions control the size of entering principal components. No boundary conditions are imposed for the exiting waves, which are simply allowed to leave the domain (without causing any reflection). The rest of the solution is uniquely defined by the RHS.

After the discrete ray equations are relaxed on the finest ray level ($n = 1$) and the new solution approximations are calculated, the ray subcycle proceeds (if necessary) to increasingly coarser ray levels ($n = 2, \dots, N$). This includes recursive derivation of ray equations on increasingly coarser ray grids, having increasingly finer lattices (larger $L = L^n$). The equations on each grid have the ray form (1.7), discretized; they are the result of a separation process applied to the residuals of the next finer neighboring grids. Namely, to calculate $\hat{r}^{n,\ell}$ for odd ℓ, the corresponding ray residual function from the finer grid is used:

$$\hat{r}^{n,\ell} = W^{n,\ell}_{n-1,(\ell+1)/2} \hat{r}^{n-1,(\ell+1)/2}, \tag{1.14}$$

where here and below $W^{n,\ell}_{n-1,j}$ consists again of a product of two weighting operators, acting from the grid $(\xi^{n-1,j}, \eta^{n-1,j})$ to the target $(\xi^{n,\ell}, \eta^{n,\ell})$ designed to approximate the smooth components and annihilate high frequencies on the considered scale.

For even ℓ, $\hat{r}^{n,\ell}$ is calculated from two finer grid residuals:

$$\begin{aligned} \hat{r}^{n,\ell} = {} & W^{n,\ell}_{n-1,\ell/2} \big(\hat{r}^{n-1,\ell/2} e^{i((k_1^{n-1,\ell/2} - k_1^{n,\ell})x + (k_2^{n-1,\ell/2} - k_2^{n,\ell})y)} \big) \\ & + W^{n,\ell}_{n-1,\ell/2+1} \big(\hat{r}^{n-1,\ell/2+1} e^{i((k_1^{n-1,\ell/2+1} - k_1^{n,\ell})x + (k_2^{n-1,\ell/2+1} - k_2^{n,\ell})y)} \big). \end{aligned} \tag{1.15}$$

The lattice count should be taken modulo L; i.e., for $\ell = L^n$ take $\ell/2+1 = L^{n-1}+1 \equiv 1 (\mathrm{mod}\, L^{n-1})$.

On the coarsest grid, $n = N$, the radiation boundary conditions are imposed, facilitated by the nearly pure ray representation at that grid, in the following way. (For the sake of explanation, we define the [illegible] as follows: if a principal component in the (ξ, η) coordinates has a frequency $k(\cos\alpha, \sin\alpha)$, then α is its inclination angle on the corresponding grid.) The radiation boundary conditions for the ray domain associated with the principal direction $(\xi^{n,\ell}, \eta^{n,\ell})$ are the Dirichlet boundary conditions for the [illegible] and the [illegible] boundaries. Each principal lattice component represents a range of principal components with inclination angles $\alpha \in [-\Delta\theta^n, \Delta\theta^n]$, $\Delta\theta^n = 2\pi/2L^n$. The values of the entering rays for all such components are summed at the entrance $\xi = \xi_0$ and the side $\eta = \pm\eta_0$ boundaries. The boundary conditions are then defined as

$$\hat{u}^{n,\ell}(\xi_0, \eta) = \int_{-\Delta\theta^n}^{\Delta\theta^n} \hat{u}_{0,\alpha} e^{i(\theta_1^\alpha \xi_0 + \theta_2^\alpha \eta)}\, d\alpha$$

at the entrance boundary, and

$$\hat{u}^{n,\ell}(\xi, -\eta_0) = \int_{0}^{\Delta\theta^n} \hat{u}_{0,\alpha} e^{i(\theta_1^\alpha \xi - \theta_2^\alpha \eta_0)}\, d\alpha$$

and

$$\hat{u}^{n,\ell}(\xi,\eta_0)=\int_{-\Delta\theta^n}^{0}\hat{u}_{0,\alpha}e^{i(\theta_1^\alpha\xi+\theta_2^\alpha\eta_0)}\,d\alpha$$

at the side boundaries. Here $\hat{u}_{0,\alpha}$ is the amplitude of the entering principal components with the inclination angle α, and $(\theta_1^\alpha,\theta_2^\alpha)=k(\cos\alpha-1,\sin\alpha)$.

The ray equations are solved there quickly by a line Gauss–Seidel relaxation. Then the boundary conditions along with corrections are interpolated through the intermediate ray grids (without any relaxation) to the finest ray grid.

Interpolation from a coarser ray grid (with a finer lattice) to the next-finer ray grid (with coarser lattice) is performed differently on the boundary of the finer grid domain and in its interior. On the boundary $\hat{\Gamma}^{n,\ell}$ of the computational domain $\hat{\Omega}^{n,\ell}$ the solution values are interpolated directly from the three neighboring coarser grids:

$$\begin{aligned}\hat{u}^{n,\ell}=&\frac{1}{2}I^{n,\ell}_{n+1,2\ell-2}\big(\hat{u}^{n+1,2\ell-2}e^{i((k_1^{n+1,2\ell-2}-k_1^{n+1,2\ell-1})x+(k_2^{n+1,2\ell-2}-k_2^{n+1,2\ell-1})y)}\big)\\&+I^{n,\ell}_{n+1,2\ell-1}\hat{u}^{n+1,2\ell-1}\\&+\frac{1}{2}I^{n,\ell}_{n+1,2\ell}\big(\hat{u}^{n+1,2\ell}e^{i((k_1^{n+1,2\ell+1}-k_1^{n+1,2\ell-1})x+(k_2^{n+1,2\ell}-k_2^{n+1,2\ell-1})y)}\big),\end{aligned}$$

where the interpolation operator $I^{n,\ell}_{n+1,j}$ interpolates from the coarser grid $\hat{\Omega}^{n+1,j}$ to the finer grid $\hat{\Omega}^{n,l}$. This interpolation of the boundary conditions is the reason for using FAS, since it requires us to approximate solutions rather than corrections on all grids.

In the interior, the solution is only [illegible] by the coarse-grid approximations:

$$\begin{aligned}\hat{u}^{n,\ell}=\hat{u}^{n,\ell}&+\frac{1}{2}I^{n,\ell}_{n+1,2\ell-2}\big(\hat{v}^{n+1,2\ell-2}e^{i((k_1^{n+1,2\ell-2}-k_1^{n+1,2\ell-1})x+(k_2^{n+1,2\ell-2}-k_2^{n+1,2\ell-1})y)}\big)\\&+I^{n,\ell}_{n+1,2\ell}\hat{v}^{n+1,2\ell}+\frac{1}{2}I^{n,\ell}_{n+1,2\ell}\big(\hat{v}^{n+1,2\ell}e^{i((k_1^{n+1,2\ell}-k_1^{n+1,2\ell-1})x+(k_2^{n+1,2\ell}-k_2^{n+1,2\ell-1})y)}\big).\end{aligned}$$

Here corrections $\hat{v}^{n+1,j}$ are given by $\hat{v}^{n+1,j}=\hat{u}^{n+1,j}-\tilde{u}^{n+1,j}$, where $\tilde{u}^{n+1,j}$ again is the value of $\hat{u}^{n+1,j}$ at the end of the previous cycle.

No relaxation is performed on the way to the finest ray grid. From there, the ray solutions (for the boundary) and the ray corrections (for the interior) are interpolated to the wave grid with $h\approx 1/k$, where they are summed as in (1.4) and used to correct the previous wave solution. Similar to the ray-to-ray interpolation, the wave correction (at the interior) and the solution values (at the boundary) are interpolated to the finer wave grids, with a small number of relaxation sweeps performed on each grid.

See the appendix below for a formal description of the entire algorithm.

2. Discretization errors. In this section we discuss how to achieve an accurate approximation to the differential solutions in the interior of the finest computational domain. We examine the pointwise approximation to the differential solutions, analyze the influence of discretization errors, and conclude that the main source of inaccuracy is the phase errors, coming from both the wave and the ray representations. In our studies we measure only the errors eventually resulting in the interior of the target finest wave grid.

2.1. Phase errors. The [illegible] error is the difference in phases of the discrete and differential waves, accumulated along their propagation path due to their different wavelengths. This error can become very significant when the solution propagates along many wavelengths.

A peculiarity of our algorithm is that, unlike common discretization schemes, in the regions where the equation is homogeneous, the coarse ray grids provide more accurate approximations to the differential solution than the finer ray, and often are better than the wave grids. The ultimate role of the fine wave grids is to resolve the RHS f at scales smaller than or just comparable to the wavelength $2\pi/k$. The details are discussed in this section.

2.1.1. Wave phase error. We define a [illegible] $e^{i(k_1^h x + k_2^h y)}$ as any one that satisfies the discrete homogeneous equations (1.2),

$$L^h e^{i(k_1^h x + k_2^h y)} = 0. \tag{2.1}$$

By Taylor expansion in (1.2) it can be shown that, for $kh \ll 1$, the value of the corresponding discrete wave number $|k^h| = \sqrt{(k_1^h)^2 + (k_2^h)^2}$ can be approximated by

$$|k^h| \approx k\left(1 + \frac{k^2h^2}{\gamma}\right), \qquad 24 \le \gamma \le 48, \tag{2.2}$$

where the exact value of γ depends on the ratio $k_1^h : k_2^h$. The accumulated value of the relative phase error (the phase error over the period $2\pi/k$) propagating through the $kd/2\pi$ wavelengths of Ω can therefore be estimated as

$$E(kd, kh) \approx kd\frac{k^2h^2}{2\pi\gamma}. \tag{2.3}$$

To provide an accurate approximation to the differential solution on the target wave grid, the finest mesh-size h should be chosen so that $E(kd, kh)$ is small:

$$E(kd, kh) \ll 1. \tag{2.4}$$

2.1.2. Ray phase error. In standard multigrid procedures the accuracy of discretization on coarse grids influences the speed of convergence of the solver rather than the accuracy of the solution. In our algorithm the ray grids serve not only to correct the characteristic error components, but also to introduce the radiation boundary conditions and to approximate the principal components invisible on the finest wave grids. Hence, the quality of the solution approximation on the ray grids needs to be addressed. (Since the discussion below is relevant for any ℓth ray function and ray operator on any nth level, we omit the superscripts ℓ and n.)

We define a [illegible] $e^{i(\theta_\xi \xi + \theta_\eta \eta)}$ as any one which satisfies

$$\hat{L}^h e^{i(\theta_\xi \xi + \theta_\eta \eta)} = 0,$$

where $\hat{L}^h$ is a second-order discretization of (1.7), given in the (ξ, η) coordinates by

$$\begin{aligned}\hat{L}^h \hat{v}_{i,j} = {} & \frac{\hat{v}_{i-1/2,j-1} + \hat{v}_{i-1/2,j+1} - 2\hat{v}_{i-1/2,j}}{2h_\eta^2} + \frac{\hat{v}_{i+1/2,j-1} + \hat{v}_{i+1/2,j+1} - 2\hat{v}_{i+1/2,j}}{2h_\eta^2} \\ & + \frac{\hat{v}_{i-3/2,j} - \hat{v}_{i-1/2,j} - \hat{v}_{i+1/2,j} + \hat{v}_{i+3/2,j}}{2h_\xi^2} + 2ik\frac{\hat{v}_{i+1/2,j} - \hat{v}_{i-1/2,j}}{h_\xi}.\end{aligned} \tag{2.5}$$

On the scale of the employed grid, the term $2ikv_\xi$ is the principal term, and its accurate (second-order) discretization is the main factor in our choice of the discrete scheme. Our operator provides an accurate short central discretization for $2ikv_\xi$, and also a second-order accuracy for the remaining terms (it employs the Crank–Nicholson scheme for $v_{\eta\eta}$). On the other hand, the standard five-point stencil for the Laplace operator would come with the long central differences for the principal term leading to a weak diagonal term.

Discretization (2.5) also allows an efficient η-line relaxation (marching from entrance to exit, solving (2.5) simultaneously for all $\hat{v}_{i,j}$ with the same j; cf. section 3.2) on the ray grids. Substitution of $e^{i(\theta_\xi \xi + \theta_\eta \eta)}$ into (2.5) gives

$$\frac{2}{h_\eta^2}\cos\left(\frac{\phi_\xi}{2}\right)\left[\cos(\phi_\eta)-1\right]-\frac{4k}{h_\xi}\sin\left(\frac{\phi_\xi}{2}\right)+\frac{1}{h_\xi^2}\left[\cos\left(\frac{3\phi_\xi}{2}\right)-\cos\left(\frac{\phi_\xi}{2}\right)\right]=0,$$

where $\phi_\xi = \theta_\xi h_\xi$ and $\phi_\eta = \theta_\eta h_\eta$. Note that the highest frequency $(\theta_\xi, \theta_\eta)$, which must be represented on each ray grid (so that any principal and characteristic component has an accurate representation on at least one of the L ray grids), is given by

$$\theta_\xi \approx k\frac{\pi^2}{2L^2}, \quad \theta_\eta \approx k\frac{\pi}{L}. \tag{2.6}$$

This is the frequency associated with the principal component at the midpoint between two neighboring lattice points. With (1.10), the arguments of sines and cosines are sufficiently small for (2.6) to be replaced by its approximation:

$$-\theta_\eta^2+\frac{1}{8}\theta_\eta^2\phi_\xi^2-2k\theta_\xi+\frac{k}{12}\theta_\xi\phi_\xi^2-\theta_\xi^2\approx 0. \tag{2.7}$$

Using (2.7), we can approximate the size of a discrete ray wave number $\hat{k}_L = (\theta_\xi + k, \theta_\eta)$,

$$|\hat{k_L}|^2 = k^2+2k\theta_\xi+\theta_\xi^2+\theta_\eta^2 \approx k^2\left(1+\frac{\theta_\eta^2\phi_\xi^2}{8k^2}+\frac{\theta_\xi\phi_\xi^2}{12k}\right), \tag{2.8}$$

or, by (1.10) and (2.6),

$$|\hat{k}_L| \approx k\left(1+\frac{\beta}{L^2}\right), \tag{2.9}$$

with $\beta \approx 0.039C^2$ and $1.25 \le C \le 2.5$. The phase error arises from the propagation of principal components emanating from the radiation boundary conditions or from the RHS source. For instance, at a distance $\hat{d}$ from the entering boundary, the phase error can be estimated as

$$E(k\hat{d}, L) \approx k\hat{d}\frac{\beta}{2\pi L^2}. \tag{2.10}$$

2.2. RHS truncation errors. The phase error is the main discretization error in approximating the homogeneous equations. To estimate those truncation errors which arise in approximating the effect of a nonhomogeneous RHS, consider an RHS $f(x, y)$, a differential solution $u(x, y)$, and an approximate difference solution $u^h(x, y)$ in Ω of the form

$$f(x,y)=\int_{\mathbf{t}}\hat{f}(\mathbf{t})e^{i(t_1x+t_2y)}d\mathbf{t}, \qquad u(x,y)=\int_{\mathbf{t}}\hat{u}(\mathbf{t})e^{i(t_1x+t_2y)}d\mathbf{t},$$

and

$$u^h(x,y) = \int_{\mathbf{t}} \hat{u}^h(\mathbf{t}) e^{i(t_1 x + t_2 y)} d\mathbf{t}.$$

For any frequency $\mathbf{t} = (t_1, t_2),\ t = |\mathbf{t}|$, the following holds:

$$\hat{u}(\mathbf{t}) = \frac{\hat{f}(\mathbf{t})}{k^2 - t^2} \tag{2.11}$$

and

$$\hat{u}^h(\mathbf{t}) \approx \frac{\hat{f}(\mathbf{t})}{k^2 - t^2(1 - 2t^2h^2/\gamma)}, \tag{2.12}$$

where γ is defined in (2.2). The difference between (2.11) and (2.12) is $O(t^2h^2)$, meaning that the finest wave mesh-size should be fine enough to resolve the highest-frequency components of f that one cares to approximate. More precisely, for $|t - k|/k \geq O(1)$, the relative error

$$E(\mathbf{t}) = \frac{|\hat{u}(\mathbf{t}) - \hat{u}^h(\mathbf{t})|}{|\hat{u}(\mathbf{t})|} \tag{2.13}$$

is small when t^2h^2 is small.

The more severe requirements on the mesh-size arise, however, from the characteristic components, where t is close to k. For $|t - k|/k = \delta \ll 1$, using (2.11)–(2.12), one can show that

$$E(\mathbf{t}) = \frac{k^2h^2(1 + O(\delta)}{\gamma\delta + k^2h^2(1 + O(\delta))} \approx \frac{k^2h^2}{\gamma\delta}. \tag{2.14}$$

For this truncation error to be small, the target mesh-size h should satisfy

$$\frac{k^2h^2}{\gamma\delta_{\min}} \ll 1, \tag{2.15}$$

where $\delta_{\min} = \min_{t=|\mathbf{t}|, \hat{u}(\mathbf{t})\neq 0} \frac{|t-k|}{k}$. The problem (1.2) is considered on a computational domain of size d and a grid with mesh-size h. Only a finite number of Fourier components is distinguished on this grid. The minimal distance between such components, in the phase space, is $2\pi/d$. Therefore, $\delta_{min} = \frac{2\pi}{kd}$, and (2.15) can be replaced by

$$kd\frac{k^2h^2}{2\pi\gamma} \ll 1. \tag{2.16}$$

Note that requirement (2.16) is identical with the condition resulting from the wave phase error. Indeed,

$$E(t) = E(kd, kh)(1 + O(\delta)). \tag{2.17}$$

We have excluded from consideration the special case of resonance, when $\hat{f}/(k^2 - t^2)$ is not integrable, which would require a different formulation in the time domain.

3. Boundary conditions. One of the possible reasons for slowness of relaxation process is inefficient introduction of boundary conditions. If done on a sufficiently fine grid, it takes many (expensive) iterations until the boundary information propagates from the boundaries into the domain. The remedy from such slowness is to use coarse grids. This, however, is not a good solution for the standard multigrid solvers applied to the Helmholtz equation, since coarse wave grids are not suitable for representing accurate solutions. Instead, the wave-ray algorithm employs its ray grids to introduce the boundary conditions. These grids are very coarse but very accurately resolve the ray functions.

In this paper we consider the Dirichlet radiation boundary conditions given by the amplitudes of the waves incoming to the computational domain from infinity. These boundary conditions are actually formulated as Dirichlet boundary conditions for the ray functions: at the coarsest ray level each entering ray is represented on the grid (or divided between two grids) with the closest propagation direction, on which its boundary values are indeed very smooth. Let us note that all exiting-only rays (rays which do not enter the computational domain but are originated in Ω_f) can actually exit only through the exit boundaries of our rotated ray domains. Hence on all other boundaries of each of the ray domains we can impose as boundary conditions the incoming rays (or zero, if no incoming rays are assumed). No boundary conditions are needed at the exit boundaries since the discrete equations (2.5) and the order in which we relax them (from entrance to exit) ensure that information propagation in the negative ξ direction is effectively prohibited.

The wave-ray approach can also be used for an efficient introduction of other types of radiation boundary conditions (RBC), for instance, Sommerfeld-like boundary conditions (discussed, for instance, in [2], [3]). This can be done since near the boundaries (remote from Ω_f), any solution is nearly purely of a ray nature and can be represented in the ray form of type (1.4). Then, from the RBC formulated as differential equations for a wave function, it is easy to find a set of differential equations that describe boundary conditions for its ray functions. This set can be separated into conditions for individual components or, at most, for a pair of components with opposite propagation direction. In this case, these two components should be approximated together, still very efficiently since considered on coarse grids. The transition to these boundary conditions has not yet been done by the authors.

The RBC can be introduced by imposing the values of the ray functions $\hat{v}^\ell$ in (1.7) at the entrance and the side boundaries on the coarsest ray level as Dirichlet boundary conditions. Since $kh_\xi \gg 1$ on all ray levels (in fact, on the finest ray grid, the mesh-size in the propagation direction satisfies $kh_\xi \approx 4$; moreover, each time the algorithm proceeds to the next coarser ray grid with double L, the size of kh_ξ increases by a factor of four), the second derivative term $\hat{v}_{\xi\xi}$ in (1.7) is negligible on the scale (h_ξ, h_η), making (1.7) almost parabolic. Therefore, no condition is needed at the exit. There, instead of (2.5), the algorithm employs strictly upstream discretization, and the solution values are treated as the interior rather than the boundary. Treating the exit in this manner, as if there is no boundary at all, allows the exiting waves to leave the domain without reflection, somewhat similar to the effect of the Sommerfeld boundary conditions.

To guarantee an accurate introduction of the RBC, we have to avoid a possible influence of the interior solution on the boundary values, which can in its turn affect the solution in the domain of interest. First we discuss how an implementation of the wave-ray algorithm can pollute the boundary conditions, and how to introduce the RBC accurately—on finite domains and with a finite number of ray functions.

The second question is how to transfer these conditions to the finer ray, and then wave, grids. In our algorithm the wave solution on the boundary is evaluated by direct interpolation from the coarser (eventually, ray) grids. It means that some interpolation error is necessarily introduced on the boundaries and affects the solution accuracy there and in the interior. However, we will show that this influence decreases rapidly as the error propagates inside the computational domain. We analyze the interpolation from both wave and ray domains.

3.1. Side boundaries: Reflected rays. When imposing the RBC at the coarsest ray level, at the entrance these conditions are naturally defined as Dirichlet boundary conditions without having any negative effects. However, this is not so for the side boundaries, because on each ray grid there are principal components that actually exit through one or the other side boundary.

In the assignment of the RBC at the coarsest ray level, each principal component with inhomogeneous boundary conditions is distributed only between the two neighboring lattice components; hence on each of the two corresponding ray grids its inclination is less than $2\pi/L$. The distribution weights are proportional to the distances between the corresponding principal frequency and the lattice points in the phase space. Hence, on each ray grid only principal components with inclinations in the range $|\alpha| \le 2\pi/L$ are represented.

Let us consider, for example, the side boundary $\eta = -\eta_0$. This is an entrance boundary for all components with $\alpha \ge 0$. Therefore, we have to impose Dirichlet boundary conditions for the ray function there, as we do at the entrance. For the principal components with $\alpha < 0$, this boundary is obviously an exit. Since the boundary conditions are already imposed, upon reaching this boundary these components do not leave the ray domain. Instead, they turn into erroneous reflected rays. To avoid their influence, we choose the parameters of the algorithm so that these rays will not enter Ω as defined in (1.3). This can be achieved by moving ray side boundaries away from Ω. We thus define the ray domain $\hat{\Omega}^\ell$ by

$$\hat{\Omega}^\ell = \{(\xi^\ell, \eta^\ell) : \ -d^\ell/2 - \Delta d_\xi \le \xi^\ell \le d^\ell/2, \ \ -d^\ell/2 - \Delta d_\eta \le \eta^\ell \le d^\ell/2 + \Delta d_\eta\},$$

where $d^\ell = c^\ell d$ and c^ℓ is chosen so that the ray domain, even for $\Delta d_\eta = 0$, completely covers the wave domain ($1 \le c^\ell \le \sqrt{2}$, depending on the lattice point). Here Δd_η is how far the ray domains should be extended in the η direction, and Δd_ξ is how far they should be extended in the negative ξ direction (as discussed in the next section). On the finest ray level, with $L = 8$ and the maximum inclination angle $\alpha = \pi/4$, if Δd_η satisfies

$$\Delta d_\eta = \tan\left(\frac{\pi}{4}\right)\frac{d}{2} = \frac{d}{2}, \tag{3.1}$$

then no reflected rays enter the target domain Ω.

If the algorithm employs more than one ray grid, the coarser ray domains must meet two requirements. First, similar to the finest ray grids, they should be large enough not to allow the erroneous reflected rays into the target domain. Since the maximum inclination angle decreases as the algorithm proceeds to the coarser grids and the finer lattices, extension (3.1) is sufficient for all grids. More important, the coarser ray domains should cover neighboring finer ray domains, i.e., they should be of the size of the finer ray domains or larger (if the propagation directions are not aligned). The latter effect, however, diminishes as the difference between the propagation directions of neighboring principal components becomes smaller on the

finer lattices. Moreover, since the involved grids are coarser, this effect is negligible in terms of the overall computational work.

Let us notice that since the algorithm allows us to avoid the effect of reflected rays in the computational domain Ω, there is no need to impose artificial absorbing boundary conditions, such as perfect matching layers (PML; see, e.g., [5], [6], [1], [16]), which are actually used for the purpose of avoiding reflection.

3.2. Entrance: Backward residual influence. In the nearest neighborhood of the entrance boundary, on each ray grid we assume the solution to be purely of the ray form; i.e., it is governed only by the RBC. Ideally, there should be no impact of the interior solution on the upstream values. This would be true if we used a strictly upstream discretization for the ray equations. To provide an accurate ray discretization we have, however, chosen the discretization (2.5), which employs one downstream grid point. This implies upstream spread of changes in the solution from the region with nonzero residuals in the RHS of (1.7). The question then arises: how far back to place the entrance boundary from the region where residuals from the finer grids are defined, so that the entrance RBC is not polluted by the solution behavior in the interior?

An easy way to see the upstream impact of residuals is to analyze the changes in neighboring-line residuals introduced upon relaxing equations on one line, e.g., the line $\xi = \tilde{\xi}$. Such equations are relaxed using line Gauss–Seidel relaxation to change the solution values on the line $\xi = \tilde{\xi} + h_\xi/2$. Assuming that a Fourier component $r(\eta) = R_0 e^{i\omega\eta}$ of residuals exists on the line $\tilde{\xi}$, one line Gauss–Seidel relaxation sweep will change the solution on the line $\tilde{\xi} + h_\xi/2$ by $V_0 e^{i\omega\eta}$, where by (2.5)

$$\left(\frac{2ik}{h_\xi} - \frac{1}{2h_\xi^2} + \frac{\cos(\omega h_\eta) - 1}{h_\eta^2}\right) V_0 = -R_0. \tag{3.2}$$

Such a change will add $R_{-1}e^{i\omega\eta}, R_1 e^{i\omega\eta}$, and $R_2 e^{i\omega\eta}$ to the residuals on the lines $\xi = \tilde{\xi} - h_\xi$, $\xi = \tilde{\xi} + h_\xi$, and $\xi = \tilde{\xi} + 2h_\xi$, respectively, where by (2.5)

$$R_{-1} = R_2 = \frac{1}{2h_\xi^2} V_0 \tag{3.3}$$

and

$$R_1 = \left(-\frac{2ik}{h_\xi} - \frac{1}{2h_\xi^2} + \frac{\cos \omega h_\eta - 1}{h_\eta^2}\right) V_0. \tag{3.4}$$

Hence, and by (1.10),

$$\frac{|R_{-1}|}{|R_0|} < \frac{1}{4kh_\xi} = \frac{4}{L^2}. \tag{3.5}$$

Since the smallest value of L in our algorithm is 8, the ratio always satisfies

$$\frac{|R_{-1}|}{|R_0|} \le \frac{1}{16}. \tag{3.6}$$

If m_ξ relaxation sweeps are performed, the backward influence of the residual will continue to propagate and will reach the line $\xi = \tilde{\xi} - m_\xi h_\xi$. The size of the residual there, however, will not exceed $R_0 L^{-2m_\xi}$. Thus, at the distance of only few mesh-sizes the backward influence of the interior residuals becomes really negligible. It also

follows from (3.5) that on coarser ray grids the backward influence per unit length remains the same.

In our solver, for technical reasons, near the boundaries the solution is always defined by the ray formulation with the homogeneous RHS. Therefore, there is no need to extend the computational domain in the negative propagation direction.

3.3. Interpolation of boundary conditions. The subject of this section is how to diminish the influence of errors in boundary interpolation. Unlike many other boundary conditions, the RBC are defined on the coarsest (ray) grid. From there the boundary values are gradually interpolated into the finer ray grid, then to the intermediate wave grids, and, eventually, to the target wave grid.

Unlike the interior values, the boundary values are never changed (improved) on the finer grids, and therefore the errors that arise in the boundary interpolations are never corrected there. Therefore, to guarantee accurate solutions, it is necessary to be able to obtain for all [illegible] (those participating in the RBC) as small interpolation errors as needed. We show below that this is always possible by employing high enough interpolation orders along the boundaries. (No such high orders are needed in the interior of the fine grid.) Furthermore, we show that the effect of these boundary errors on the solution [illegible] the domain remains equally small.

3.3.1. Interpolation from a ray grid. In the coming discussion we need to distinguish between entrance and side boundaries.

Let us first consider a Fourier component of the error originated from interpolating a principal component on the [illegible], $\xi = \xi_0$, and thus having the form

$$v(\xi, \eta) = V(\xi) e^{i\omega_\eta \eta}. \tag{3.7}$$

Substitution of (3.7) into (1.7) leads to an ODE for $V(\xi)$:

$$V''(\xi) + 2ikV'(\xi) - \omega_\eta^2 V(\xi) = 0. \tag{3.8}$$

All principal components that should be approximated on a given ray grid have the frequencies that satisfy

$$(w_\xi, w_\eta) = k(\cos\theta - 1, \sin\theta), \quad -\pi/L \le \theta \le \pi/L. \tag{3.9}$$

This, in particular, means that $|\omega_\eta|$ is always less or equal to k, and therefore the error amplitude

$$V(\xi) = V(\xi_0) e^{-i(k \pm \sqrt{k^2 - \omega_\eta^2})\xi} \tag{3.10}$$

is oscillatory and neither decreases nor increases as the error propagates from the boundary into the domain. The initial amplitude $V(\xi_0)$ can be made arbitrarily small, because by (3.8) and (1.10), the ray functions represented on each grid are relatively smooth in the η direction:

$$|w_\eta h_\eta| \le 1, \tag{3.11}$$

where h_η is the mesh-size in the η direction of the coarse grid. Such functions can be interpolated from the coarse ray grids to the finer grids very accurately just by using an interpolation of a high enough order.

Let us next consider a [illegible], for definiteness the boundary $\eta = -\eta_0$. Substituting an interpolation error component of the form

$$\hat{v}(\xi, \eta) = \hat{V}(\eta) e^{i\omega_\xi \xi} \tag{3.12}$$

into the ray equation (1.7), we obtain an ODE for $V(\eta)$:

$$\hat{V}'' - (\omega_\xi^2 + 2k\omega_\xi)\hat{V} = 0. \tag{3.13}$$

By (3.9), $-2k < \omega_\xi < 0$, and therefore $\omega_\xi^2 + 2k\omega_\xi < 0$; any solution of (3.13) is again oscillatory, with an amplitude that neither decreases nor increases as it propagates into the computational domain. Fortunately, it too can be reduced as much as one wishes by increasing the interpolation order. This is because, by (1.10) and (3.9), for the highest principal frequency that still needs to be accurately represented on the fine ray grid there holds

$$|\omega_\xi| h_\xi < 0.4. \tag{3.14}$$

Comparison between (3.11) and (3.14) shows that the order of interpolation operators used along the side boundaries can be lower than that needed along the entrance.

3.3.2. Interpolation from a wave grid. We consider here two wave grids: a fine one, with a mesh-size h in both directions, and a coarse, with the mesh-size $2h$. We examine the propagation of a component of an interpolation error, resulting from interpolating a principal component $e^{i(k_1 x + k_2 y)}$, $k_1^2 + k_2^2 = k^2$, in the form

$$v(x, y) = V(y) e^{ik_1 x}, \tag{3.15}$$

introduced, for example, on the boundary $y = y_0 = -d/2$. Its amplitude, $V(y)$, satisfies the equation

$$V''(y) + (k^2 - k_1^2)V(y) = 0, \tag{3.16}$$

and hence

$$V(y) = V(y_0) e^{\pm ik_2^2 y}, \tag{3.17}$$

so $|V(y)| \le |V(y_0)|$. Using interpolation operators of a sufficiently high order, the initial amplitude $V(y_0)$ for such values of k_2 can be made as small as desired, since $|k_2|2h \le 2hk \le 1$; this is because for any wave grid to which the boundary conditions are interpolated, its mesh-size satisfies $kh \le 0.5$. These are grids finer than the reconstruction level scale h_b, where $kh_b \approx 1$; no boundary conditions are interpolated to the coarser levels.

All these results show that the boundary interpolation errors do not decay as they propagate into the computational domains; they can be diminished only by achieving small initial (on the boundaries) error amplitudes. To guarantee that, for the components of interest, the algorithm should employ interpolations of high orders (possibly higher than the typical, for standard multigrid solvers, second or fourth orders). Fortunately such high-order, and therefore expensive, interpolation procedures are needed only for interpolation to the ray entrance boundaries and to the boundaries of the wave grids with $kh \approx 1$ and $kh \approx 0.5$. (For the finer wave grids the principal components are smooth and can be accurately approximated by regular interpolation procedures.) Since this interpolation work is needed only on the boundaries and only on the coarse grid domains, its total cost is insignificant.

The results in Table 3.1 show how the increase of the interpolation order decreases the interpolation error.

TABLE 3.1
Error amplitudes obtained in interpolation of the Fourier component exp(iωx) from the grid with ωh ≈ 1 (the most oscillatory component that appears in the boundary interpolation) in the x direction. The parameter of the table is the interpolation order.

Order	2	4	6	8
Error	1.2e-01	2.1e-02	4.1e-03	8.4e-04

4. Numerical experiments. In all numerical experiments the wave-ray algorithm was applied to the model problem (1.1) considered on the computational domain

$$\Omega = \{(x, y) : -d/2 \le x, y \le d/2\} \tag{4.1}$$

and with the RHS support

$$\Omega_f = \{(x, y) | x^2 + y^2 \le 2^2\}. \tag{4.2}$$

In all experiments with the nonhomogeneous RHS, $f(x, y)$ is chosen to satisfy

$$f(x, y) = L\underline{u}(x, y), \tag{4.3}$$

where $\underline{u}(x, y)$ is a polynomial of $r = \sqrt{x^2 + y^2}$ for $|r| \le 2$, a Hankel function of r for $|r| > 2$, and in $\mathbf{C}^4(\mathbf{R^2})$. The RHS $f(x, y)$ is therefore a polynomial for $|r| \le 2$ and zero elsewhere. In most experiments we assume $k = 1$.

The size d of the target domain Ω varies for different tables. The algorithm employs only one ray level, with eight lattice components ($n = 1, L = L^1 = 8$). The RBC are either homogeneous, or inhomogeneous with just one principal component. This component is imposed on one ray grid that corresponds to the lattice principal component $(\cos(0), \sin(0))$. The algorithm employs (except in Tables 4.11–4.13 below) cubic interpolation. The number of extended mesh-sizes in the ξ direction for ray domains is usually $m_\xi = \Delta d_\xi / h_\xi = 2$, and in the η direction it is $m_\eta = \Delta d_\eta / h_\eta = 10$. The values of m_ξ and m_η become experimental parameters in Tables 4.5–4.7 below.

The errors, calculated in the L_2 norm, measure the difference between the discrete solution $u^h_{i,j}$ and the differential one $\underline{u}(x_i, y_j)$. In almost all experiments (unless stated otherwise) the errors are calculated in the domain

$$\Omega_e = \{(x, y) : \sqrt{x^2 + y^2} \le 15\}. \tag{4.4}$$

Note that Ω_e is located at the center of Ω. Therefore, the wave and the ray phase errors arising from the RHS do not depend on d. On the other hand, the phase errors caused by the nonhomogeneous RBC depend on d, since d defines the distance that the principal component travels before reaching Ω_e.

In all experiments, the error is measured after five wave-ray cycles; after that the error remains stable. The algorithm, however, continues to reduce the residual. (For numerical experiments on the residual convergence, see [9].)

The goal of our numerical experiments is to show the influence of different error terms and to confirm our theoretical conclusions.

4.1. Wave truncation errors. Since the RHS in Table 4.1 is not homogeneous, the error includes the wave truncation (nonphase) and the wave phase errors. For this model problem (with homogeneous RBC) and for the error measured in Ω_e, none of these errors depend on d, and both are proportional to h^2. The errors for $h = 0.25$ and $h = 0.5$ indeed exhibit these properties, and therefore we conclude that there

TABLE 4.1
Errors for different finest mesh-sizes h and computational domain sizes d; the results shown are for the model problems with the RHS defined by (4.3). *The RBC are homogeneous.*

	$d = 32$	$d = 64$	$d = 128$
$h = 0.125$	3.15e-03	3.26e-03	2.97e-03
$h = 0.25$	4.36e-03	5.73e-03	5.37e-03
$h = 0.5$	1.98e-02	2.25e-02	2.26e-02

TABLE 4.2
Errors for different finest mesh-sizes h and computational domain sizes d; the results shown are for the model problems with the homogeneous RHS. The RBC are nonhomogeneous for the lattice component $(\cos(0), \sin(0))$.

	$d = 32$	$d = 64$	$d = 128$
$h = 0.125$	1.10e-02	1.94e-02	3.90e-02
$h = 0.25$	4.08e-02	7.95e-02	1.61e-01
$h = 0.5$	1.71e-01	3.30e-01	3.90e-01

TABLE 4.3
Errors for different finest mesh-sizes h and computational domain sizes d. The RHS is homogeneous. The RBC are nonhomogeneous for the nonlattice principal component $(\cos(\pi/8), \sin(\pi/8))$.

	$d = 32$	$d = 64$	$d = 128$
$h = 0.125$	5.51e-02	5.88e-02	7.41e-02
$h = 0.25$	7.61e-02	1.07e-01	1.73e-01
$h = 0.5$	1.70e-01	3.07e-01	5.71e-01

the wave errors are dominant. This, however, changes for $h = 0.125$, where the error decrease with h weakens, meaning that the resulting error is a combination of other errors, such as ray phase errors (the nonpolynomial part of the solution is of the ray nature, and consists of a variety of ray components) and interpolation errors. In the next experiments we will investigate the behavior of such errors and the ways to diminish them.

4.2. Phase errors. In Table 4.2, there are neither RHS-related errors (the RHS is homogeneous), nor RBC phase ray errors, since the solution ray functions are constant (because the RBC are defined at a lattice component). The errors depend linearly on the domain size d and quadratically on the target mesh-size h. These are the properties of the [illegible] (2.3), which is the main error source in these experiments.

In the experiments presented in Table 4.3, the RBC conditions are defined for a nonlattice principal component, and this introduces the RBC ray phase errors. Unlike the wave phase errors, they should not depend on the target mesh-size h but still depend on d.

The results show that the errors depend almost linearly on d, confirming their phase nature. Their dependence on h, however, is mixed: the error is a combination of the $O(h^2)$ wave and the $O(1)$ ray phase errors (by (2.3) and (2.10), respectively).

In Table 4.4 we compare the errors obtained for different values of the wave number, k. The range of k is chosen to allow the use of the same [illegible] wave and ray grids for all values of k. (For any pair of values $k = k_0$ and $k = 2k_0$, different grids should be chosen for separation ($kh \approx 1$) and for the ray representation (1.10). Therefore, in our experiments we consider k such that $k_0 = 0.8 \le k < 1.6 = 2k_0$.)

TABLE 4.4

Errors for different values of k; the RHS is homogeneous, the RBC are nonhomogeneous for the principal component $(\cos(\alpha), \sin(\alpha))$, $h = 0.125$, $d = 64$.

k	0.8	0.9	1.0	1.1	1.2	1.3	1.4	1.5
$\alpha = 0$	1.02e-02	1.43e-02	1.94e-02	2.60e-02	3.47e-02	4.66e-02	6.36e-02	8.73e-02
$\alpha = \pi/16$	1.30e-02	1.84e-02	2.27e-02	2.96e-02	3.78e-02	4.80e-02	6.50e-02	8.90e-02

TABLE 4.5

Errors for different values of $m_\eta = \Delta d_\eta / h_\eta$, the number of additional mesh-sizes in the positive (and negative) η direction for ray domains. The model problem is with the nonhomogeneous RHS defined by (4.3); *the RBC are homogeneous, $h = 0.125$.*

m_η	0	2	5	10
$d = 64$	3.34e-03	3.30e-03	3.25e-03	3.25e-03
$d = 32$	3.91e-03	3.26e-03	3.15e-03	3.15e-03

TABLE 4.6

Errors for different values of $m_\eta = \Delta d_\eta / h_\eta$, the number of additional mesh-sizes in the positive (and negative) η direction for ray domains. The model problem is with the homogeneous RHS; the RBC is nonhomogeneous for the principal component $(k_1, k_2) = (\cos(\alpha), \sin(\alpha))$, $d = 64$, $h = 0.125$.

m_η	0	2	5	10	20	40
$\alpha = \pi/16$	1.59e-01	1.38e-01	3.96e-02	2.27e-02	2.09e-02	2.09e-02
$\alpha = \pi/8$	2.28e-01	1.86e-01	6.99e-02	5.88e-02	5.73e-02	5.73e-02

TABLE 4.7

Errors for different values of $m_\xi = \Delta d_\xi / h_\xi$, the number of additional mesh-sizes in the negative ξ direction for ray domains. The model problem is with the nonhomogeneous RHS defined by (4.3); *the RBC are homogeneous, $d = 64$, $h = 0.125$.*

m_ξ	0	5	10
error	3.27e-03	3.25e-03	3.25e-03

For $\alpha = 0$, the error ratios are proportional to the cubic power of the wave number ratios, which is a characteristic of the wave phase error. For $\alpha = \pi/16$, instead of cubic dependence, the power is $2 + \epsilon$, $0 < \epsilon < 1$, reflecting the fact the error is a combination of the $O(k^3)$ wave and the $O(k)$ ray phase errors (by (2.3) and (2.10)).

4.3. Reflected rays. The results in Table 4.5 show almost no difference for different values of m_η for $d = 64$, and this is because of the character of the solution near the boundary: the solution decays as it propagates toward boundaries, and therefore the size of the reflected waves is small and does not affect the total error. The errors for $d = 32$ are slightly higher for small m_η, and this is because the size of the solution near the boundaries, and, consequently, the size of the reflected waves is higher. This, even more profoundly, can be seen when the solution amplitudes remain large near the boundary, as in Table 4.6.

Table 4.6 shows the influence of Δd_η (section 3.1) on the accuracy of the solution. The result confirm that in order to avoid influence of the reflected rays, the side boundaries of ray domains should be sufficiently extended.

4.4. Backward residual propagation. As discussed at the end of section 3.2, the results in Table 4.7 show no influence of the number of extra mesh-sizes in the direction opposite to the propagation direction ξ on the solution accuracy.

TABLE 4.8

Errors for the solver that employs ray discrete operators of second order (2.5); *the solver employs cubic interpolation, both in the interior and on the boundaries; the RHS is homogeneous, and the RBC are nonhomogeneous for the principal component* $(k_1, k_2) = (\cos(\alpha), \sin(\alpha))$.

h	d	$\alpha = 0.0$	$\alpha = \pi/16$	$\alpha = \pi/12$	$\alpha = \pi/8$
0.125	64	1.94e-02	2.27e-02	2.71e-02	5.88e-02
0.25	64	7.95e-02	7.84e-02	8.10e-02	1.07e-01
0.125	128	3.90e-02	4.00e-02	4.39e-02	7.41e-02
0.25	128	1.61e-01	1.55e-01	1.55e-01	1.73e-01

TABLE 4.9

Errors for the solver that employs ray discrete operators of fourth order; the solver employs cubic interpolation, both in the interior and on the boundaries; the RHS is homogeneous, and the RBC are nonhomogeneous for the principal component $(k_1, k_2) = (\cos(\alpha), \sin(\alpha))$.

h	d	$\alpha = \pi/16$	$\alpha = \pi/12$	$\alpha = \pi/8$
0.125	64	2.02e-02	2.02e-02	2.75e-02
0.25	64	7.79e-02	7.57e-02	7.74e-02
0.125	128	3.87e-02	3.79e-02	4.36e-02
0.25	128	1.54e-01	1.49e-01	1.43e-01

4.5. Ray phase errors. There are two ways to decrease ray phase errors. First, one can employ a high-order discretization for the ray equation (1.7). In the next two tables the accuracy of the solution obtained by the solver that employs the second- and the fourth-order discretization schemes for the ray equations are presented and compared.

Comparing the results in Tables 4.8 and 4.9 shows that the higher-order discretization does not much improve the accuracy for small α, i.e., $\alpha = \pi/16$, the expected results since very small discretization error is committed for nearly lattice component. The dominant error in this case, even for the second-order discretization scheme, is the wave phase error. For larger α, the improvement is significant (especially for the most oscillatory $\alpha = \pi/8$, the middle point between two lattice components): the dominant part of the remaining error is clearly of a wave nature; it depends almost quadratically on h and linearly on d. Such improvement is observed since the higher-order discretization diminishes all discretization errors, including the phase error—the main error source for larger α in Table 4.8.

To reduce the ray phase errors [illegible] increasing discretization orders, one needs to employ not one, as in the experiments discussed above, but several ray grids, with increasingly finer lattice discretization on the coarser grids. The finer the lattice discretization is, the smaller the ray phase errors (2.10) are. The goal of the intermediate (finer) ray grids is to reduce errors to the almost pure rays. The overall accuracy of the ray representation is then defined by the coarsest ray grid with the largest number of lattice components, $L = 2^{N+2}$, where N is the number of ray levels employed. There the radiation boundary conditions are imposed, facilitated by the nearly pure ray representation at that grid. With the right choice of N and, correspondingly, L, one can make the ray phase errors arbitrary small.

This approach is perfectly suitable for the problems considered on very large computational domains: it requires significant coarsening in space, as L grows (1.10), which makes the ray approximation very inexpensive even for large L and d.

Large L, however, cannot be used for the problems with not sufficiently large kd. This is because to allow effective separation and representation of the ray residuals, each ray grid should consist of at least a few points in each direction. Since the mesh-

TABLE 4.10

Errors for solutions of discrete ray equations with the choice of h_ξ, h_η, and the RBC for principal components that depends on the choice of $L = 8, 16$. The values of α in the top line are given for $L = 8$; the values of alpha for the same principal components for $L = 16$ is given in parentheses (for instance, $\alpha = \pi/8$ becomes a lattice component, i.e., its new value for $L = 16$ is $\alpha = 0$); the RHS is homogeneous, and the RBC are nonhomogeneous for the principal component $(k_1, k_2) = (\cos(\alpha), \sin(\alpha))$. The error is calculated in Ω_E.

L	(h_ξ, h_η)	d	0(0)	$\pi/32(\pi/32)$	$\pi/16(\pi/16)$	$\pi/12(\pi/24)$	$\pi/8(0)$
8	(4,2)	256	0.00	2.01e-04	3.20e-02	1.06e-01	5.50e-01
16	(16,4)	256	0.00	8.27e-03	1.42e-01	2.66e-02	0.00
16	(16,4)	512	0.00	1.64e-02	2.80e-01	5.20e-02	0.00

size in the ξ direction satisfies (for $L = 16$, for example) $kh_\xi \approx 16$, it is impossible to use larger L for $kd = 32, 64$, and technically tricky (employing more sophisticated separation) for $kd = 128$. Therefore, in all the numerical experiments presented here, the algorithm employs eight ray components. A higher number of ray functions, e.g., $L = 16, 32$, etc., should be used for values of kd larger than the ones presented here. For larger values of L the same kind of errors arise and same measures are taken as for $L = 8$.

In the next set of experiments we compare different discretization (with different mesh-sizes and different ranges of principal components) of the ray equation (2.5). The results allow us to compare the errors for different L. At each experiment, we employ only one ray grid (analogue of the coarsest ray grid for one lattice component) and solve there the discrete ray problem (1.7) with the homogeneous RHS and the nonhomogeneous boundary conditions for one principal component. (The choice of mesh-sizes and the range of components depends on L.) Such experiments are relevant to our discussion, since the obtained solutions (and their accuracy) would be barely changed on the finer (if any) ray grids (the ray solution is not even relaxed on the finer ray grids, after visiting the coarser ones) and on the wave grids as previously discussed. Indeed some error, such as from interpolation, is introduced on the way to the target grid, but this has already been analyzed and calculated. For Table 4.10 only the error is calculated in the larger (than Ω_e) domain Ω_E:

$$\Omega_E = \{(x, y) : \sqrt{x^2 + y^2} \le 50\}. \tag{4.5}$$

Clearly, the results in Table 4.10 are in agreement with (2.10): the maximal error (and this error is indeed of a phase nature, as it depends linearly on d) for $L = 8$ is observed for $\alpha = \pi/8$, the middle point between two lattice components for $L = 8$, and it is four times larger than the maximal error for $L = 16$ observed for $\alpha = \pi/16$, the middle point between two lattice components for $L = 16$.

4.6. Interpolation of boundary conditions. In Tables 4.11–4.13 the influence of the interpolation orders is tested. The difference between the tables is in the choice of the inclination angle α of the principal component for which the nonhomogeneous RBC are imposed. The influence of the interpolation orders on the accuracy of the solution is tested for different values of h and d, as well. Fourth-order discretization is employed for the ray operators in order to reduce phase ray errors, in this way making the influence of interpolation errors visible.

We start with the RBC nonhomogeneous for the lattice principal component with $\alpha = 0$. In this case the ray functions are constant, and therefore the accuracy of the ray interpolation does not matter, since interpolation of any order accurately interpolates

TABLE 4.11

Errors for interpolation operators of different orders. Here $(p_w(p_w^b), p_r(p_r^b))$ *are the order of wave and ray interpolation operators used in the algorithm, both in the interior and on the boundaries; the numbers in parentheses are the orders of the operators used to interpolate to the boundaries from the wave and the ray coarser grids, respectively, if they are different from the orders used for the interior interpolation; the RHS is homogeneous, and the RBC are nonhomogeneous for the principal component* $(\cos(0), \sin(0))$.

h	d	$(4,4)$	$(4,2)$	$(2,2)$	$(4(6),4)$	$(2(4),2)$
0.125	64	1.94e-02	1936e-02	6.20e-02	1.94e-02	6.20e-02
0.25	64	7.95e-02	7.96e-02	1.00e-01	7.94e-02	1.00e-02
0.25	128	1.61e-01	1.61e-01	1.71e-01	1.61e-01	1.71e-01

TABLE 4.12

Like Table 4.11*, except that the RBC are nonhomogeneous for the principal component* $(\cos(\pi/16), \sin(\pi/16))$.

h	d	$(4,4)$	$(4,2)$	$(2,2)$	$(4(6),4)$	$(2(6),2(6))$
0.125	64	2.27e-02	2.56e-02	6.81e-02	2.23e-02	6.80e-02
0.25	64	7.84e-02	7.90e-02	1.02e-01	7.90e-02	9.98e-02
0.25	128	1.55e-01	1.55e-01	1.68e-01	1.55e-01	1.68e-01

TABLE 4.13

Like Table 4.11*, except that the RBC are nonhomogeneous for the principal component* $(\cos(\pi/8), \sin(\pi/8))$.

h	d	$(4,4)$	$(2,2)$	$(4(6),4(6))$
0.125	64	5.88e-02	1.13e-01	5.84e-02
0.25	64	1.07e-01	1.41e-01	1.06e-01
0.5	128	1.73e-01	1.73e-01	1.73e-01

constant functions. In the wave representation, however, this e^{ix} component is the most oscillatory principal component that yields the highest truncation (phase) and boundary interpolation errors.

The results in Table 4.11 show that as long as the cubic interpolation ($p = 4$) is employed in the wave interpolation, the errors depend quadratically on h and linearly on d, meaning that the wave phase error is the main error source. The influence of the interpolation errors is insignificant (for instance, increasing the interpolation order on the boundary barely improves the accuracy). This, however, changes for $p = 2$ in the wave interpolation, for which the interpolation errors become visible. The ray interpolation order does not influence the accuracy of the solution for this α.

Next we consider the principal component with $\alpha = \pi/16$. Linear interpolation $p = 2$ from the ray grids sufficient for the previous example ($\alpha = 0$) introduces a significant error here. For $\alpha = \pi/16$, $|\omega_\eta h_\eta| \approx 0.4$ (section 3.3.1), and therefore interpolation from the ray grid can be accurately performed using cubic interpolation ($p = 4$). For the wave representation, however, on the reconstruction level $|wh| \approx 1$ (section 3.3.2), and the higher-order interpolation ($p = 4$ or $p = 6$) from the wave grids (especially on the boundaries) leads to a better accuracy.

In Table 4.13 the nonhomogeneous RBC are imposed for the principal component $(\cos(\pi/8), \sin(\pi/8))$, the midpoint between the neighboring lattice components $(\cos(0), \sin(0))$ and $(\cos(\pi/4), \sin(\pi/4))$. For this component, both the ray principal components ($|\omega_\eta h_\eta| \approx 0.8$) and the wave principal components ($|\omega h| \approx 1$ at the reconstruction level) are rather oscillatory, and it seems that the algorithm could benefit

from increasing interpolation orders, at least for the boundary interpolation. This, however, is not reflected in the results shown in Table 4.13.

The errors in Table 4.13 are almost independent of the interpolation accuracy. They, however, depend on the size of the computational domain, meaning that the ray phase errors discussed in section 4.5 are the main source of inaccurate approximations.

5. Conclusions and future application. For most multigrid procedures the accuracy of the solution is defined by the discretization accuracy at the finest grid. This is not always so for problems on unbounded domains, where it is computationally beneficial to use progressively extended coarse grid domains, so that in its outer part each coarse grid provides not a correction to the fine grid solution, but the solution itself. This feature is present in the wave-ray algorithm for solving indefinite Helmholtz equations with radiation boundary conditions. The developed solver shows a textbook multigrid efficiency for highly indefinite problems on unbounded domains. With the right choice of parameters (such as computational domains, wave and ray mesh-sizes, order of interpolation operators) the produced solution has the accuracy defined by the finest grid discretization. The analysis offered here can be applied to other problems given on infinite domains with boundary conditions defined at infinity. When restricted to finite computational domains, the numerical solvers for such problems have to deal with interpolation and other errors, similar to the ones discussed in this paper.

The wave-ray algorithm discussed in this paper is rather technical and employs several special features: rotated coordinate systems, special weighting operators, etc. Some of these technicalities, however, could be simplified (for example, the use of the (x, y) coordinates for all ray components and the standard full weighting for the residual separation), and the obtained simplified algorithm would still be an efficient (though less efficient) solver for the Helmholtz equation. One of the restrictions that cannot be lifted in this [illegible] multigrid solver is the requirement to represent principal components in their analytical form, which is available only for sufficiently smooth k. These principal components are used to obtain differential equations for the ray functions, which are then discretized and solved.

Our next step (currently in progress; see preliminary results in [10]) is to use the wave-ray approach in the [illegible] multigrid framework, where only numerical approximation to the principal components are needed and used to obtain discrete ray equations (skipping the differential equations step). This approach allows application of the wave-ray algorithm to problems with nonsmooth wave numbers. Moreover, similar ideas can be used in multigrid algorithms developed for other differential operators if some approximations to their algebraically smooth error components are known or can be inexpensively discovered (e.g., [7]) and then used as a part of the algebraic multigrid solver.

Appendix. Pseudocode. A brief pseudocode of the wave-ray algorithm is given next. It describes the solver that was used in all experiments presented in this paper.

Begin wave ray cycle
Wave subcycle: from the finest to the coarsest wave grid
`Finest wave grid` $m = M_t$
`Relaxation` $L^m u^m = f^m$
`Residual calculation by full weighting` $r^{m-1} = W_m^{m-1}(f^m - L^m u^m)$
`Solution initialization by injection` $u^{m-1} = I_m^{m-1} u^m$

for $m = M-1, \dots, 2$
  `FAS right-hand side` $f^m = r^m + L^m u^m$
  if $(kh_m \not\approx 2)$, `Relaxation` $L^m u^m = f^m$
  `Residual calculation by full weighting` $r^{m-1} = W_m^{m-1}(r^m - L^m u^m)$
  `Solution initialization by injection` $u^{m-1} = I_m^{m-1} u^m$
`Coarsest wave grid` $(m = 1)$ `Relaxation` $L^m u^m = f^m$
Wave subcycle: from the coarsest to some fine wave grid
for $m = 2, \dots, M_f$, $M_f \le M$ is the wave grid with a small phase error
  `Interpolation` $u^m = u^m + I_{m-1}^m v^{m-1}$
  if $(kh_m \not\approx 2)$, `Relaxation` $L^m u^m = f^m$
`Wave residual calculation for ray subcycle` $(m = M_f)$ $r^m = f^m - L^m u^m$
for $(m = M_f - 1, \dots, M_c)$, M_c is the wave grid with $kh_c \approx 1$
  `Residual transfer by full weighting` $r^m = W_{m+1}^m r^{m+1}$
Ray subcycle with N ray levels
`Finest ray grid` $n = 1$
for $(\ell = 1, \dots, L_1)$
  `Residual separation` $\hat{r}^{1,\ell} = W_{M_c}^{1,\ell}(r^{M_c}\overline{e}^{1,\ell})$,
  $\overline{e}^{1,\ell}$ is the complex conjugate of $e^{1,\ell}$, the ℓth component on the 1st lattice
  `Ray FAS right-hand side`
    $\hat{f}^{1,\ell} = \hat{r}^{1,\ell} + \hat{L}^1 \hat{u}^{1,\ell}$ `where` $(\hat{r}^{1,\ell} \ne 0)$ `and` $\hat{f}^{1,\ell} = 0$ `elsewhere`
for $(n = 2, \dots, N)$
  `Residual separation`
  for $(\ell = 1, \dots, N, \ell = \ell + 2)$ `for odd` ℓ
    $\hat{r}^{n,\ell} = W_{n-1,(\ell+1)/2}^{n,\ell}(\hat{r}^{n-1,(\ell+1)/2})$
  for $(\ell = 2, \dots, N, \ell = \ell + 2)$ `for even` ℓ
    $\hat{r}^{n,\ell} = W_{n-1,\ell/2}^{n,\ell}(\hat{r}^{n-1,\ell/2} e^{n-1,\ell/2} \overline{e}^{n,\ell}) + W_{n-1,\ell/2+1}^{n,\ell}(\hat{r}^{n-1,\ell/2+1} e^{n-1,\ell/2+1} \overline{e}^{n,\ell})$,
    `where` $e^{n,\ell}$ `is the` ℓ`th component on the` n`th lattice`
  for $(\ell = 1, \dots, L_n)$
    `Ray FAS right-hand side`
    $\hat{f}^{n,\ell} = \hat{r}^{n,\ell} + \hat{L}^n \hat{u}^{n,\ell}$ `where` $(\hat{r}^{n,\ell} \ne 0)$ `and` $\hat{f}^{n,\ell} = 0$ `elsewhere`
    if $(n \ne N)$, `Relaxation` $L^n u^{n,\ell} = f^{n,\ell}$
    else
      `Imposition of RBC at entrance and side boundaries`
      $\hat{u}^{n,\ell}(\xi, \eta) = \hat{u}_0(\eta)$ `for` $\xi = \xi_0$; $\hat{u}^{n,\ell}(\xi, \eta) = \hat{u}_\pm(\xi)$ `for` $\eta = \pm\eta_0$
      `Solving` $\hat{L}^n \hat{u}^{n,\ell} = \hat{f}^{n,\ell}$
for $(n = N - 1, \dots, 1)$
  `Interpolation to the next finer ray grid`
  for $(\ell = 1, \dots, L_n)$
    `Interior interpolation:` $\hat{u}^{n,\ell} = \hat{u}^{n,\ell} + I_{n+1,2\ell-1}^{n,\ell} \hat{v}^{n+1,2\ell-1} +$
    $\frac{1}{2} I_{n+1,2\ell-2}^{n,\ell}(\hat{v}^{n+1,2\ell-2} e^{n+1,2\ell-2}/e^{n,\ell}) + \frac{1}{2} I_{n+1,2\ell}^{n,\ell}(\hat{v}^{n+1,2\ell} e^{n+1,2\ell}/e^{n,\ell})$
    `Boundary interpolation:` $\hat{u}^{n,\ell} = 0$ on $\hat{\Gamma}^{n,\ell}$
    $\hat{u}^{n,\ell} = I_{n+1,2\ell-1}^{n,\ell} \hat{u}^{n+1,2\ell-1} + \frac{1}{2} I_{n+1,2\ell-2}^{n,\ell}(\hat{u}^{n+1,2\ell-2} e^{n+1,2\ell-2}/e^{n,\ell}) +$
    $\frac{1}{2} I_{n+1,2\ell}^{n,\ell}(\hat{u}^{n+1,2\ell} e^{n+1,2\ell}/e^{n,\ell})$
`Wave solution reconstruction` $m = M_c$:
`Zero initialization at boundaries` $u^m = 0$ on Γ_m
for $(\ell = 1, \dots, L^1)$
  `In interior` $u^m = u^m + e^\ell I_{1,\ell}^m \hat{v}^{1,\ell}$; `At boundaries` $u^m = u^m + e^\ell I_{1,\ell}^m \hat{u}^{1,\ell}$
`End wave reconstruction`
for $(m = M_c + 1, \dots, M)$
  `Zero initialization at boundaries` $u^m = 0$ on Γ_m
  `In interior` $u^m = u^m + I_{m-1}^m v^{m-1}$; `At boundaries` $u^m = u^m + I_{m-1}^m u^{m-1}$
  `Relaxation` $L^m u^m = f^m$

Acknowledgment. The final part of the work on this paper was completed while the authors were visiting the Center for Applied Scientific Computing at Lawrence Livermore National Laboratory.

REFERENCES

[1] S. Abarbanel and D. Gottlieb, *A mathematical analysis of the PML method*, J. Comput. Phys., 134 (1997), pp. 357–363.

[2] A. Bayliss and E. Turkel, *Radiation boundary conditions for wave-like equations*, Comm. Pure Appl. Math, 33 (1980), pp. 707–725.

[3] A. Bayliss, C. I. Goldstein, and E. Turkel, *On accuracy conditions for the numerical computations of waves*, J. Comput. Phys., 59 (1985), pp. 396–404.

[4] I. M. Babuška and S. A. Sauter, *Is the pollution effect of the FEM avoidable for the Helmholtz equations considering high wave numbers?*, SIAM Rev., 42 (2000), pp. 451–484.

[5] J. P. Berenger, *A perfectly matched layer for the absorption of electromagnetic waves*, J. Comput. Phys., 114 (1994), pp. 185–200.

[6] J. P. Berenger, *Three-dimensional perfectly matched layer for the absorption of electromagnetic waves*, J. Comput. Phys., 127 (1996), pp. 363–379.

[7] M. Brezina, R. Falgout, S. MacLachlan, T. Manteuffel, S. McCormick, and J. Ruge, *Adaptive Smooth Aggregation (αSA)*, SIAM J. Sci. Comput., 25 (2004), pp. 1896–1920.

[8] A. Brandt, *Multigrid Techniques:* 1984 *Guide with Applications to Fluid Dynamics*, Monograph, GMD-Studie 85, GMD-FIT, Sankt Augustin, Germany, 1985.

[9] A. Brandt and I. Livshits, *Wave-ray multigrid method for standing wave equations*, Electron. Trans. Numer. Anal., 6 (1997), pp. 162–181.

[10] I. Livshits, *An algebraic multigrid wave-ray algorithm to solve eigenvalue problems for the Helmholtz operator*, Numer. Linear Algebra Appl., 11 (2004), pp. 229–239.

[11] A. Brandt and S. Ta'asan, *Multigrid methods for nearly singular and slightly indefinite problems*, in Multigrid Methods II, W. Hackbush and U. Trottenberg, eds., Lecture Notes in Math. 1228, Springer-Verlag, 1985, pp. 100–122.

[12] A. Deraemaker, I. Babuska, and P. Bouillard, *Dispersion and pollution of the FEM solution for the Helmholtz equation in one, two and three dimensions,* Int. J. Numer. Methods Engrg., 46 (1999), pp. 471–499.

[13] H. C. Elman, O. G. Ernst, and D. P. O'Leary, *A multigrid method enhanced by Krylov subspace iterations for discrete Helmholtz equations*, SIAM J. Sci. Comput., 23 (2001), pp. 1291–1315.

[14] F. Ihlenburg and I. Babuska, *Finite element solution to the Helmholtz equation with high wave numbers, Part* I: *The h-version of the FEM*, Comput. Math. Appl., 30 (1995), pp. 9–37.

[15] B. Lee, T. A. Manteuffel, S. F. McCormick, and J. Ruge, *First-order system least-squares for the Helmholtz equation*, SIAM J. Sci. Comput., 21 (2000), pp. 1927–1949.

[16] S. Tsynkov and E. Turkel, *A Cartesian perfectly matched layer for the Helmholtz equation*, in Absorbing Boundaries and Layers, Domain Decomposition Method Applications to Large Scale Computation, L. Tourette and L. Harpern, eds., Springer-Verlag, Berlin, 2001, pp. 279–309.

[17] P. Vanek, J. Mandel, and M. Brezina, *Two level algebraic multigrid for the Helmholtz problem*, Contemp. Math., 218 (1998), pp. 349–356.

SIAM J. SCI. COMPUT.
Vol. 28, No. 4, pp. 1252–1274
© 2006 Society for Industrial and Applied Mathematics

FAST COMPUTATION OF THE SLATER INTEGRALS*

L. GR. IXARU†, N. S. SCOTT‡, AND M. P. SCOTT§

Abstract. Slater integrals are two-dimensional radial integrals whose integrand is constructed from normalized eigenfunctions of the Schrödinger equation. These integrals occur in many atomic structure and scattering computations. However, in two-dimensional R-matrix propagation they represent a significant computational bottleneck. The problem involves two steps, that is, the numerical solution of the Schrödinger equation and the computation of the integrals, respectively. By exploiting the characteristic features of the problem we seek to devise a two-stage computational strategy, where the second stage is influenced and informed by the first. In particular, we focus on the development of extended frequency-dependent quadrature rules (EFDQRs) that improve the accuracy and significantly reduce the computation time of the integrals. The performance of these ad hoc rules is examined and compared to the currently used Newton–Cotes method in the construction of Hamiltonian matrices involving up to 300×10^6 integrals. A gain of two orders of magnitude in the CPU time is obtained.

Key words. computational atomic physics, CP methods, exponential fitting, frequency dependent weights, R-matrix, Slater integral

AMS subject classifications. 65D25, 65D32, 65L15, 81V45

DOI. 10.1137/050641004

1. Introduction.

Electron collisions with atoms and ions have been the subject of international interest for many years. Data from these processes are essential in the analysis of important physical phenomena in many scientific and technological areas [5]. Despite the importance of these applications, relatively little accurate cross section data are known for many of the processes involved. For example, no accurate cross section data, involving high lying excited states, have been calculated for electron collisions with the simplest atomic target, hydrogen. Such data is essential, for example, to resolve discrepancies in the modeling of early-type stars [33]. The theoretical treatment of electron collisions at these [illegible] is difficult because of the need accurately to allow for excitation to Rydberg states and for ionization of the target. Recent theoretical and computational advances pioneered in Australia [6], Europe [12], [4], [8], and the USA [2] have, however, provided a new context within which to attack this problem.

A particularly promising technique is the two-dimensional R-matrix propagator [13], a variant of the general R-matrix approach [7], that enables the accurate study of electron collisions with hydrogen-like (H-like) and quasi-H-like atoms and ions at intermediate energies. As illustrated in Figure 1, the two-electron configuration space (x_1, x_2) is divided into square sectors. The two-electron wavefunction describing the motion of the target electron and the colliding electron is expanded within each sector

*Received by the editors September 23, 2005; accepted for publication (in revised form) February 2, 2006; published electronically July 31, 2006. This work was supported by the UK Engineering and Physical Sciences Research Council under grants EP/C001753/1 and GR/R3118/01.
http://www.siam.org/journals/sisc/28-4/64100.html

†Institute of Physics and Nuclear Engineering, Magurele, Bucharest, R-76900 Romania (ixaru@theory.nipne.ro).

‡School of Electronics, Electrical Engineering and Computer Science, The Queen's University of Belfast, Belfast, BT7 1NN, UK (ns.scott@qub.ac.uk).

§School of Mathematics and Physics, The Queen's University of Belfast, Belfast, BT7 1NN, UK (m.p.scott@qub.ac.uk).

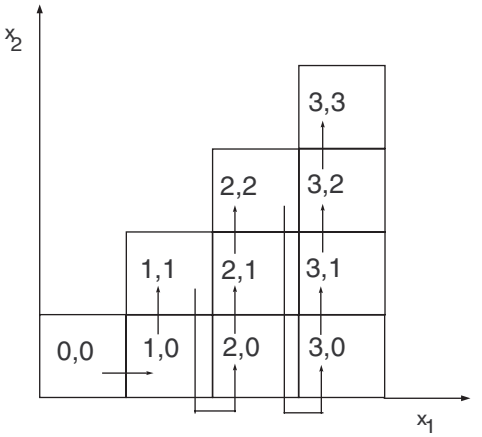


FIG. 1. *Subdivision of the configuration space* (x_1, x_2) *into a set of connected sectors. The arrows indicate the order in which the R-matrix propagation is performed.*

in terms of one-electron basis functions that are eigenfunctions of the Schrödinger equation, solved subject to certain fixed boundary conditions. The expansion coefficients are determined by diagonalizing the corresponding Hamiltonian matrix. The R-matrix may then be propagated across the sectors at each scattering energy and the scattering properties of interest determined.

An advantage of the technique is that the construction and diagonalization of each sector Hamiltonian matrix is independent. This renders the application suitable for exploiting both high performance computing (HPC) [34] and Grid architectures [10]. However, in large-scale simulations efficient exploitation is severely impeded by a significant load imbalance between the construction of the Hamiltonian matrix on diagonal and off-diagonal sectors. The root of the bottleneck is the large number of two-dimensional radial integrals, the so-called Slater integrals, that are required on each diagonal sector. Using standard "off the shelf" quadrature formula, Hamiltonian matrix construction on each diagonal sector, to approximately seven figures of accuracy, can take of the order of hours while each off-diagonal sector takes only tens of seconds.

We therefore pose the question, "Do the constituent components of the integrands have characteristics that can be exploited to enable the construction of a hand crafted quadrature formula that will both improve the accuracy and significantly reduce the computation time of the Slater integrals?" Answering this question is the focus of this paper. We begin in section 2 by specifying the problem; atomic units are used throughout the paper. In section 3 we investigate the solution of the Schrödinger equation and the computation of Slater integrals and outline a computational strategy that exploits the characteristic features of the problem. This strategy is further developed in section 4 and, in particular, in section 5 where we develop specialized sets of frequency-dependent quadrature rules that enable the Slater integrals to be computed to improved accuracy and at small computational cost. In section 6 we examine the efficiency of the quadrature rules when used to construct Hamiltonian matrices. Finally, in section 7 we present our concluding remarks.

2. The problem. Let $X_i = iH$, $i = 0, 1, \ldots, i_{max}$, be equidistant points on the half-axis $x \geq 0$. On each interval $I_i = (X_i, X_{i+1})$ we consider the eigenvalue problem

for the radial Schrödinger equation

$$y''_{nl} = (V(x) - E_{nl})y_{nl}(x), \quad x \in I_i, \tag{2.1}$$

in a Coulomb field,

$$V(x) = \frac{l(l+1)}{x^2} - \frac{2Z}{x}, \tag{2.2}$$

where l and Z represent the orbital angular momentum (a nonnegative integer) and the electric charge (a positive real number), respectively. The boundary conditions are as follows:

$$i = 0: \qquad y_{nl}(0) = 0, \qquad a_1 y_{nl}(X_1) + b_1 y'_{nl}(X_1) = 0,$$

$$i > 0: \ a_0 y_{nl}(X_i) + b_0 y'_{nl}(X_i) = 0, \ a_1 y_{nl}(X_{i+1}) + b_1 y'_{nl}(X_{i+1}) = 0,$$

where a_0, b_0, a_1, and b_1 are constants dictated by the physics of the problem but subject to the conditions $|a_0| + |b_0| \neq 0$ and $|a_1| + |b_1| \neq 0$. Throughout this paper we use $a_0 = a_1 = 0$ and $b_0 = b_1 = 1$.

The eigenfunctions, y_{nl}, normalized as

$$\int_{X_i}^{X_{i+1}} y_{nl}(x) y_{n'l}(x) dx = \delta_{nn'}, \tag{2.3}$$

are then used to compute two-dimensional radial integrals of the form

$$R = \int_{X_j}^{X_{j+1}} dx_2 \int_{X_i}^{X_{i+1}} dx_1 \, y_{n_3 l_3}(x_2) y_{n_4 l_4}(x_2) \Delta(x_1, x_2; \lambda) y_{n_1 l_1}(x_1) y_{n_2 l_2}(x_1), \tag{2.4}$$

where

$$\Delta(x_1, x_2; \lambda) := \begin{cases} \dfrac{x_1^\lambda}{x_2^{\lambda+1}} & \text{if } x_1 < x_2, \\ \dfrac{x_2^\lambda}{x_1^{\lambda+1}} & \text{if } x_1 \geq x_2. \end{cases}$$

Integrals of this form are sometimes called Slater integrals and have to be evaluated in each H by H square sector identified by the pair i, j, where $i = 0, 1, \ldots, i_{max}$ and $j = 0, \ldots, i$. An illustration for $i_{max} = 3$ is shown in Figure 1.

The following modest values give a lower limit for the computational effort involved in a realistic computation: $Z = 1$ (hydrogen atom), $H = 15$, $i_{max} \geq 10$, $1 \leq n \leq n_{max} = 20$, $0 \leq l \leq l_{max} = 9$, and $0 \leq \lambda \leq \lambda_{max} = 2l_{max} = 18$. Potentially, $n_{max}^4 \times (l_{max}+1)^4 \times (\lambda_{max}+1)$, i.e., 3.04×10^{10}, Slater integrals need to be computed in each sector. However, the invariance of R with respect to permutations $n_1 l_1 \rightleftarrows n_2 l_2$ or $n_3 l_3 \rightleftarrows n_4 l_4$, and the existence of physical selection rules based on correlations between λ and the orbital angular momenta l_1, l_2, l_3, l_4, reduce this number to a few hundred million integrals per sector. Nonetheless the computational effort remains substantial.

Not all Slater integrals exhibit the same computational complexity. Let us denote

$$J_1(a, z, \lambda, \phi) := \int_a^z dx \, x^\lambda \phi(x), \quad J_2(z, b, \lambda, \phi) := \int_z^b dx \, \frac{1}{x^{\lambda+1}} \phi(x). \tag{2.5}$$

If $i = j$, i.e., a diagonal sector of Figure 1, then

$$R = \int_{X_j}^{X_{j+1}} dz\, y_{n_3 l_3}(z) y_{n_4 l_4}(z) I(z, \lambda, \phi), \tag{2.6}$$

where

$$I(z, \lambda, \phi) = \left[\frac{1}{z^{\lambda+1}} J_1(X_j,\, z,\, \lambda, \phi) + z^{\lambda} J_2(z,\, X_{j+1},\, \lambda, \phi) \right], \tag{2.7}$$

and

$$\phi(x) = y_{n_1 l_1}(x) y_{n_2 l_2}(x).$$

This is a genuine double integral. However, if $i > j$, i.e., an off-diagonal sector, then R is simply a product of two one-dimensional integrals,

$$R = J_1(X_j,\, X_{j+1},\, \lambda, \phi_1) \times J_2(X_i,\, X_{i+1},\, \lambda, \phi_2), \tag{2.8}$$

where

$$\phi_1(x) = y_{n_3 l_3}(x) y_{n_4 l_4}(x), \quad \phi_2(x) = y_{n_1 l_1}(x) y_{n_2 l_2}(x).$$

It follows that the computation of an on-diagonal integral will take much longer than that of an off-diagonal integral and thus deserves special attention.

2.1. Graphical illustrations. To give a visual impression of the form of the Slater integral defined by (2.6) we present its constituent components on the diagonal sectors $I_0 \times I_0$ and $I_1 \times I_1$, denoted as 0, 0 and 1, 1, respectively, in Figure 1. We use the following typical, but randomly chosen, eigenfunctions $n_1 = 5$, $l_1 = 2$, $n_2 = 10$, $l_2 = 4$, $n_3 = 6$, $l_3 = 1$, $n_4 = 12$, $l_4 = 3$.

Figures 2 and 3 show the products of eigenfunctions, $\phi_1(x) = y_{n_1 l_1}(x) y_{n_2 l_2}(x)$ and $\phi_2(x) = y_{n_3 l_3}(x) y_{n_4 l_4}(x)$, that occur in the inner and outer integrals of (2.6), respectively. The behavior of (2.7), the inner integral $I(x, \lambda, \phi_1)$, for $\lambda = 2, 3, 4, 5, 6$ is displayed in Figures 4 and 5. Finally, Figures 6 and 7 show the complete Slater integrand again for $\lambda = 2, 3, 4, 5, 6$. These graphs illustrate the progressively complicated interference patterns among the constituent components of the double integral's integrand. Of particular note is the influence of λ on the amplitudes of the integrand.

3. Development of an ad-hoc computational strategy. The efficiency of the computation of the Slater integrals depends primarily on the number of points at which the values of the integrand are needed in order to achieve some predefined accuracy. For example, with a sector size of $H = 15$, the Newton–Cotes formulae typically require more than 1,000 mesh points to achieve about eight figures of accuracy. In this paper we target 10 figures of accuracy.

A significant reduction in the number of mesh points is possible but this requires an investigation of the two stages of the process, i.e., both the numerical solution of the Schrödinger equation and the actual computation of the Slater integrals. In fact, such an investigation should cover not only each separate stage but also the way in which the first is of assistance to the second. Our aim, therefore, is to design a sequence of numerical methods which advantageously exploits as many characteristic features of the problem as possible.

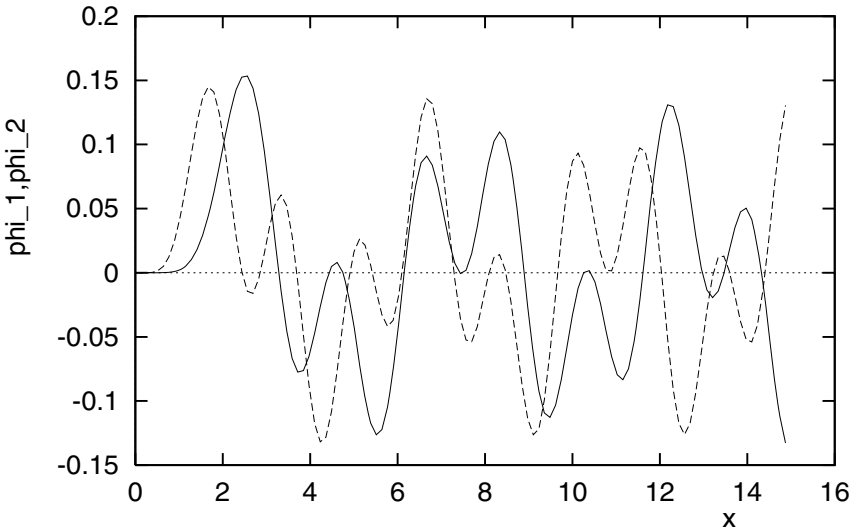


FIG. 2. *The functions* $\phi_1(x)$ (*solid lines*) *and* $\phi_2(x)$ (*broken lines*) *on interval* I_0.

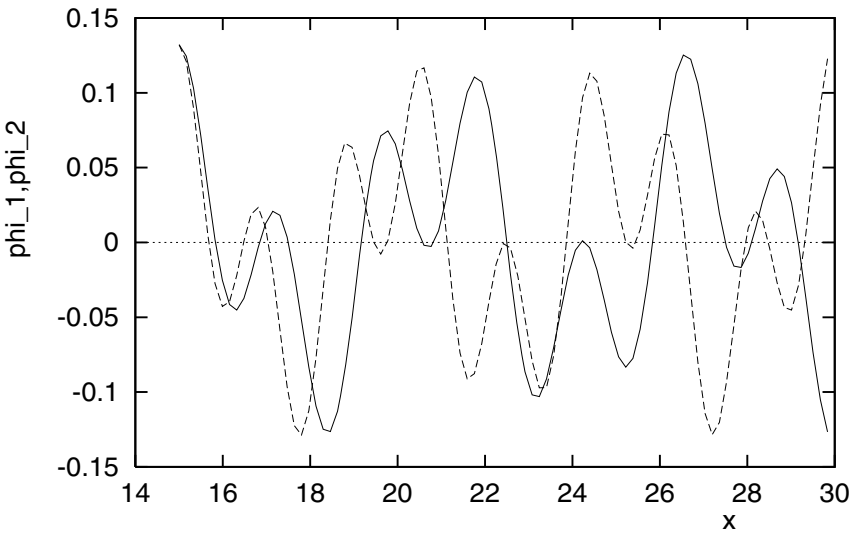


FIG. 3. *The functions* $\phi_1(x)$ (*solid lines*) *and* $\phi_2(x)$ (*broken lines*) *on interval on interval* I_1.

3.1. Solution of the Schrödinger equation. The potential function, $V(x)$, is singular at the origin, and therefore a tailored method is needed to compute a correct numerical solution in the vicinity of the origin. We specify the origin-enclosing interval by $(0,\, x_{cut}]$. To fix x_{cut} we have constructed and used the empirical formula

$$x_{cut} = \begin{cases} \min\{9(\exp(l)+1)/(\exp(l)+21),\, 3.75(l+0.5)/Z\} & \text{if } l \le 3, \\ 12(1+0.5/l)/Z & \text{if } l > 3. \end{cases}$$

For the solution on $(0,\, x_{cut}]$ we have opted for a piecewise perturbation version in which the centrifugal and Coulomb terms are taken as a reference and as a per-

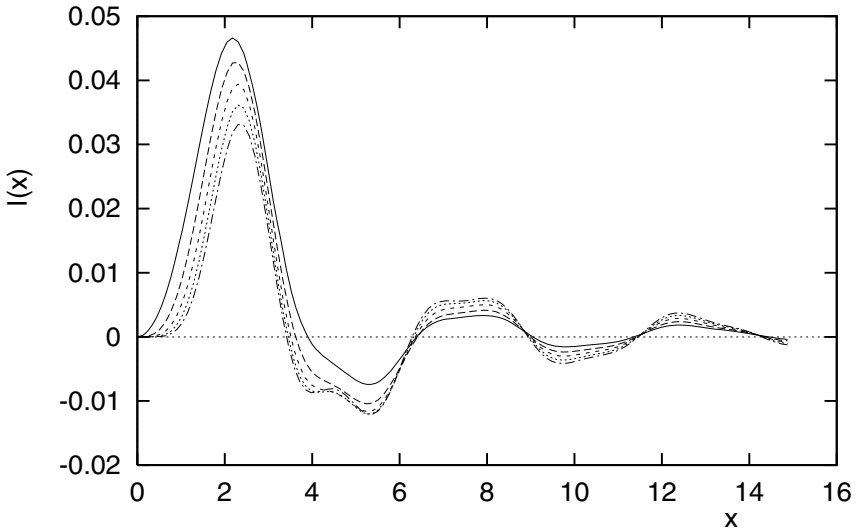


FIG. 4. *The inner integral* $I(x, \lambda, \phi_1)$ *on* I_0*:* $\lambda = 2$ (*solid lines*), $\lambda = 3$ (*long-dashed lines*), $\lambda = 4$ (*short-dashed lines*), $\lambda = 5$ (*dotted lines*), $\lambda = 6$ (*dashed-dotted lines*).

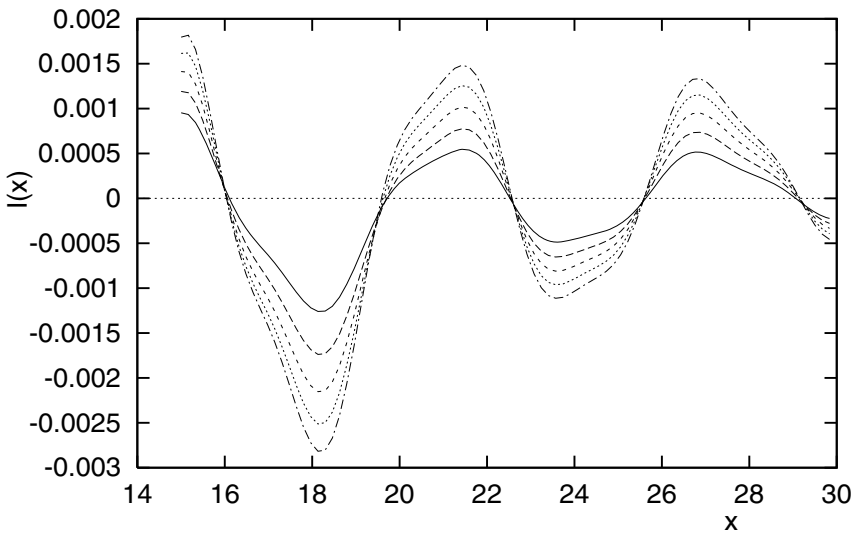


FIG. 5. *The inner integral* $I(x, \lambda, \phi_1)$ *on interval* I_1*:* $\lambda = 2$ (*solid lines*), $\lambda = 3$ (*long-dashed lines*), $\lambda = 4$ (*short-dashed lines*), $\lambda = 5$ (*dotted lines*), $\lambda = 6$ (*dashed-dotted lines*).

turbation potential, respectively. The procedure is described in [21], where another empirical formula for x_{cut} is described (see (3.28) of [21]). However, we have found the construction above to be slightly better in our context.

For $x > x_{cut}$ there are many options because, in principle, any numerical method for solving ODEs can be used; examples include multistep and Runge–Kutta methods. However, before making an informed choice we need to be cognizant of the following three conditions.

First, we note that eigenvalues and eigenfunctions for n up to 20 are required; higher values of n may also be needed in the future. Since the nth eigenfunction

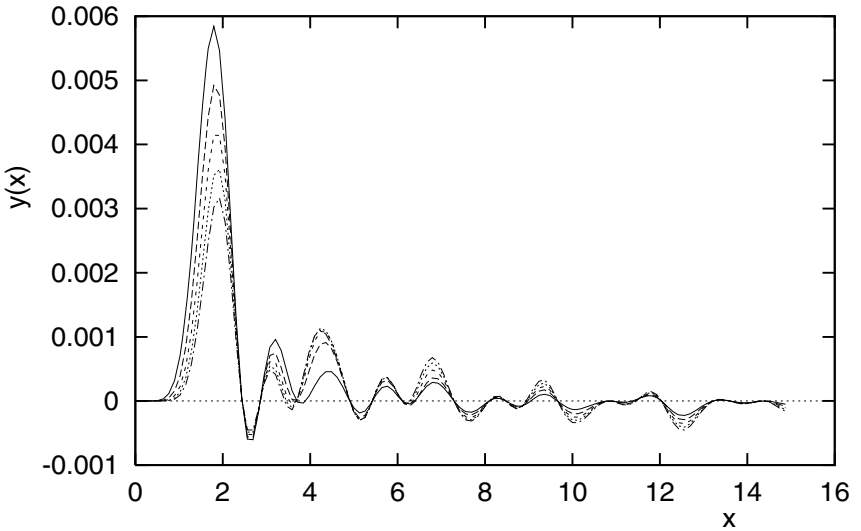


FIG. 6. *The full integrand* $y(x) = \phi_2(x)I(x, \lambda, \phi_1)$ *for the outer integral on* I_0*:* $\lambda = 2$ (*solid lines*), $\lambda = 3$ (*long-dashed lines*), $\lambda = 4$ (*short-dashed lines*), $\lambda = 5$ (*dotted lines*), $\lambda = 6$ (*dashed-dotted lines*).

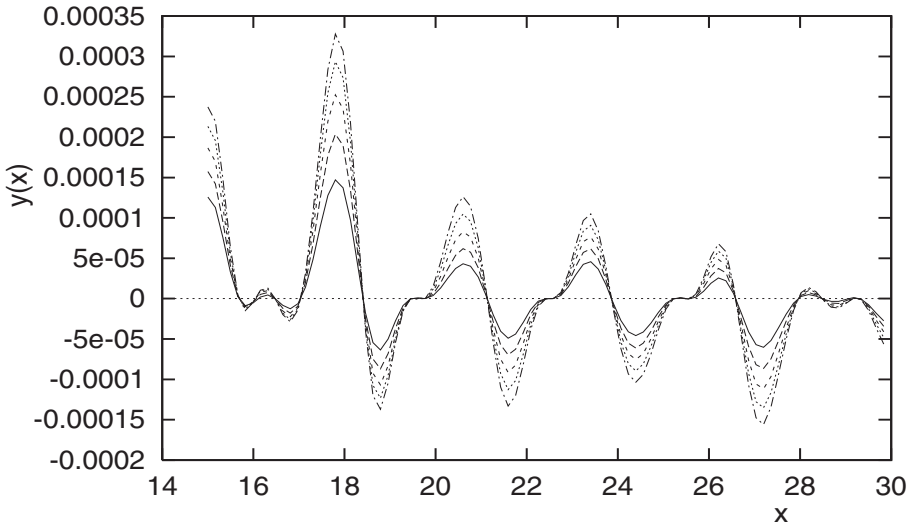


FIG. 7. *The full integrand* $y(x) = \phi_2(x)I(x, \lambda, \phi_1)$ *for the outer integral on* I_1*:* $\lambda = 2$ (*solid lines*), $\lambda = 3$ (*long-dashed lines*), $\lambda = 4$ (*short-dashed lines*), $\lambda = 5$ (*dotted lines*), $\lambda = 6$ (*dashed-dotted lines*).

has $n - 1$ nodes on the current interval, a convenient numerical method would be one in which, for a predetermined accuracy, the number of steps within the partition remains reasonably small, irrespective of n. Such a condition is not easily met because for most of the methods in current use the number of steps clearly increases with n. A second condition is the capacity of the method easily to allow for the evaluation of potentially valuable extra data. For example, a method which is able to furnish not only the eigenvalues and their eigenfunctions but also the first derivative of the

eigenfunctions should be preferred because the knowledge of the derivatives plays an important role in increasing the efficiency of the evaluation of the Slater integrals, as will be seen below. Finally, we require not only an ODE method but also a quadrature method. The latter is for the computation of the normalization integral, (2.3). The normalization should be sufficiently accurate using only the data provided by the ODE solver. For example, a pair of methods in which the ODE solver requires 100 steps while the quadrature rule needs data at 1,000 points is clearly unsuitable.

After a comparative examination of a number of possible candidates (methods written especially for Sturm–Liouville problems (see, e.g., [3], [29], [30], [31], and [32]) were considered with particular attention) we have opted for CPM{12, 10} of [19] which is a highly accurate version in the class of the constant (based) perturbation (CP) methods. A brief description of the method and of how it meets the aforementioned conditions is given in section 4. The theory of the CP methods is described in Chapter 3 of [16].

3.2. Computation of the Slater integrals. Only a few tens of steps are required for the CPM{12, 10} method to produce eigenvalues and eigenfunctions with a typical accuracy of 12 figures. However, the total number of the mesh points at which data are needed for an accurate computation of the Slater integrals by standard rules is generally much greater; as mentioned earlier, about 1,000 mesh points are required to achieve an accuracy of eight figures using Simpson's rule. Replacing the latter by a rule with more points, e.g., the Newton–Cote rule with five or six points, will indeed reduce the total number of mesh points, but such a reduction is still insufficient. Two other possible ways of improvement are available.

The first consists of using extended rules, i.e., rules in which the pointwise values of the integrand and of a number of its derivatives, if the latter are available, are involved. Rules of this type appear from time to time in the literature (see, e.g., [25]) but we are unaware of any systematic application of them.

Another way relies on a piece of information of a different nature. In addition to the pointwise values of the integrand and of its derivatives, this method also uses information on the global behavior of the integrand along the integration interval. In our case the integrand is a product of oscillatory functions whose frequencies are available. This opens up the possibility of formulating rules whose weights are no longer constant, as for the classical rules, but are frequency-dependent.

In section 5 a rule which combines all these possibilities will be described. It requires data between only 66 and 118 points.

3.3. Proposed procedure. In summary, the following main ingredients have been detected for a suitable two-stage computational strategy.

Stage 1: Solution of the Schrödinger equation using two different methods (one for $0 < x \leq x_{cut}$ and another for $x > x_{cut}$), each based on the piecewise perturbation theory.

Stage 2: Computation of Slater integrals using sets of extended frequency-dependent quadrature rules (EFDQRs).

These two stages are developed in section 4 and 5, respectively.

4. CP methods for the Schrödinger equation. These are methods for the initial value problem but they can also be used in a shooting scheme for solving the boundary value problem; see [16] for the general theory.

To illustrate the concept we consider the one-step Schrödinger equation,

$$y'' = (V(x) - E)y, \ X < x < X + h. \tag{4.1}$$

The solution is advanced from one end to the other by means of the propagation matrix algorithm:

$$\begin{bmatrix} y(X+h) \\ y'(X+h) \end{bmatrix} = \begin{bmatrix} u(h) & v(h) \\ u'(h) & v'(h) \end{bmatrix} \begin{bmatrix} y(X) \\ y'(X) \end{bmatrix}. \tag{4.2}$$

The problem then reduces to finding accurate values for the four elements of this matrix. We note that the algorithm automatically meets one of the previously stated conditions; i.e., it allows for the determination of both the solution and its first derivative.

The functions $u(\delta)$ and $v(\delta)$, with $\delta = x - X$, are called propagators and satisfy the local problem

$$y''(\delta) = (V(X+\delta) - E)y(\delta), \qquad \delta \in [0, h].$$

The initial values for u are

$$y(0) = 1,\ y'(0) = 0,$$

while those for v are

$$y(0) = 0,\ y'(0) = 1.$$

Evaluation, at $\delta = h$, of these propagators and of their derivatives with respect to δ provides the matrix elements required.

The computation of the four matrix elements involves three steps.

, , 1 The potential $V(X+\delta)$ is approximated by a sum over shifted Legendre polynomials $P_n^*(\delta/h)$ with a finite number N of terms:

$$V(X+\delta) \approx \tilde{V}(x+\delta) = \sum_{n=0}^{N} V_n h^n P_n^*(\delta/h).$$

The coefficient V_n is calculated by the quadrature:

$$V_n = \frac{2n+1}{h} \int_0^h V(X+\delta) P_i^* \left(\frac{\delta}{h}\right)\, d\delta.$$

For shifted Legendre polynomials, see [1]. The first three Legendre polynomials are $P_0^*(\gamma) = 1$, $P_1^*(\gamma) = -1 + 2\gamma$, and $P_2^*(\gamma) = 1 - 6\gamma + 6\gamma^2$ for $\gamma \in [0, 1]$.

, , 2 The potential $\tilde{V}(X+\delta)$ is separated into two parts: the constant reference potential

$$\overline{V} = V_0,$$

and the perturbation

$$\Delta V^{(N)}(\delta) = \sum_{n=1}^{N} V_n h^n P_n^*(\delta/h).$$

Indeed, the method's name is derived from this separation: as stated earlier, CP is an abbreviation for constant (based) perturbation. One way to identify the version

of a CP method[1] is to specify the values of N and Q; the latter is the number of perturbative corrections included in the algorithm. It should be noted that as the approximation concerns only the potential function, the accuracy of the solution should be practically independent of E. This expectation is confirmed by a theorem given in [19].

We have opted for version CPM{12, 10} of [19].[2] Numerical tests using this version and its comparison against some well-established codes are detailed in [19], [20], [21], and [22].

3 The four matrix elements are computed by means of some ready-to-use expressions which result from the perturbation treatment described above. These expressions become especially simple and transparent if written in terms of some ad-hoc functions. These functions were first introduced in [16] and are defined as follows:

$$\eta_{-1}(Z) = \begin{cases} \cos(|Z|^{1/2}) & \text{if } Z \le 0, \\ \cosh(Z^{1/2}) & \text{if } Z > 0, \end{cases}$$

$$\eta_0(Z) = \begin{cases} \sin(|Z|^{1/2})/|Z|^{1/2} & \text{if } Z < 0, \\ 1 & \text{if } Z = 0, \\ \sinh(Z^{1/2})/Z^{1/2} & \text{if } Z > 0, \end{cases}$$

$$\eta_m = [\eta_{m-2}(Z) - (2m-1)\eta_{m-1}(Z)]/Z, \qquad m = 1, 2, 3, \ldots.$$

Some useful properties are given below:

(i) Series expansion:

$$\eta_m(Z) = 2^m \sum_{q=0}^{\infty} \frac{g_{m\,q}\, Z^q}{(2q+2m+1)!},$$

with

$$g_{m\,q} = \begin{cases} 1 & \text{if } m = 0, \\ (q+1)(q+2)\cdots(q+m) & \text{if } m > 0. \end{cases}$$

In particular,

$$\eta_m(0) = \frac{1}{(2m+1)!!}.$$

(ii) Asymptotic behavior at large $|Z|$:

$$\eta_m(Z) \approx \begin{cases} \eta_{-1}(Z)/Z^{(m+1)/2} & \text{for odd } m, \\ \eta_0(Z)/Z^{m/2} & \text{for even } m. \end{cases}$$

[1] Other rules for identification are also available. Thus, the rule used in [19] is aimed at highlighting the accuracy properties. The version identified there as CPM$\{n_0, n_{as}\}$ is a method in which the number of terms in $\Delta V^{(N)}(\delta)$ which are used at each order of the perturbation theory varies with the order in such a way that the one-step error behaves as h^{n_0+1} when E is such that $|E - \bar{V}_0|h^2$ is small and as $h^{n_{as}+1}/\sqrt{E}$ when $E \to \infty$.

[2] Higher-order versions have recently became available; see [26], [27]. In fact, the perturbations can be calculated analytically to any order, but further progress in this direction will involve increasingly complex sets of symbolic computations.

(iii) Differentiation properties:

$$\eta'_m(Z) = \frac{1}{2}\eta_{m+1}(Z), \qquad m = -1, 0, 1, 2, \ldots.$$

Each propagator is expressed as a linear combination of such functions with perturbation-dependent coefficients. The argument of each of the η functions involved is $Z = (\bar{V} - E)h^2$; i.e., it accounts for the reference potential and for the energy.

For example, the first propagator of CPM{12, 10} is

$$u(h) = \eta_{-1}(Z) + \sum_{m=1}^{5} C_m \eta_m(Z). \tag{4.3}$$

With $\bar{V}_n = V_n h^{n+2}$, the first two coefficients are

$$C_1 = -\left[\bar{V}_1 + \bar{V}_3 + \bar{V}_5 + \bar{V}_7 + \bar{V}_9\right]/2,$$

which includes a set of first-order contributions, and

$$C_2 = \left[5\bar{V}_3 + 14\bar{V}_5 + 27\bar{V}_7 + 44\bar{V}_9\right]/2 - \left[105\bar{V}_1^2 + 63\bar{V}_2^2 + 45\bar{V}_3^2 + 35\bar{V}_4^2\right]/2520,$$

which collects some contributions from the first and second orders. The expressions of the other elements of the propagation matrix have a similar structure.

A practical advantage of any CP method with propagators expressed as linear combinations of the η functions is that the C coefficients can be calculated once, at the beginning of the whole computation. Any further computation of the propagation matrix at different E will require only the evaluation of the set of η values. This leads to a substantial reduction in CPU time.

4.1. Normalization of the solution. Differentiating (4.1) and (4.2) with respect to E gives the equation

$$y''_E = (V(x) - E)y_E - y, \; X < x < X + h, \tag{4.4}$$

and its one-step algorithm

$$\begin{bmatrix} y_E(X+h) \\ y'_E(X+h) \end{bmatrix} = \begin{bmatrix} u_E(h) & v_E(h) \\ u'_E(h) & v'_E(h) \end{bmatrix} \begin{bmatrix} y(X) \\ y'(X) \end{bmatrix} + \begin{bmatrix} u(h) & v(h) \\ u'(h) & v'(h) \end{bmatrix} \begin{bmatrix} y_E(X) \\ y'_E(X) \end{bmatrix}. \tag{4.5}$$

Calculation of the propagators with E suffixes requires only minimal extra effort. The case of $u_E(h)$ for CPM{12, 10} provides an illustration. Here $u_E(h)$ is the partial derivative of (4.3) with respect to E, an operation that involves only the η components. We have

$$u_E(h) = -\frac{1}{2}h^2\left[\eta_0(Z) + \sum_{m=1}^{5} C_m \eta_{m+1}(Z)\right], \tag{4.6}$$

where property (iii) of the η functions has been used. We see that no extra data are needed for the evaluation of $u_E(h)$.

The solution of this related equation is of direct importance in our context; see [17], [19]. First, it helps in constructing a Newton–Raphson recurrence scheme for fast localization of the eigenvalues. Second, it furnishes a simple way of evaluating the normalization constant, thus fulfilling one of the previously addressed conditions. By multiplying (4.1) by y_E, (4.4) by y, then subtracting and integrating over the current step, we get

$$\int_X^{X+h} dx\, y^2(x) = [y'(x)y_E(x) - y'_E(x)y(x)]|_X^{X+h}.$$

4.2. Determination of the frequency of the eigenfunctions. Each eigenfunction $y_{nl}(x)$, $x \in I_i$, has an oscillatory behavior, and the knowledge of its frequency, F_{nl}, will play an important role in the formulation of the frequency-dependent quadrature rules to be developed in section 5. There is no need for a highly accurate computation of the frequencies; an accuracy of up to 10% is sufficient.

The WKB approximation [28] provides a convenient method for such a purpose. Let x_{low} and x_{up} be the two ends of the so-called classically admissible interval, i.e., such that $E_{nl} - V(x) \geq 0$ for all $x \in [x_{low}, x_{up}]$. Then

$$F_{nl} = \frac{1}{x_{up} - x_{low}} \int_{x_{low}}^{x_{up}} dx \sqrt{E_{nl} - V(x)}. \tag{4.7}$$

On interval I_0 this formula must be slightly modified. The modification consists of the Langer correction, that is, using

$$V^L(x) = \frac{(l+1/2)^2}{x^2} - \frac{2Z}{x} \tag{4.8}$$

in place of $V(x)$.

A practical problem is that the resulting number of frequencies is $n_{max}^4 \times (l_{max} + 1)^4$, and this is too large for efficient computation and storage of the weights of the extended frequency-dependent quadrature rules. The following analysis shows that a reduction of this number to only n_{max}^4 is possible.

In Figure 8 we plot $V(x)$ for $l = 0, 1, 2, 3, 4$, and $Z = 1$. We see that the four curves differ visibly from each other only in the vicinity of the origin. As x is advanced they merge into each other and collectively tend to zero. Accordingly, it is expected that the l dependence of the eigenvalues E_{nl} (fixed n) and of the associated eigenfunctions $y_{nl}(x)$ on the interval I_i will be maximal for $i = 0$ and will gradually decline as i is increased. It is also expected that the shape of $y_{nl}(x)$ will tend to approach the pure trigonometric form corresponding to the free potential function $V(x) = 0$.

All these expectations are confirmed in Table 1 and Figures 9 and 10, where we have used $n = 5$. The consequence is that on any interval with $i \geq 1$ the frequencies can be accepted as being practically independent of l. This is less justified for $i = 0$. However, in what follows, this approximation is tacitly accepted on all intervals. On each interval, the values $f_n = F_{nl_{ref}}$ are retained for further use, where $l_{ref} = l_{max}/2$. The comparatively poorer quality of this approximation for $i = 0$ will be taken into account and compensated for.

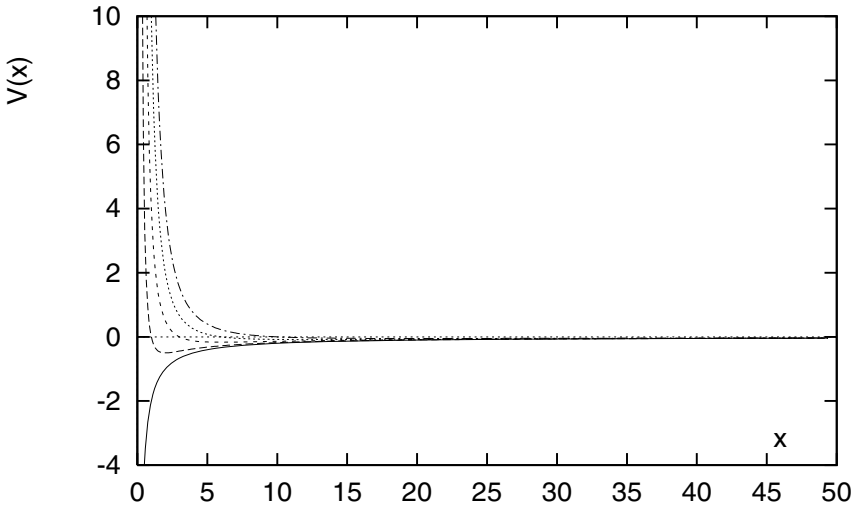


FIG. 8. *Potential function* $V(x) = l(l+1)/x^2 - 2/x$ *for various* l: $l = 0$ (*solid lines*), $l = 1$ (*long-dashed lines*), $l = 2$ (*short-dashed lines*), $l = 3$ (*dotted lines*), $l = 4$ (*dashed-dotted lines*).

TABLE 1
l dependence of eigenvalues E_{nl}, $n = 5$, *on four intervals.*

l	$i = 0$	$i = 1$	$i = 2$	$i = 3$
0	0.371436713890	0.609407751755	0.647763776061	0.663474817541
1	0.671040708780	0.613849124976	0.649245947289	0.664215844443
2	0.940539695698	0.622740318829	0.652210592373	0.665697935656
3	1.200829991937	0.636098448176	0.656658317453	0.667921166028
4	1.462529921844	0.653949762568	0.662590033989	0.670885647901

5. EFDQRs. We begin this section by showing that by using information about the nature of integrand we can improve the accuracy of the integral. As an illustration we compute

$$I(\omega) = (\omega + 1) \int_0^1 dx \, \frac{1}{(1+x)^2} \cos[(\omega + 1)x] \tag{5.1}$$

by dividing the interval [0, 1] into 20 equal subranges and by applying a three-point quadrature rule on each subrange. The following three-point versions are used, where x_1, x_2, x_3 are equidistant points and $h = (x_3 - x_1)/2$:

• the standard Simpson rule,

$$\int_{x_1}^{x_3} dx \, y(x) \approx h[y(x_1) + 4y(x_2) + y(x_3)]/3\,; \tag{5.2}$$

• the extended version of it [25] which additionally uses the values of the derivatives,

$$\int_{x_1}^{x_3} dx \, y(x) \approx h[7y(x_1) + 16y(x_2) + 7y(x_3) + h(y'(x_3) - y'(x_1))]/15\,; \tag{5.3}$$

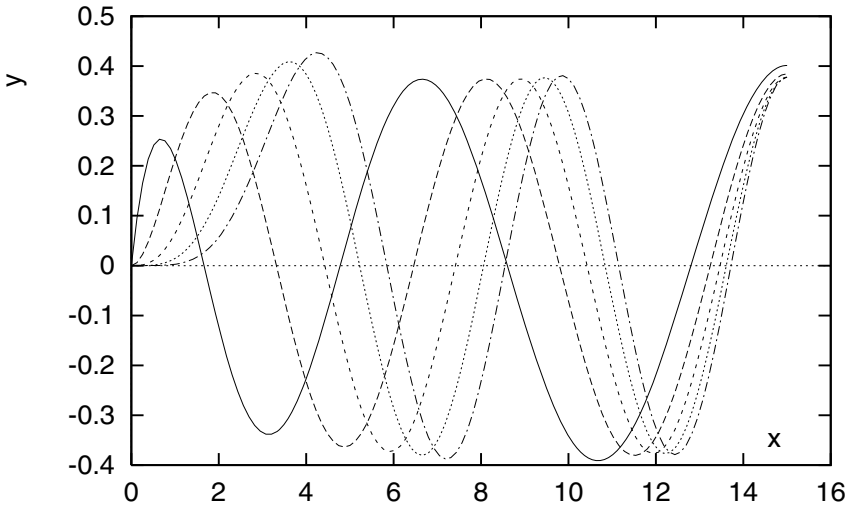


FIG. 9. *Eigenfunctions, $y_{nl}(x)$, on interval $I_0 = (0, 15]$ for $n = 5$ and various l: $l = 0$ (solid lines), $l = 1$ (long-dashed lines), $l = 2$ (short-dashed lines), $l = 3$ (dotted lines), $l = 4$ (dashed-dotted lines).*

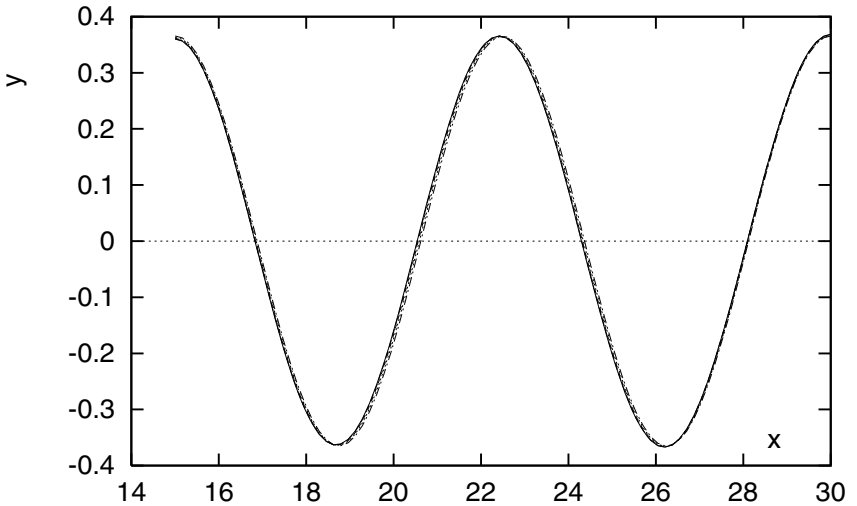


FIG. 10. *Eigenfunctions, $y_{nl}(x)$, on interval $I_1 = [15, 30]$, for $n = 5$ and various l: $l = 0$ (solid lines), $l = 1$ (long-dashed lines), $l = 2$ (short-dashed lines), $l = 3$ (dotted lines), $l = 4$ (dashed-dotted lines).*

- a version which uses the same set of input data on the integrand,

$$\int_{x_1}^{x_3} dx\, y(x) \approx h[a_1^{(0)}(y(x_1) + y(x_3)) + a_2^{(0)} y(x_2) + a_1^{(1)} h(y'(x_3) - y'(x_1))] \tag{5.4}$$

but where the coefficients $a_1^{(0)}$, $a_2^{(0)}$, and $a_1^{(1)}$ are chosen appropriately for integrands

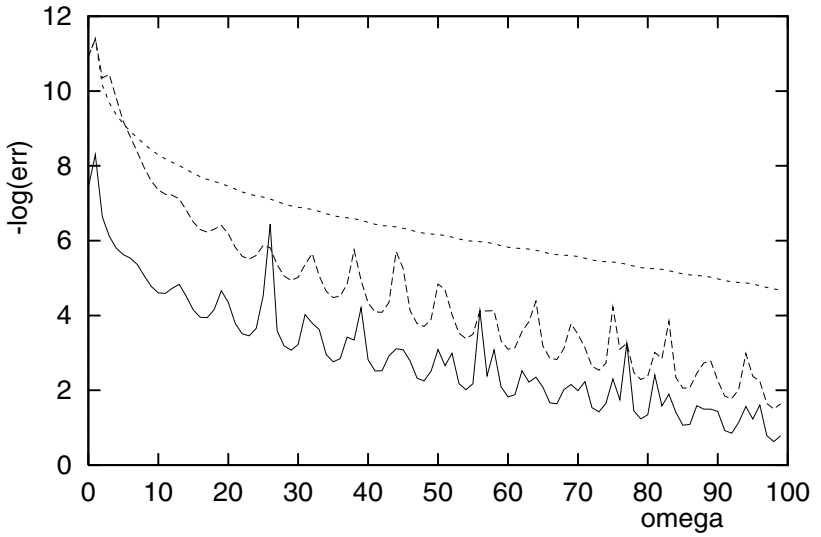


FIG. 11. *A comparison of the accuracies produced by the three quadrature rules for integral* (5.1): *Simpson* (*solid lines*), *extended Simpson* (*long-dashed lines*), *and extended Simpson with frequency dependent coefficients* (*short-dashed lines*).

of the form

$$y(x) = f_1(x)\sin(\omega x) + f_2(x)\cos(\omega x), \tag{5.5}$$

with smoothly varying $f_1(x)$ and $f_2(x)$. This case is covered by the exponential fitting procedure discussed in [23]; see also [18] for a succinct presentation. We do not enter into details about this general procedure but note that for the problem under discussion it consists of imposing the condition that (5.4) be exact if y is any of the three pairs of functions

$$x^p \exp(\pm i\omega x),\ p = 0,\ 1,\ 2. \tag{5.6}$$

The resulting system of linear equations for the three coefficients can be solved either analytically or numerically. We have opted for the numerical solution by means of the subroutine `efextqs` in the CD included with [23].

As to the quality of the three rules, the first two are exact when y is a polynomial of degree 3 and 5, respectively, while the third is exact if y is of the form (5.5), and $f_1(x)$ and $f_2(x)$ are second degree polynomials. The quantity

$$-\log[|I(\omega) - I^{comput}(\omega)|]$$

represents roughly the number of exact figures in $I(\omega)$ after the decimal point. This is displayed in Figure 11 for the three versions and for $0 < \omega < 100$. We see that all three are rather accurate for small ω. We also see that the accuracy exhibits a deterioration as ω is increased but at different rates, much in the favor of the last version, as expected.

The example clearly shows that, by exploiting the oscillatory nature of the integrand, we can significantly improve the accuracy of the result at little or no extra cost. This is the strategy adopted in the following section.

5.1. The quadrature rules. All quadrature rules involved in the calculation of the Slater integrals will be N-point extended frequency-dependent rules using equidistant mesh points, and will be of the form

$$\int_{x_1}^{x_N} dx\, y(x) \approx h \sum_{n=1}^{N} [a_n^{(0)} y(x_n) + h a_n^{(1)} y'(x_n)], \tag{5.7}$$

with $h = (x_N - x_1)/2$, where the coefficients satisfy the following symmetry/antisymmetry properties: $a_n^{(j)} = (-1)^j a_{N-n+1}^{(j)}$, $j = 0, 1$, $n = 1, 2, \ldots, N$. It follows that only N coefficients have to be calculated.

In principle, N may vary from one integral to another, but we adopted one and the same value, $N = 14$. A notable exception will be in the evaluation of the one-step inner integrals, where $N = 15$ will be used; see below.

5.1.1. Computation of the off-diagonal sector integrals J_1 and J_2. Here we are interested in the computation of $J_1(X_j, X_{j+1}, \lambda, \phi)$ and $J_2(X_j, X_{j+1}, \lambda, \phi)$ (see (2.5) and (2.8)) that occur in off-diagonal sectors. The integrand is

$$y(x) = u(x)\phi(x) = u(x) y_{n_1 l_1}(x) y_{n_2 l_2}(x), \tag{5.8}$$

where $u(x) = x^\lambda$, $1/x^{\lambda+1}$ for J_1 and J_2, respectively.

Normally, the interval $[X_j, X_{j+1}]$ is divided into n_{int} equidistant subranges and the $N = 14$ point rule (5.7) is used on each subrange with the coefficients fixed once for all subranges. The integration then requires the values of $y_{nl}(x)$ and $y'_{nl}(x)$ at $npnt = 13 n_{int} + 1$ equidistant points.

The cases $j \neq 0$ and $j = 0$ need separate treatment because some special features are present in the latter. In particular, the wavefunctions are definitely nonoscillatory in the vicinity of the origin (see also Figure 9), the frequencies exhibit a relatively important l dependence, and special care is required in J_2 at the origin because of the presence of the factor $1/x^{\lambda+1}$.

$j \neq 0$ Extensive tests have shown that $n_{int} = 5$ is sufficient for an accuracy of 10 figures in J_1 and J_2. Data is therefore required only at $npnt = 66$ points. The frequencies involved in the computation of the rule coefficients are not the eigenfunction frequencies f_{n_1} and f_{n_2}, but some linear combinations of these. This results from examination of the shape of the integrand of (5.8). Indeed, factor $u(x)$ is a smooth function on each subrange, but each of $y_{n_1 l_1}(x)$ and $y_{n_2 l_2}(x)$ is well described by the form (5.5) with the frequencies f_{n_1} and f_{n_2}, respectively. The product of the three is therefore a sum of two (5.5)-like forms, viz.,

$$y(x) = f_1^1(x)\sin(\omega_1 x) + f_2^1(x)\cos(\omega_1 x) + f_1^2(x)\sin(\omega_2 x) + f_2^2(x)\cos(\omega_2 x), \tag{5.9}$$

where $f_m^k(x)$, $m, k = 1, 2$, are smooth functions and $\omega_1 = f_{n_1} + f_{n_2}$, $\omega_2 = |f_{n_1} - f_{n_2}|$.

The 14 equations for the same number of coefficients are generated on this basis, that is, by imposing that (5.7) be exact for any of the following 14 pairs of functions:

$$x^p \exp[\pm \omega_k x],\ p = 0, 1 \ldots, 6,\ k = 1, 2, \tag{5.10}$$

and the rule with the coefficients obtained in this way will be exact when $y(x)$ is of form (5.9) and each $f_m^k(x)$ is a polynomial of at most degree 6.

$j = 0$ The coefficients are computed in the same way as for the previous case. However, since the input individual frequencies f_{n_1} and f_{n_2} now provide a less accurate representation of the actual behavior of the eigenfunctions, a narrower partition in subranges is used to compensate for the inaccuracy. Again, extensive tests have shown that $n_{int} = 9$ is satisfactory.

The computation of J_2 requires special consideration because the factor $1/x^{\lambda+1}$ is singular at the origin. As a matter of fact, an interesting property is that for the values of λ involved in the Slater integrals, that is, $|l_1 - l_2| \le \lambda \le l_1 + l_2$, the integral J_2 is finite. To see this it is sufficient to check that its integrand is finite at the origin. We rely on the well-known fact that the eigenfunction y_{nl} can be written as $y_{nl}(x) = x^{l+1} g_{nl}(x)$, where function $g_{nl}(x)$ is smooth and slowly varying near the origin and tends to a nonzero value when $x \to 0$. The integrand $y(x)$ of J_2 then reads

$$y(x) = x^{l_1+l_2+1-\lambda} g_{n_1 l_1}(x) g_{n_2 l_2}(x) \tag{5.11}$$

such that $y(0)$ is finite, and then J_2 is also finite.

Strictly speaking this result is irrelevant in the present context because the computation of J_2 is involved only for $j > 0$. However, formula (5.11) has its own importance in a different context which appears elsewhere in our computation. This is when overlap integrals of the coulomb and centrifugal behaviors $1/x$ and $1/x^2$ are required (this means that $\lambda = 0$, 1) but calculating $y(0)$ by direct use of $1/x^{\lambda+1}$ gives overflow. This can be easily avoided in the following way. Since $y'_{nl}(0) = \delta_{l\,0} g_{nl}(0)$ the evaluation at $x = 0$ of the first derivative of $y(x)$ given in (5.11) leads to the following expressions in which only the values of the first derivative of the eigenfunctions are involved:

$$\lambda = 0: \qquad y(0) = 0, \qquad y'(0) = y'_{n_1 l_1}(0) y'_{n_2 l_2}(0),$$

$$\lambda = 1: \qquad y(0) = y'_{n_1 l_1}(0) y'_{n_2 l_2}(0), \qquad y'(0) = 0.$$

5.1.2. Computation of the diagonal sector two-dimensional integral, (2.6).

The inner integral was defined earlier as

$$I(z, \lambda, \phi) = \left[\frac{1}{z^{\lambda+1}} J_1(X_i,\, z,\, \lambda, \phi) + z^{\lambda} J_2(z,\, X_{i+1},\, \lambda, \phi) \right], \tag{5.12}$$

with $\phi(x) = y_{n_1 l_1}(x) y_{n_2 l_2}(x)$. We are interested in the computation of the value of this integral and of its first derivative at the mesh points $z = x_1 (= X_i)$, $x_2, \ldots,$ $x_{npnt} (= X_{i+1})$ with $npnt = 13 n_{int} + 1$. We have used $n_{int} = 9$ on interval I_0 and $n_{int} = 7$ on all other intervals.

A convenient way to compute these integrals involves separate evaluations of the appropriate one-step integrals. We use the identities

$$\int_{x_k}^{x_{k+1}} dx\, y(x) = \int_{x_{k-m}}^{x_{k+1}} dx\, y(x) - \int_{x_{k-m}}^{x_k} dx\, y(x) \tag{5.13}$$

and

$$\int_{x_k}^{x_{k+1}} dx\, y(x) = \int_{x_k}^{x_{k+m}} dx\, y(x) - \int_{x_{k+1}}^{x_{k+m}} dx\, y(x) \tag{5.14}$$

for positive integer m. The advantage here is that the evaluation of the required one-step integral is based on the evaluation of the two integrals on the r.h.s., which can be made as accurate as needed by choosing m sufficiently large. An adverse effect is also present: the near-cancellation of similar sized numbers potentially leads to some accuracy loss. However, by using the round-off analysis tool CADNA [9] we have verified that with $m = 14$ this loss is insignificant.

We use this approach to obtain a single final (5.7)-like form for the numerical evaluation. For reasons of clarity let us take $k = 1$; obviously, in this case only the form of (5.14) can be considered. With $m = 14$ we use an $N = 15$ point rule for the first integral,

$$\int_{x_1}^{x_{15}} dx\, y(x) \approx h^* \sum_{n=1}^{15} [a_n^{*(0)} y(x_n) + h^* a_n^{*(1)} y'(x_n)], \tag{5.15}$$

where $h^* = (x_{15} - x_1)/2$, and a rule with $N - 1 = 14$ points for the second, viz.,

$$\int_{x_2}^{x_{15}} dx\, y(x) \approx h \sum_{n=2}^{15} [b_{n-1}^{(0)} y(x_n) + h b_{n-1}^{(1)} y'(x_n)], \tag{5.16}$$

where $h = (x_{15} - x_2)/2$. On subtracting we get

$$\int_{x_1}^{x_2} dx\, y(x) \approx \sum_{n=1}^{15} [c_n^{(0)} y(x_n) + c_n^{(1)} y'(x_n)], \tag{5.17}$$

where

$$c_1^{(j)} = (h^*)^{j+1} a_1^{*(j)}, \qquad j = 0,\, 1, \tag{5.18}$$

and

$$c_n^{(j)} = (h^*)^{j+1} a_n^{*(j)} + h^{j+1} b_{n-1}^{(j)}, \quad n = 2, 3, \ldots, 15, \quad j = 0,\, 1. \tag{5.19}$$

All one-step evaluations of J_1 and J_2 were made in this way, and $N = 15$ was used in all cases. Since the integrand is of the form of (5.8), its behavior is as in (5.9), and therefore the coefficients a^* of the $N = 15$ point rule are computed by imposing the conditions

$$x^{p_k} \exp[\pm \omega_k x],\ k = 1,\, 2,\ p_1 = 0,\, 1 \ldots, 7,\ p_2 = 0,\, 1 \ldots, 6 \tag{5.20}$$

(one new pair is added to conditions (5.10)) while the set (5.10) was used for the 14 coefficients b.

With the one-step values of J_1 and J_2, the integral $I(x_n,\, \lambda,\, \phi)$ is simply

$$I(x_n,\, \lambda,\, \phi) = \frac{1}{x_n^{\lambda+1}} \sum_{k=1}^{n-1} J_1(x_k,\, x_{k+1},\, \lambda,\, \phi) + x_n^{\lambda} \sum_{k=n}^{npnt-1} J_2(x_k,\, x_{k+1},\, \lambda,\, \phi).$$

As for the first derivative, the direct differentiation of (5.12) gives

$$I'(z,\, \lambda,\, \phi) = -\frac{\lambda+1}{z^{\lambda+2}} J_1(X_j,\, z,\, \lambda,\, \phi) + \lambda z^{\lambda-1} J_2(z,\, X_j,\, \lambda,\, \phi),$$

whence

$$I'(x_n,\, \lambda,\, \phi) = -\frac{\lambda+1}{x_n^{\lambda+1}} \sum_{k=1}^{n-1} J_1(x_k,\, x_{k+1},\, \lambda,\, \phi) + \lambda x_n^{\lambda-1} \sum_{k=n}^{npnt-1} J_2(x_k,\, x_{k+1},\, \lambda,\, \phi).$$

. On the interval I_0 the values to be used at $x_1 = 0$ are $I(0,\, \lambda,\, \phi) = I'(0,\, \lambda,\, \phi) = 0$, as follows from (5.8) with $y_{nl}(x) = x^{l+1} g_{nl}(x)$.

An $N = 14$ point frequency-dependent rule of form (5.7) is used to compute the outer integral, R, defined earlier as

$$R = \int_{X_j}^{X_{j+1}} dx\, y_{n_3 l_3}(x) y_{n_4 l_4}(x)\, I(x, \lambda, \phi).$$

However, the number of the frequencies is larger than before. Indeed, the x dependence of function $I(x, \lambda, \phi)$ on $x \in [X_j, X_{j+1}]$ is of the form (5.9) with $\omega_1 = f_{n_1} + f_{n_2}$ and $\omega_2 = |f_{n_1} - f_{n_2}|$. The other factor in the integrand, $y_{n_3 l_3}(x) y_{n_4 l_4}(x)$, is of the same form but with the frequencies $\omega_3 = f_{n_3} + f_{n_4}$ and $\omega_4 = |f_{n_3} - f_{n_4}|$. It follows that their product will be a function that is well described by a sum of eight functions of form (5.5) with the following eight possible frequencies: $|\omega_1 \pm \omega_3|$, $|\omega_1 \pm \omega_4|$, $|\omega_2 \pm \omega_3|$, and $|\omega_2 \pm \omega_4|$. Let these be Ω_k, $k = 1, \ldots, 8$, in descending order. The 14 coefficients are then constructed on the condition that the rule is exact for the set

$$\exp[\pm\Omega_k x],\ x \exp[\pm\Omega_k x] \text{ (for } k = 1, \ldots, 6), \exp[\pm\Omega_7 x] \text{ and } \exp[\pm\Omega_8 x]. \tag{5.21}$$

6. Hamiltonian matrix construction. Finally, we turn our attention to the efficiency of the extended frequency-dependent quadrature rules (EFDQRs) in constructing the Hamiltonian matrix in each sector. The Hamiltonian matrix in a given sector is defined by

$$(\Theta_k^{LS\Pi} | \mathcal{H} | \Theta_{k'}^{LS\Pi}), \tag{6.1}$$

where

$$\mathcal{H} = \sum_{i=1}^{2} \left(-\frac{1}{2}\nabla_i^2 - \frac{Z}{r_i} \right) + \frac{1}{r_{12}}. \tag{6.2}$$

Here, Z is the nuclear charge and $r_{12} = |\mathbf{r}_1 - \mathbf{r}_2|$, with $\mathbf{r}_1$ and $\mathbf{r}_2$ being the vector coordinates of the two electrons. L, S, and Π are the orbital, spin, and parity of the two-electron system, respectively.

The R-matrix basis states, $\Theta_k^{LS\Pi}(\mathbf{r}_1, \mathbf{r}_2)$, are expanded in each sector in terms of two-electron functions, $\Phi_{n_1 l_1 n_2 l_2}^{LS\Pi}(\mathbf{r}_1, \mathbf{r}_2)$, as follows:

$$\Theta_k^{LS\Pi}(\mathbf{r}_1, \mathbf{r}_2) = \sum_{n_1 l_1 n_2 l_2} \Phi_{n_1 l_1 n_2 l_2}^{LS\Pi}(\mathbf{r}_1, \mathbf{r}_2) \alpha_{n_1 l_1 n_2 l_2 k}^{LS\Pi}. \tag{6.3}$$

In a diagonal sector the normalized two-electron functions, $\Phi_{n_1 l_1 n_2 l_2}^{LS\Pi}(\mathbf{r}_1, \mathbf{r}_2)$, are given by

$$n_1 l_1 = n_2 l_2 : \quad \Phi_{nlnl}^{LS\Pi}(\mathbf{r}_1, \mathbf{r}_2) = x_1^{-1} y_{nl}(x_1) x_2^{-1} y_{nl}(x_2) \mathcal{Y}_{llLM_L}(\widehat{\mathbf{r}}_1, \widehat{\mathbf{r}}_2) \tag{6.4}$$

and

$$n_1 l_1 > n_2 l_2 :$$

$$\begin{aligned} \Phi_{n_1 l_1 n_2 l_2}^{LS\Pi}(\mathbf{r}_1, \mathbf{r}_2) = \frac{1}{\sqrt{2}} [& x_1^{-1} y_{n_1 l_1}(x_1) x_2^{-1} y_{n_2 l_2}(x_2) \mathcal{Y}_{l_1 l_2 LM_L}(\widehat{\mathbf{r}}_1, \widehat{\mathbf{r}}_2) \\ & + (-1)^{L+S+l_1+l_2} x_2^{-1} y_{n_1 l_1}(x_2) x_1^{-1} y_{n_2 l_2}(x_1) \mathcal{Y}_{l_2 l_1 LM_L}(\widehat{\mathbf{r}}_1, \widehat{\mathbf{r}}_2)]. \end{aligned} \tag{6.5}$$

In this latter case the sum over $n_1 l_1$ and $n_2 l_2$ in (6.2) is such that only terms in which $n_1 l_1 \geq n_2 l_2$ are included, i.e., $l_1 \geq l_2$ with the proviso that if $l_1 > l_2$, then there are no restrictions on n_1 and n_2 (except a specified maximum value, n_{max}), but if $l_1 = l_2$, then $n_1 > n_2$ if $L + S$ is odd and $n_1 \geq n_2$ if $L + S$ is even.

In (6.4) and (6.5) the angular function $\mathcal{Y}_{l_1 l_2 L M_L}$ is only nonzero if $|l_1 - l_2| \leq L \leq l_1 + l_2$. The expansion coefficients, $\alpha^{LS\Pi}_{n_1 l_1 n_2 l_2 k}$, are eventually obtained by diagonalizing the Hamiltonian matrix in each sector.

The time taken to construct a Hamiltonian matrix is dominated by the computation of the Slater integrals on diagonal sectors. In Tables 2, 3, and 4 we consider three typical cases with fixed n but increasing l that result in 8187600, 85741800, and 302869500 Slater integrals on the respective diagonal sectors.[3] We compare the EFDQR strategy developed in this paper against the Simpson Rule (SR) approach implemented in the current two-dimensional propagator [13]. We focus on the broad computational trends rather than a detailed performance analysis of the two implementations. For each case we present timings for the construction of the Hamiltonian matrix on the four diagonal sectors shown in Figure 1. The timings were performed on a single processing element on each of the following three installations.

[14] IBM Power4+ processor.

[15] HP Itanium 2 cluster.

[11] SGI Origin 2000.

TABLE 2
Case 1. $L = 0$, $S = 0$, $\Pi = even$, $n_{max} = 20$, $l_{max} = 7$.

Sector	No. of Slater integrals	Matrix size	HPCx (secs)		Harvey (secs)		CSAR (secs)	
			SR	EFDQR	SR	EFDQR	SR	EFDQR
0, 0	8187600	1680	943	38	1005	64	3421	105
1, 1	8187600	1680	943	30	1005	33	3421	81
2, 2	8187600	1680	943	15	1005	18	3421	46
3, 3	8187600	1680	943	15	1005	18	3421	46

TABLE 3
Case 2. $L = 2$, $S = 0$, $\Pi = even$, $n_{max} = 20$, $l_{max} = 9$.

Sector	No. of Slater integrals	Matrix size	HPCx (secs)		Harvey (secs)		CSAR (secs)	
			SR	EFDQR	SR	EFDQR	SR	EFDQR
0, 0	85741800	5090	9880	163	10523	234	35830	494
1, 1	85741800	5090	9880	128	10523	162	35830	387
2, 2	85741800	5090	9880	113	10523	148	35830	353
3, 3	85741800	5090	9880	113	10523	148	35830	353

TABLE 4
Case 3. $L = 4$, $S = 0$, $\Pi = even$, $n_{max} = 20$, $l_{max} = 11$.

Sector	No. of Slater integrals	Matrix size	HPCx (secs)		Harvey (secs)		CSAR (secs)	
			SR	EFDQR	SR	EFDQR	SR	EFDQR
0, 0	302869500	8900	34899	472	37172	629	126565	1461
1, 1	302869500	8900	34899	373	37172	471	126565	1162
2, 2	302869500	8900	34899	358	37172	457	126565	1130
3, 3	302869500	8900	34899	358	37172	457	126565	1123

[3]The computation on off-diagonal sectors is much faster. For example, Case 3 on HPCx takes under 25 seconds.

TABLE 5
Average speedup of the EFDQRs across the four diagonal sectors of Figure 1.

Case	HPCx	Harvey	CSAR
1	38	30	49
2	76	60	90
3	89	74	103

The SR uses 1191 mesh points for each outer integral and scales consistently as expected across the three cases on each hardware platform. The resulting integrals are accurate to no better than seven significant figures. It should be noted that doubling the number of points in the SR points adds only one extra digit of accuracy. Since this rule converges linearly [24] it would require eight times as many points to achieve our targeted 10 figures of accuracy. However, the resulting computation time would be inordinately long.

The EFDQR strategy exhibits a more complicated, but equally consistent, behavior across the cases. The resulting integrals are accurate to no worse than 10 significant figures. Sector 0,0 always takes longer than sector 1,1 because of its special treatment of the origin. As mentioned in section 4.2 the frequency is only required to be accurate to within a few percent. This enables the frequency dependent weights computed in sector 1,1 to be reused in all subsequent sectors without significant loss of accuracy. This accounts for the improved performance on sectors 2,2 and 3,3. Indeed this setup time is independent of l and, within timing vagaries, can be seen to be constant on each platform.

The proportion of reusable inner integrals, $I(x_i, \lambda, \phi)$, increases progressively from Case 1 to Case 3. The EFDQR implementation exploits this characteristic by first computing and storing all possible inner integrals within a given sector. Such a strategy is practically impossible in SR because of the very large number of mesh points used. The benefit of this can be clearly seen in Table 5 which summarizes the average speedup of the EFDQR compared to SR.

7. Concluding remarks. In this paper we advocate the construction of handcrafted algorithms that exploit a problem's characteristics rather than the use of general "off the shelf" algorithms. The problem in this case is the computation of Slater integrals. We have focused on exploiting as many characteristics of the problem as possible to develop a two-stage strategy, where the first stage informs the second. In particular, we have focused on the l independence and n dependence of the frequencies of the oscillatory eigenfunctions that are solutions of the Schrödinger equation.

In the first stage the CP{12,10} method enables us to produce, to an accuracy of about 12 figures, eigenvalues, frequencies, normalized eigenfunctions, and the first derivative of the eigenfunctions in a few tens of steps using a fixed step size which is independent of n. By using this information and by exploiting the independence of the frequencies on l, we have been able to construct EFDQRs for the Slater integrals using the same mesh points. Further, the frequency-dependent weights calculated on a nonorigin diagonal sector can be used directly on any other nonorigin diagonal sector. The small number of mesh points enables storage and hence reuse of the inner integrals.

On realistic cases the resulting computational strategy is more accurate, giving an accuracy of 10 rather than 7 figures, and typically two orders of magnitude faster than the existing SR implementation. This new algorithm also improves the load balance

between diagonal and off-diagonal sectors, and it removes the bottleneck impeding current simulations involving high lying states of hydrogen.

Construction of the Hamiltonian matrix is an intermediate computation followed by parallel diagonalization. This is typically performed using multithreaded LAPACK or ScaLAPACK routines. Since each matrix element is independent there is considerable potential for parallel improvement. The next step is to exploit shared and distributed memory parallelism in the construction of the Hamiltonian matrix.

Acknowledgments. We are grateful to the UK EPSRC for the provision of computer time on the UK's two national supercomputing services. LGrI is indebted to the UK EPSRC for a Visiting Fellowship to Queen's University Belfast, where much of this work was carried out. NSS is grateful to Jean-Marie Chesneaux, Fabienne Jézéquel, and Christophe Denis for help with CADNA. NSS is also indebted to l'Université Pierre et Marie Curie, Paris for a Visiting Professorship during which some of this work was performed.

REFERENCES

[1] M. Abramowitz and I. A. Stegun, *Handbook of Mathematical Functions*, 8th ed., Dover, New York, 1972.

[2] M. Baertschy, T. N. Rescigno, W. C. McCurdy, J. Coglan, and M. S. Pindzloa, *Ejected-energy differential cross sections for the near-threshold electron-impact ionization of hydrogen*, Phys. Rev. A, 63 (2001), pp. R50701–R50705.

[3] P. B. Bailey, M. K. Gordon, and L. F. Shampine, *Automatic solution of the Sturm-Liouville problem*, ACM Trans. Math. Software, 4 (1978), pp. 193–207.

[4] K. Bartschat, E. T. Hudson, M. P. Scott, P. G. Burke, and V. M. Burke, *Electron atom scattering at low and intermediate energies using a pseudo-state R-matrix basis*, J. Phys. B: At. Mol. Opt. Phys., 29 (1996), pp. 115–123.

[5] K. H. Becker, C. W. McCurdy, T. M. Orlando, and T. N. Rescigno, *Final Report on the Workshop on Electron-Driven Processes: Scientific Challenges and Technological Opportunities,* Stevens Institute of Technology, Hoboken, NJ, 2000. http://attila.stevens-tech.edu/physics/People/Faculty/Becker/EDP/index.html

[6] I. Bray, D. V. Fursa, A. S. Kheifets, and A. T. J. Stelbovics, *Electrons and photons colliding with atoms: Development and application of the convergent close-coupling method*, J. Phys. B: At. Mol. Opt. Phys., 35 (2002), pp. R117–R146.

[7] P. G. Burke and K. A. Berrington, *Atomic and Molecular Processes: An R-Matrix Approach*, IOP Publishing, Bristol/Philadelphia, 1993.

[8] P. G. Burke, C. J. Noble, and M. P. Scott, *R-matrix theory of electron scattering at intermediate energies*, Proc. Roy. Soc. Ser. A, 410 (1987), pp. 289–310.

[9] CADNA, http://www-anp.lip6.fr/cadna/Accueil.php

[10] A. Carson, T. H. Harmer, N. S. Scott, V. Faro-Mazo, M. P. Scott, and P. G. Burke, *2DRMP-G: Migrating a large-scale numerical mathematical application to a grid environment*, in High Performance Computing in Computational Science (VECPAR 2004), Lecture Notes in Comput. Sci. 3402, M. Daydé et al., eds., Springer, Berlin, 2005, pp. 233–246.

[11] CSAR (Computer Services for Academic Research), UK National Supercomputing Service, University of Manchester, Manchester, UK, http://www.cfs.ac.uk/

[12] K. M. Dunseath, M. Le Dourneuf, M. Terao-Dunseath, and J. M. Launay, *Two-dimensional R-matrix propagator: Application to electron-hydrogen scattering*, Phys. Rev. A, 54 (1996), pp. 561–572.

[13] J. W. Heggarty, P. G. Burke, M. P. Scott, and N. S. Scott, *Computational aspects of the two-dimensional propagation of R-matrices on MPPs*, Comput. Phys. Commun., 114 (1998), pp. 195–209.

[14] HPCx, UK National Supercomputing Service, Edinburgh, UK, http://www.hpcx.ac.uk/

[15] Harvey Itanium 2 cluster, Queen's University Belfast, Belfast, UK, http://harvey.am.qub.ac.uk/

[16] L. Gr. Ixaru, *Numerical Methods for Differential Equations and Applications*, D. Reidel, Dordrecht, 1984.

[17] L. Gr. Ixaru, M. Rizea, and T. Vertse, *Piecewise perturbation methods for calculating eigensolutions of a complex optical potential*, Comput. Phys. Commun., 85 (1995), pp. 217–230.
[18] L. Gr. Ixaru, *Operations on oscillatory functions*, Comput. Phys. Commun., 105 (1997), pp. 1–19.
[19] L. Gr. Ixaru, H. De Meyer, and G. Vanden Berghe, *CP methods for the Schrödinger equation revisited*, J. Comput. Appl. Math., 88 (1998), pp. 289–314.
[20] L. Gr. Ixaru, H. De Meyer, and G. Vanden Berghe, *SLCPM12—A program for solving regular Sturm–Liouville problems*, Comput. Phys. Commun., 118 (1999), pp. 259–277.
[21] L. Gr. Ixaru, H. De Meyer, and G. Vanden Berghe, *Highly accurate eigenvalues for the distorted Coulomb potential*, Phys. Rev. E, 61 (2000), pp. 3151–3159.
[22] L. Gr. Ixaru, *CP methods for the Schrödinger equation*, J. Comput. Appl. Math., 125 (2000), pp. 347–357.
[23] L. Gr. Ixaru and G. Vanden Berghe, *Exponential Fitting*, Kluwer, Dordrecht, 2004.
[24] F. Jézéquel, *Dynamical Control of Approximation Methods*, Habilitation à Diriger des Recherches, Université Pierre et Marie Curie, Paris 6, 2005.
[25] K. J. Kim, R. Cools, and L. Gr. Ixaru, *Quadrature rules using first derivatives for oscillatory integrands*, J. Comput. Appl. Math., 140 (2002), pp. 479–498.
[26] V. Ledoux, M. Van Daele, and G. Vanden Berghe, *CP methods of higher order for Sturm–Liouville and Schrödinger equations*, Comput. Phys. Commun., 162 (2004), pp. 151–165.
[27] V. Ledoux, M. Van Daele, and G. Vanden Berghe, *CP methods and the evaluation of negative Coulomb Whittaker function*, J. Comput. Appl. Math., 183 (2005), pp. 168–176.
[28] N. F. Mott and H. S. W. Massey, *The Theory of Atomic Collisions*, 3rd ed., Oxford University Press, Oxford, UK, 1965.
[29] S. Pruess and C. T. Fulton, *Mathematical software for Sturm-Liouville problems*, ACM Trans. Math. Software, 19 (1993), pp. 360–376.
[30] J. D. Pryce, *Numerical Solution of Sturm-Liouville Problems*, Oxford University Press, Oxford, UK, 1993.
[31] J. D. Pryce, *SLDRIVER Installation Guide and User Guide*, Technical report SEAS/CISE/JDP03/96, Royal Military College of Sciences, Shrivenham, UK, 1996.
[32] J. D. Pryce and M. Marletta, *Automatic solution of Sturm-Liouville problems using the Pruess method*, J. Comput. Appl. Math., 39 (1992), pp. 57–78.
[33] N. Przybilla and K. Butler, *Non-LTE, line formation for hydrogen revisited*, The Astrophysical Journal, 609 (2004), pp. 1181–1191.
[34] T. Stitt, N. S. Scott, M. P. Scott, and P. G. Burke, 2-*D R-matrix propagation: A large scale electron scattering simulation dominated by the multiplication of dynamically changing matrices*, in High Performance Computing for Computational Science (VECPAR 2002), Lecture Notes in Comput. Sci. 2565, J. M. L. M. Palma et al., eds., Springer, Berlin, 2003, pp. 354–367.

SIAM J. SCI. COMPUT.
Vol. 28, No. 4, pp. 1275–1288
© 2006 Society for Industrial and Applied Mathematics

EFFICIENT DETECTION OF PERIODIC ORBITS IN CHAOTIC SYSTEMS BY STABILIZING TRANSFORMATIONS*

JONATHAN J. CROFTS† AND RUSLAN L. DAVIDCHACK†

Abstract. An algorithm for detecting periodic orbits in chaotic systems [R. L. Davidchack and Y.-C. Lai, *Phys. Rev. E* (3), 60 (1999), pp. 6172–6175], which combines the set of stabilizing transformations proposed by Schmelcher and Diakonos [*Phys. Rev. Lett.*, 78 (1997), pp. 4733–4736] with a modified semi-implicit Euler iterative scheme and seeding with periodic orbits of neighboring periods, has been shown to be highly efficient when applied to low-dimensional systems. The difficulty in applying the algorithm to higher-dimensional systems is mainly due to the fact that the number of the stabilizing transformations grows extremely fast with increasing system dimension. Here we analyze the properties of stabilizing transformations and propose an alternative approach for constructing a smaller set of transformations. The performance of the new approach is illustrated on the four-dimensional kicked double rotor map and the six-dimensional system of three coupled Hénon maps.

Key words. unstable periodic orbits, kicked double rotor map, coupled Hénon maps

AMS subject classifications. 65P99, 65H20, 90C53

DOI. 10.1137/050623401

1. Introduction.

Periodic orbits play an important role in the analysis of various types of dynamical systems. In systems with chaotic behavior, unstable periodic orbits (UPOs) form a "skeleton" for chaotic trajectories [3]. A well-regarded definition of chaos [7] requires the existence of an infinite number of UPOs that are dense in the chaotic set. Different geometric and dynamical properties of chaotic sets, such as natural measure, Lyapunov exponents, fractal dimensions, and entropies [19], can be determined from the location and stability properties of the embedded UPOs. Periodic orbits are central to the understanding of quantum-mechanical properties of nonseparable systems: The energy level density of such systems can be expressed in a semiclassical approximation as a sum over the UPOs of the corresponding classical system [9]. Topological description of a chaotic attractor also benefits from the knowledge of periodic orbits. For example, a large set of periodic orbits is highly constraining to the symbolic dynamics and can be used to extract the location of a generating partition [5, 22]. The significance of periodic orbits for the experimental study of dynamical systems has been demonstrated in a wide variety of systems [16], especially for the purpose of controlling chaotic dynamics [20] with possible application in communication [2].

It is therefore not surprising that much effort has been put into the development of methods for locating periodic solutions in different types of dynamical systems. In a limited number of cases, this can be achieved due to the special structure of the systems. Examples include the Biham–Wenzel method applicable to Hénon-like maps [1], or systems with known and well-ordered symbolic dynamics [11]. For generic systems, however, most methods described in the literature use some type of an

*Received by the editors January 27, 2005; accepted for publication (in revised form) February 9, 2006; published electronically July 31, 2006. This work was supported by the EPSRC grant GR/S98986/01.

http://www.siam.org/journals/sisc/28-4/62340.html

†Department of Mathematics, University of Leicester, Leicester LE1 7RH, UK (jc84@le.ac.uk, rld8@mcs.le.ac.uk).

iterative scheme that, given an initial condition (seed), converges to a periodic orbit of the chaotic system. In order to locate all UPOs with a given period p, the convergence basin of each orbit for the chosen iterative scheme must contain at least one seed. The seeds are often chosen either at random from within the region of interest, from a regular grid, or from a chaotic trajectory with or without close recurrences. Typically, the iterative scheme is chosen from one of the "globally" convergent methods of quasi-Newton or secant type. However, experience suggests that even the most sophisticated methods of this type suffer from a common problem: With increasing period, the basin size of the UPOs becomes so small that placing a seed within the basin with one of the above listed seeding schemes is practically impossible [18].

A different approach, which appears to effectively deal with the problem of reduced basin sizes, has been proposed by Schmelcher and Diakonos (SD) [25, 26]. The basic idea is to transform the dynamical system in such a way that the UPOs of the original system become stable and can be located by simply following the evolution of the transformed dynamical system. That is, to locate period-p orbits of a discrete dynamical system

$$U: \quad x_{j+1} = f(x_j), \quad f: \mathbb{R}^n \mapsto \mathbb{R}^n, \tag{1}$$

one considers an associated flow

$$\Sigma: \quad \frac{dx}{ds} = Cg(x), \tag{2}$$

where $g(x) = f^p(x) - x$ and C is an $n \times n$ constant orthogonal matrix. It is easy to see that map $f^p(x)$ and flow Σ have identical sets of fixed points for any C, while C can be chosen such that unstable period-p orbits of U become stable fixed points of Σ.

Since it is not generally possible to choose a single matrix C that would stabilize all UPOs of U, the idea is to find the smallest possible set of matrices $\mathcal{C} = \{C_k\}_{k=1}^{K}$ such that, for each UPO of U, there is at least one matrix $C \in \mathcal{C}$ that transforms the unstable orbit of U into a stable fixed point of Σ. To this end, Schmelcher and Diakonos have put forward the following conjecture [25].

CONJECTURE 1. $\mathcal{C}_{\mathrm{SD}}$ $n \times n$ ± 1 $n \times n$ G $C \in \mathcal{C}_{\mathrm{SD}}$ CG

OBSERVATION 1. $\mathcal{C}_{\mathrm{SD}}$ B_n [14] n $\mathcal{C}_{\mathrm{SD}}$ $K = 2^n n!$

The above conjecture has been verified for $n \leq 2$ and appears to be true for $n > 2$, but, thus far, no proof has been presented. According to this conjecture, any periodic orbit whose stability matrix does not have eigenvalues equal to one can be transformed into a stable fixed point of Σ with $C \in \mathcal{C}_{\mathrm{SD}}$. In practice, to locate periodic orbits of the map U, we try to integrate the flow Σ from a given initial condition (seed) using different matrices from the set $\mathcal{C}_{\mathrm{SD}}$. Some of the resulting trajectories will converge to fixed points, while others will fail to do so, either leaving the region of interest or failing to converge within a specified number of steps.

The main advantage of the SD approach is that the convergence basins of the stabilized UPOs appear to be much larger than the basins produced by other iterative schemes [26, 15, 6], making it much easier to select a useful seed. Moreover, depending on the choice of the stabilizing transformation, the SD method may converge to several different UPOs from the same seed.

The flow Σ can be integrated by any off-the-shelf numerical integrator. Schmelcher and Diakonos have enjoyed considerable success using a simple Euler method. However, the choice of the integrator for this problem is governed by considerations very different from those typically used to construct an ODE solver. Indeed, to locate a fixed point of the flow, it may not be very efficient to follow the flow with some prescribed accuracy. Therefore, local error considerations, for example, are not as important. Instead, the goal is to have a solver that can reach the fixed point in as few integration steps as possible. In fact, as shown by Davidchack and Lai [4], the efficiency of the method can be improved dramatically when the solver is constructed specifically with the above goal in mind. In particular, recognizing the typical stiffness of the flow Σ, Davidchack and Lai have proposed a modified semi-implicit Euler method:

$$x_{j+1} = x_j + [\beta s_j C^{\mathsf{T}} - G_j]^{-1} g(x_j), \tag{3}$$

where $\beta > 0$ is a scalar parameter, $s_j \equiv \|g(x_j)\|$ is an L_2 norm, $G_j \equiv Dg(x_j)$ is the Jacobian matrix, and "T" denotes transpose. Note that, away from the root of g, the above iterative scheme is a semi-implicit Euler method with step size $h = (\beta s_j)^{-1}$ and, therefore, it can follow the flow Σ with a much larger step size than an explicit integrator (e.g., Euler or Runge–Kutta). Close to the root, the proposed scheme can be shown to converge quadratically [15], analogous to the Newton–Raphson method.

Another important ingredient of the algorithm presented in [4] is the seeding with already detected periodic orbits of neighboring periods. This seeding scheme appears to be superior to the typically employed schemes and enables fast detection of plausibly all[1] periodic orbits of increasingly larger periods in generic low-dimensional chaotic systems. For example, for the Ikeda map at traditional parameter values, the algorithm presented in [4] was able to locate plausibly all periodic orbits up to period 22 for a total of more than 10^6 orbit points. Obtaining a comparable result with generally employed techniques requires an estimated 10^5 larger computational effort.

While the stabilization approach is straightforward for relatively low-dimensional systems, direct application to higher-dimensional systems is much less efficient due to the rapid growth of the number of matrices in $\mathcal{C}_{\mathrm{SD}}$. Even though it appears that, in practice, far fewer transformations are required to stabilize all periodic orbits of a given chaotic system [21], the sufficient subset of transformations is not known a priori. It is also clear that the route of constructing a universal set of transformations is unlikely to yield substantial reduction in the number of such transformations. For instance, a smaller set of universal transformations with $K = (n+1)!$, which is isomorphic to the Weyl group A_n, is sufficient to stabilize all types of periodic orbits for $n < 4$ but can be shown to fail for certain types of orbits when $n \geq 4$. Therefore, a more promising way of using stabilizing transformations for locating periodic orbits in high-dimensional systems is to design such transformations based on the information about the properties of the system under investigation.

[1] It is not possible to prove, within our approach, the completeness of the detected sets of UPOs. Rather, our assertion of completeness is based on the plausibility argument. The following three criteria are used for the validation of the argument:

(i) Methods based on rigorous numerics (e.g., in [8]) have located the same UPOs in cases where such comparison is possible (usually for low periods, since these methods are less efficient).

(ii) Our search strategy scales with the period p (see section 4 and [6]). If we can tune it to locate all UPOs for low periods (where we can verify the completeness using (i)), it is likely (but not provably) capable of locating all UPOs of higher periods as well.

(iii) For maps with symmetries, we test the completeness by verifying that all the symmetric partners for all detected UPOs have been found (see sections 4.1 and 4.2).

In this paper, we propose to construct stabilizing transformations based on the knowledge of the stability matrices of already detected periodic orbits (used as seeds). The advantage of our approach is in a substantial reduction of the number of transformations, which increases the efficiency of the detection algorithm, especially in the case of higher-dimensional systems. The layout of the paper is as follows. In section 2 we study the properties of the stabilizing transformations for $n = 2$ and their relationship to the properties of the stability matrix of a periodic orbit. In section 3 we extend the analysis to higher-dimensional systems and show how to construct stabilizing transformations using the knowledge of the stability matrices of already detected periodic orbit points. In particular, we argue that the stabilizing transformations depend essentially on the signs of unstable eigenvalues and the directions of the corresponding eigenvectors of the stability matrices. Section 4 illustrates the application of the new stabilizing transformations to the detection of periodic orbits in a four-dimensional kicked double rotor map and a six-dimensional system of three coupled Hénon maps. We conclude in section 5 with the summary and discussion of possible further developments of the stabilizing transformations approach.

2. Stabilizing transformations in two dimensions. The stability of a fixed point x^* of the flow Σ is determined by the real parts of the eigenvalues of the matrix CG, where $G \equiv Dg(x^*)$ is the Jacobian matrix of $g(x)$ evaluated at x^*. For x^* to be a stable fixed point of Σ, the matrix C has to be such that all the eigenvalues of CG have negative real parts. In order to understand what properties of G determine the choice of a particular stabilizing transformation C, we use the following parametrization for the general two-dimensional orthogonal matrices:

$$C_{s,\alpha} = \begin{pmatrix} s\cos\alpha & \sin\alpha \\ -s\sin\alpha & \cos\alpha \end{pmatrix}, \tag{4}$$

where $s = \pm 1$ and $-\pi < \alpha \le \pi$. When $\alpha = -\pi/2,\ 0,\ \pi/2$, or π, we obtain the set of matrices $\mathcal{C}_{\rm SD}$. For example, $C_{1,\pi/2} = \left(\begin{smallmatrix} 0 & 1 \\ -1 & 0 \end{smallmatrix}\right)$ and $C_{-1,\pi} = \left(\begin{smallmatrix} 1 & 0 \\ 0 & -1 \end{smallmatrix}\right)$.

If we write $G \equiv g_{ij}$ $(i, j = 1, 2)$, then the eigenvalues of $C_{s,\alpha}G$ are given by the following equations:

$$\sigma_{1,2} = -A\cos(\alpha - \theta) \pm \sqrt{A^2\cos^2(\alpha - \theta) - s\det G}, \tag{5}$$

where $\det G = g_{11}g_{22} - g_{12}g_{21}$, $A = \frac{1}{2}\sqrt{(sg_{11} + g_{22})^2 + (sg_{12} - g_{21})^2}$, and

$$\tan\theta = \frac{sg_{12} - g_{21}}{-sg_{11} - g_{22}}, \qquad -\pi < \theta \le \pi. \tag{6}$$

Note that the signs of the numerator and denominator are significant for defining the angle θ in the specified range and should not be canceled out. It is clear from (5) that both eigenvalues have negative real parts when

$$s = \bar{s} \equiv \operatorname{sgn}\det G \qquad \text{and} \qquad |\alpha - \theta| < \tfrac{\pi}{2}, \tag{7}$$

provided that $\det G \neq 0$. This result proves the validity of Conjecture 1 for $n = 2$. Moreover, it shows that there are typically two matrices in $\mathcal{C}_{\rm SD}$ that stabilize a given fixed point.

The parameter θ clearly plays an important role in the above analysis. The following theorems show its relationship to the eigenvalues and eigenvectors of the stability matrix of a fixed point.

THEOREM 1. x^* $f^p(x) : \mathbb{R}^2 \mapsto \mathbb{R}^2$ $Df^p(x^*)$ $\lambda_{1,2}$ $|\lambda_2| < 1 < |\lambda_1|$ $0 \le \phi_{1,2} < \pi$ $v_{1,2} = (\cos\phi_{1,2}, \sin\phi_{1,2})^\mathsf{T}$ θ (6):

1. $\lambda_1 < -1$:

$$\theta \in \left(-\tfrac{\pi}{2}, \tfrac{\pi}{2}\right). \tag{8}$$

$|\lambda_1| \gg 1$

$$\theta \approx (\phi_1 - \phi_2)(\mathrm{mod}\ \pi) - \tfrac{\pi}{2}. \tag{9}$$

2. $\lambda_1 > 1$:

$$\theta = \begin{cases} \frac{3\pi}{2} - \phi_1 - \phi_2, & 0 < \phi_1 - \phi_2 < \pi, \\ \frac{\pi}{2} - \phi_1 - \phi_2, & -\pi < \phi_1 - \phi_2 < 0. \end{cases} \tag{10}$$

. The matrix $G = Df^p(x^*) - I$, where I is the identity matrix, can be written as follows:

$$G \equiv \begin{pmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{pmatrix} = \begin{pmatrix} \cos\phi_1 & \cos\phi_2 \\ \sin\phi_1 & \sin\phi_2 \end{pmatrix} \begin{pmatrix} \lambda_1 - 1 & 0 \\ 0 & \lambda_2 - 1 \end{pmatrix} \begin{pmatrix} \cos\phi_1 & \cos\phi_2 \\ \sin\phi_1 & \sin\phi_2 \end{pmatrix}^{-1}. \tag{11}$$

1. Since $\det G = (\lambda_1 - 1)(\lambda_2 - 1) > 0$, we set $s = 1$ and obtain from (6)

$$\tan\theta = \frac{(\lambda_1 - \lambda_2)\cot(\phi_1 - \phi_2)}{2 - \lambda_1 - \lambda_2}, \tag{12}$$

where, just as in (6), as well as in (13) and (16) below, the signs of the numerator and denominator should not be canceled out. Since $2 - \lambda_1 - \lambda_2 > 0$, we have that $\cos\theta > 0$ or

$$\theta \in \left(-\tfrac{\pi}{2}, \tfrac{\pi}{2}\right).$$

For $|\lambda_1| \gg 1$, (12) yields

$$\tan\theta \approx -\cot(\phi_1 - \phi_2),$$

and, given (8), the result in (9) immediately follows.

2. In this case $\det G = (\lambda_1 - 1)(\lambda_2 - 1) < 0$; thus, from (6) with $s = -1$,

$$\begin{aligned} \tan\theta &= \frac{(\lambda_2 - \lambda_1)\cos(\phi_1 + \phi_2)/\sin(\phi_1 - \phi_2)}{(\lambda_2 - \lambda_1)\sin(\phi_1 + \phi_2)/\sin(\phi_1 - \phi_2)} \\ &= \frac{-\cos(\phi_1 + \phi_2)/\sin(\phi_1 - \phi_2)}{-\sin(\phi_1 + \phi_2)/\sin(\phi_1 - \phi_2)}, \end{aligned} \tag{13}$$

since $\lambda_2 - \lambda_1 < 0$. The result in (10) follows. □

THEOREM 2. x^* $f^p(x) : \mathbb{R}^2 \mapsto \mathbb{R}^2$ $Df^p(x^*)$ $\lambda_{1,2} = \lambda \pm \mathrm{i}\omega$

$$\begin{aligned} &\theta \in \left(-\tfrac{\pi}{2}, \tfrac{\pi}{2}\right) && \lambda < 1, \\ &\theta \in \left(-\pi, -\tfrac{\pi}{2}\right) \bigcup \left(\tfrac{\pi}{2}, \pi\right) && \lambda > 1. \end{aligned} \tag{14}$$

. The stability matrix can be decomposed as follows:

$$Df^p(x^*) = \begin{pmatrix} \cos\phi & \mathrm{e}^{\eta} \\ \sin\phi & 0 \end{pmatrix} \begin{pmatrix} \lambda & \omega \\ -\omega & \lambda \end{pmatrix} \begin{pmatrix} \cos\phi & \mathrm{e}^{\eta} \\ \sin\phi & 0 \end{pmatrix}^{-1}, \tag{15}$$

where $\eta \in \mathbb{R}$. Given that $G = Df^p(x^*) - I$, we have from (6) that

$$\tan\theta = \frac{-\omega \cosh\eta / \sin\phi}{1-\lambda}. \tag{16}$$

The result in (14) follows from the sign of the denominator. □

The key message of the above theorems is that the stabilizing transformation matrix depends mostly on the directions of the eigenvectors and the signs of the unstable[2] eigenvalues (or their real parts), and only marginally on the actual magnitudes of the eigenvalues. This means that a transformation that stabilizes a given fixed point x^* of f^p will also stabilize fixed points of all periods with similar directions of eigenvectors and signs of the unstable eigenvalues. In the next section, we will show how this observation can be used to construct stabilizing transformations for efficient detection of periodic orbits in systems with $n > 2$.

3. Extension to higher-dimensional systems. To extend the analysis of the preceding section to higher-dimensional systems, we note that the matrix $C_{\bar{s},\theta}$, as defined by (4), (6), and (7), is closely related to the orthogonal part of the of G [10]. Recall that any nonsingular $n \times n$ matrix can be uniquely represented as a product

$$G = QB, \tag{17}$$

where Q is an orthogonal matrix and B is a symmetric positive definite matrix. The following theorem provides the link between $C_{\bar{s},\theta}$ and Q for $n = 2$.

THEOREM 3. $G \in \mathbb{R}^{2\times 2}$ $G = QB$ Q B $C_{\bar{s},\theta}$ (4), (6), (7) Q

$$C_{\bar{s},\theta} = -Q^{\mathsf{T}}. \tag{18}$$

. Since $C_{\bar{s},\theta}$ is an orthogonal matrix by definition, it is sufficient to prove that $C_{\bar{s},\theta}G$ is symmetric negative definite. Then, by the uniqueness of the polar decomposition, it must be equal to $-B$.

Denote by b_{ij} the element in the ith row and jth column of $C_{\bar{s},\theta}G$. We must show that $b_{12} = b_{21}$. Using (6), we have that

$$\begin{aligned} b_{12} &= \bar{s}g_{12}\cos\theta + g_{22}\sin\theta \\ &= \left[\bar{s}g_{12} + g_{22}\frac{\bar{s}g_{12} - g_{21}}{-\bar{s}g_{11} - g_{22}}\right]\cos\theta \\ &= \left[\frac{g_{11}g_{12} + g_{21}g_{22}}{\bar{s}g_{11} + g_{22}}\right]\cos\theta \end{aligned} \tag{19}$$

[2]That is, eigenvalues whose magnitude is larger than one.

and, similarly,

$$b_{21} = g_{21}\cos\theta - \bar{s}g_{11}\sin\theta = \left[g_{21} - \bar{s}g_{11}\frac{\bar{s}g_{12} - g_{21}}{-\bar{s}g_{11} - g_{22}}\right]\cos\theta = \left[\frac{g_{11}g_{12} + g_{21}g_{22}}{\bar{s}g_{11} + g_{22}}\right]\cos\theta\,; \tag{20}$$

hence the matrix $C_{\bar{s},\theta}G$ is symmetric. Since, by definition, θ and $\bar{s}$ are chosen such that the eigenvalues of $C_{\bar{s},\theta}G$ are negative, the matrix $C_{\bar{s},\theta}G$ is negative definite. Finally, by the uniqueness of the polar decomposition,

$$C_{\bar{s},\theta}G = -B = -Q^{\mathsf{T}}G\,,$$

which completes the proof. □

For $n > 2$, we can always use the polar decomposition to construct a transformation that will stabilize a given fixed point. Indeed, if a fixed point x^* of an n-dimensional flow has a nonsingular matrix $G \equiv Dg(x^*)$, then we can calculate the polar decomposition $G = QB$ and use

$$C = -Q^{\mathsf{T}} \tag{21}$$

to stabilize x^*. Moreover, by analogy with the two-dimensional case, we can expect that the same matrix C will also stabilize fixed points $\tilde{x}$ with the matrix $\tilde{G} \equiv Dg(\tilde{x})$, as long as the orthogonal part $\tilde{Q}$ of the polar decomposition $\tilde{G} = \tilde{Q}\tilde{B}$ is sufficiently close to Q. More precisely, we have the following observation.

OBSERVATION 2. *C [illegible] $\tilde{x}$ [illegible] $Q\tilde{Q}^{\mathsf{T}}$ [illegible]*

We base this observation on the following corollary of Lyapunov's stability theorem [13].

COROLLARY 4. *[illegible] $B \in \mathbb{R}^{n\times n}$ [illegible] $Q \in \mathbb{R}^{n\times n}$ [illegible] QB [illegible]*. According to Lyapunov's theorem, a matrix $A \in \mathbb{R}^{n\times n}$ has all eigenvalues with positive real parts if and only if there exists a symmetric positive definite $G \in \mathbb{R}^{n\times n}$ such that $GA + A^{\mathsf{T}}G = H$ is positive definite.

Let $A = QB$ and let us choose G in the form $G = \frac{1}{2}QB^{-1}Q^{\mathsf{T}}$. Since B is positive definite, its inverse B^{-1} is also positive definite, and, since G and B^{-1} are related by a congruence transformation, according to Sylvester's inertia law [12], G is also positive definite. Now,

$$GQB + (QB)^{\mathsf{T}}G = \tfrac{1}{2}QB^{-1}Q^{\mathsf{T}}QB + \tfrac{1}{2}BQ^{\mathsf{T}}QB^{-1}Q^{\mathsf{T}} = \tfrac{1}{2}[Q + Q^{\mathsf{T}}].$$

Therefore, QB has eigenvalues with positive real parts if and only if $\frac{1}{2}[Q + Q^{\mathsf{T}}]$ is positive definite. The proof is completed by observing that, for orthogonal matrices, the eigenvalues of $\frac{1}{2}[Q + Q^{\mathsf{T}}]$ are equal to the real parts of the eigenvalues of Q. □

Note that Observation 2 is a direct generalization of conditions in (7) which are equivalent to requiring that the eigenvalues of $C_{s,\alpha}C_{\bar{s},\theta}^{\mathsf{T}}$ have positive real parts.

In the scheme where already detected periodic orbits are used as seeds to detect other orbits [4], we can use C in (21) as a stabilizing matrix for the seed x^*. Based on the analysis in section 2, this will allow us to locate a periodic orbit in the neighborhood of x^* with similar invariant directions and the same signs of the unstable

eigenvalues. Note, however, that the neighborhood of the seed x^* can also contain periodic orbits with the similar invariant directions but with some eigenvalues having the opposite sign (i.e., orbits with and without reflections). To construct transformations that would stabilize such periodic orbits, we can determine the eigenvalues and eigenvectors of the stability matrix of x^*,

$$Df^p(x^*) = V\Lambda V^{-1}, \tag{22}$$

where $\Lambda \equiv \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$ is the diagonal matrix of eigenvalues of $Df^p(x^*)$ and V is the matrix of eigenvectors, and then calculate the polar decomposition of the matrix

$$\hat{G} = V(S\Lambda - I)V^{-1}, \tag{23}$$

where $S = \mathrm{diag}(\pm 1, \pm 1, \ldots, \pm 1)$. Note that, as follows from the analysis in section 2 for $n = 2$ and numerical evidence for $n > 2$, changing the sign of a stable eigenvalue will not result in a substantially different stabilizing transformation. Therefore, we restrict our attention to the following subset of S:

$$S_{ii} = \begin{cases} \pm 1, & |\lambda_i| > 1 \\ 1, & |\lambda_i| < 1 \end{cases} \quad \text{for} \quad i = 1, \ldots, n. \tag{24}$$

For a seed with k real unstable eigenvalues, this results in 2^k possible transformations. Note that, on the one hand, this set is much smaller than $\mathcal{C}_{\mathrm{SD}}$, while on the other hand, it allows us to target all possible types of periodic orbits that have invariant directions similar to those at the seed.

4. Numerical results. In this section we illustrate the performance of the new stabilizing transformations on a four-dimensional kicked double rotor map [24] and a six-dimensional system of three coupled Hénon maps [23]. Both systems are highly chaotic, and the number of UPOs is expected to grow rapidly with increasing period. The goal is to locate many UPOs of increasingly larger period. Of course, the completeness of the set of orbits for each period cannot be guaranteed, but it can be established with a high degree of certainty by using the plausibility criteria outlined in the introduction.

In order to start the detection process, we need to have a small set of periodic orbits (of period $p > 1$) that can be used as seeds. Such orbits can be located using, for example, random seeds and the standard Newton–Raphson method (or the scheme in (3) with $\beta = 0$). We can then use these periodic orbits as seeds to construct the stabilizing transformations and detect more UPOs with higher efficiency. The process can be iterated until we find no more orbits of a given period. In our previous work [4, 6] we showed that for two-dimensional maps such as Hénon and Ikeda it is sufficient to use period-$(p-1)$ orbits as seeds to locate plausibly all period-p orbits. For higher-dimensional systems, such as those considered in the present work, these seeds may not be sufficient. However, it is always possible to use more seeds by, for example, locating some of the period-$(p+1)$ orbits, which can then be used as seeds to complete the detection of period-p orbits. The following recipe can be used as a general guideline for developing a specific detection scheme for a given system:

1. Find a set of orbit points of low period using random seeds and the iterative scheme in (3) with $\beta = 0$ (i.e., the Newton–Raphson scheme).

2. To locate period-p orbits, first use period-$(p-1)$ orbits as seeds. For each seed x_0, construct 2^k stabilizing transformations C using (22)–(24), where k is the number of unstable eigenvalues of $Df^{p-1}(x_0)$.

3. Starting from x_0 and with a fixed value of $\beta > 0$, use the iteration scheme in (3) to construct a sequence $\{x_i\}$ for each of the 2^k stabilizing transformations. If a sequence converges to a point, check whether it is a new period-p orbit point and, if so, proceed to find a complete orbit by iterating the map f.

4. Repeat steps 2–3 for several β in order to determine the optimal value of this parameter (see explanation below).

5. Repeat steps 2–4 using newly found period-p points as seeds to search for period-$(p+1)$ orbits.

6. Repeat steps 2–4 using an incomplete set of period-$(p+1)$ orbits as seeds to find any missing period-p orbits.

Although we know that the action of β is to increase the basin size of the stabilized points, it is not known a priori what values of β to use for a given system and period. Monitoring the fraction of seeds that converge to periodic orbits, we observe that it grows with increasing β until it reaches saturation, indicating that the iterative scheme faithfully follows the flow Σ. On the other hand, larger β translates into smaller integration steps and, therefore, longer iteration sequences. Thus the optimal value of β is just before the saturation point. As demonstrated in our previous work [4] and observed in the numerical examples presented in the following sections, this value appears to scale exponentially with the period and can be estimated based on the information about the detection pattern at lower periods.

The stopping criteria in step 3, which we use in the numerical examples discussed below, are as follows. The search for UPOs is conducted within a rectangular region containing a chaotic invariant set. The sequence $\{x_i\}$ is terminated if (i) x_i leaves the region, (ii) i becomes larger than a predefined maximum number of iterations (we use $i > 100 + 5\beta$), (iii) the sequence converges, such that $\|g(x_i)\| <$, g. In cases (i) and (ii) a new sequence is generated from a different seed and/or with a different stabilizing matrix. In case (iii) five Newton iterations are applied to x_i to allow convergence to a fixed point to within the round-off error. A point x^* for which $\|g(x^*)\|$ is the smallest is identified with a fixed point of f^p. The maximum round-off error over the set $\mathcal{X}_p$ of all detected period-p orbit points,

$$\epsilon_{\max}(p) = \max\{\|g(x^*)\| : x^* \in \mathcal{X}_p\}, \tag{25}$$

is monitored in order to assess the accuracy of the detected orbits.

To check if the newly detected orbit is different from those already detected, its distance to other orbit points is calculated: If $\|x^* - y^*\|_\infty >$, x for all previously detected orbit points y^*, then x^* is a new orbit point. Even for a large number of already detected UPOs, this check can be done very quickly by presorting the detected orbit points along one of the system coordinates and performing a binary search for the points within , x of x^*. The infinity norm in the above expression is used for the computational efficiency of this check.

The minimum distance between orbit points,

$$d_{\min}(p) = \min\{\|x^* - y^*\|_\infty : x^*, y^* \in \mathcal{X}_p\}, \tag{26}$$

is monitored, and the algorithm is capable of locating all isolated UPOs of a given period p as long as $\epsilon_{\max}(p) <$, $g \lesssim$, $x < d_{\min}(p)$. Since typically $\epsilon_{\max}(p)$ increases and $d_{\min}(p)$ decreases with p (see Tables 1 and 2), the above conditions can be satisfied up to some period, after which higher-precision arithmetic needs to be used in the evaluation of the map. For the numerical examples presented in the following sections we use double-precision computation with , $g = 10^{-6}$ and , $x = 10^{-5}$.

TABLE 1

Number $n(p)$ of prime period-p UPOs, and the number $N(p)$ of fixed points of p-times iterated map for the kicked double rotor map. The asterisk for $p = 8$ indicates that this set of orbits is not complete. Parameters $\epsilon_{\max}(p)$ and $d_{\min}(p)$ are defined in (25) and (26).

p	$n(p)$	$N(p)$	$\epsilon_{\max}(p)$	$d_{\min}(p)$
1	12	12	$1.0 \cdot 10^{-14}$	$1.3 \cdot 10^{0}$
2	45	102	$5.9 \cdot 10^{-14}$	$3.4 \cdot 10^{-1}$
3	152	468	$5.8 \cdot 10^{-13}$	$6.2 \cdot 10^{-2}$
4	522	2190	$2.7 \cdot 10^{-12}$	$6.9 \cdot 10^{-3}$
5	2200	11 012	$2.6 \cdot 10^{-11}$	$1.1 \cdot 10^{-3}$
6	9824	59 502	$1.6 \cdot 10^{-10}$	$1.8 \cdot 10^{-4}$
7	46 900	328 312	$9.7 \cdot 10^{-10}$	$9.1 \cdot 10^{-5}$
8*	229 082	1 834 566	$1.2 \cdot 10^{-8}$	$5.5 \cdot 10^{-5}$

4.1. Kicked double rotor map. The kicked double rotor map describes the dynamics of a mechanical system known as the double rotor under the action of a periodic kick [24]. It is a four-dimensional map defined by

$$\begin{pmatrix} x_{n+1} \\ y_{n+1} \end{pmatrix} = \begin{pmatrix} My_n + x_n \pmod{2\pi} \\ Ly_n + c \sin x_{n+1} \end{pmatrix}, \tag{27}$$

where $x_n \in \mathbb{S}^2$ are the angle coordinates and $y_n \in \mathbb{R}^2$ are the angular velocities after each kick. Parameters L and M are constant 2×2 matrices that depend on the masses, lengths of rotor arms, and friction at the pivots, while $c \in \mathbb{R}^2$ is a constant vector whose magnitude is proportional to the kicking strength f_0. In our numerical tests we have used the same parameters as in [24], with the kicking strength $f_0 = 8.0$.

The following example illustrates the stabilizing properties of the transformations constructed on the basis of periodic orbits. Let us take a typical period-3 orbit point $x^* = (0.6767947, 5.8315697)$, $y^* = (0.9723920, -7.9998313)$ as a seed for locating period-4 orbits. The Jacobian matrix $Df^3(x^*, y^*)$ of the seed has eigenvalues $\Lambda = \mathrm{diag}(206.48, -13.102, -0.000373, 0.000122)$. Therefore, based on the scheme discussed in section 3, (22)–(24), we can construct four stabilizing transformations C corresponding to (S_{11}, S_{22}) in (24) being equal to $(+,+)$, $(-,+)$, $(+,-)$, and $(-,-)$. Of the total of 2190 orbit points of period 4 (see Table 1), the transformations C_1, C_2, C_3, and C_4 stabilize $\#(1) = 532$, $\#(2) = 544$, $\#(3) = 474$, and $\#(4) = 516$ orbit points, respectively, and these sets of orbits are almost completely nonoverlapping. That is, the number of orbits stabilized by both C_1 and C_2 is $\#(1 \cap 2) = 2$. Similarly, $\#(1 \cap 3) = 16$, $\#(1 \cap 4) = 0$, $\#(2 \cap 3) = 0$, $\#(2 \cap 4) = 14$, and $\#(3 \cap 4) = 0$. On the other hand, the number of period-4 orbits stabilized by at least one of the four transformations is $\#(1 \cup 2 \cup 3 \cup 4) = 2034$. This is a typical picture for other seeds of period 3 as well as other periods.

This example provides evidence for the validity of our approach to constructing the stabilizing transformations in high-dimensional systems based on periodic orbits. It also shows that, in the case of the double rotor map, a single seed is sufficient for constructing transformations that stabilize a majority of the UPOs. Of course, in order to locate the UPOs, we need to ensure that the seeds are in the convergence basins of the stabilized periodic orbits. That is why we need to use more seeds to locate plausibly all periodic orbits of a given period. Still, because of the enlarged basins of the stabilized orbits, the number of seeds is much smaller than that required with iterative schemes that do not use the stabilizing transformations.

Compared to the total of 384 matrices in $\mathcal{C}_{\mathrm{SD}}$, we use only two or four transformations for each seed, depending on the number of unstable directions of the seed

orbit points. Yet, the application of the detection scheme outlined in section 4 allows us to locate plausibly all periodic orbits of the double rotor map up to period 7. Table 1 also includes the number of detected period-8 orbits that were used as seeds to complete the detection of period 7.

The confidence with which we claim to have found plausibly complete sets of periodic orbits for each period is enhanced by the symmetry consideration. That is, since the double rotor map is invariant under the change of variables $(x, y) \mapsto (2\pi - x, -y)$, a necessary condition for the completeness of the set of orbits for each period is that for any orbit point (x^*, y^*) the set also contains an orbit point $(2\pi - x^*, -y^*)$. Even though this condition was not used in the detection scheme, we find that the detected sets of orbits (apart from period 8) satisfy this symmetry condition. Of course, this condition is not sufficient to prove the completeness of the detected sets of UPOs, but, combined with the exhaustive search procedure presented above, it provides a strong indication of the completeness.

4.2. Coupled Hénon maps. Another system we use to test the efficacy of our approach is a six-dimensional system of three coupled Hénon maps (CHM),

$$x_{n+1}^j = a - (\tilde{x}_n^j)^2 + b x_{n-1}^j \quad \text{for} \quad j = 1, 2, 3\,, \tag{28}$$

where $a = 1.4$ and $b = 0.3$ are the standard parameter values of the Hénon map and the coupling is given by

$$\tilde{x}_n^j = (1 - \epsilon) x_n^j + \frac{1}{2}\epsilon (x_n^{j+1} + x_n^{j-1}) \tag{29}$$

with $x_n^0 \equiv x_n^3$ and $x_n^4 \equiv x_n^1$. We have chosen the coupling parameter $\epsilon = 0.15$. Our choice of this system is motivated by the work of Politi and Torcini [23] in which they locate periodic orbits in CHM for a small coupling parameter by extending the method of Biham and Wenzel [1]. This makes the CHM an excellent test system, since we can compare our results against those for the Biham–Wenzel (BW) method. The BW method defines the following artificial dynamics:

$$\dot{x}_n^j(t) = (-1)^{s(n,j)} \{ x_{n+1}^j(t) - a + [\tilde{x}_n^j(t)]^2 - b x_{n-1}^j(t) \} \tag{30}$$

with $s(n, j) \in \{0, 1\}$. Given the boundary condition $x_{p+1}^j = x_1^j$, the equilibrium states of (30) are the period-p orbits for the CHM. The BW method is based on the property that every equilibrium state of (30) can be made stable by one of the 2^{3p} possible sequences of $s(n, j)$ and, therefore, can be located by simply integrating (30) to convergence starting from the same initial condition $x_n^j = 0.0$. It is also found that, for the vast majority of orbits, each orbit is stabilized by a unique sequence of $s(n, j)$.

In order to reduce the computational effort, Politi and Torcini suggest restricting the search to only those sequences $s(n, j)$ which are allowed in the uncoupled system, i.e., with $\epsilon = 0$. This reduction is possible because the introduction of coupling has the effect of pruning some of the orbits found in the uncoupled Hénon map without creating any new orbits.

We have implemented the BW method with both the full search and the reduced search (BW-r) up to as high a period as is computationally feasible (see Table 2). In the case of the full search we detect UPOs up to period 8 and in the case of the reduced search up to period 12. The seed $x_n^j = 0.0$ was used for all periods except for period 4, where it was found that with this seed both BW and BW-r located only 28 orbits. We found a maximum of 34 orbits using the seed $x_n^j = 0.5$. It is possible

TABLE 2

The number of prime UPOs for the system of CHM detected by three different methods: BW—full Biham–Wenzel, BW-r—reduced Biham–Wenzel, ST—our method based on stabilizing transformations. Max is the maximum number of detected UPOs obtained from all three methods and the system symmetry. See text for details.

p	BW	BW-r	ST	Max	$\epsilon_{\max}(p)$	$d_{\min}(p)$
1	8	8	8	8	$1.3 \cdot 10^{-14}$	$9.9 \cdot 10^{-1}$
2	28	28	28	28	$4.6 \cdot 10^{-14}$	$5.2 \cdot 10^{-1}$
3	0	0	0	0	-	-
4	34	34	40	40	$2.7 \cdot 10^{-8}$	$4.2 \cdot 10^{-2}$
5	0	0	0	0	-	-
6	74	74	72	74	$9.5 \cdot 10^{-10}$	$8.6 \cdot 10^{-3}$
7	28	28	28	28	$1.0 \cdot 10^{-8}$	$5.6 \cdot 10^{-3}$
8	271	271	285	286	$1.1 \cdot 10^{-6}$	$5.5 \cdot 10^{-3}$
9	-	63	64	66	$9.9 \cdot 10^{-7}$	$2.6 \cdot 10^{-4}$
10	-	565	563	568	$1.3 \cdot 10^{-8}$	$4.1 \cdot 10^{-4}$
11	-	272	277	278	$7.1 \cdot 10^{-9}$	$5.4 \cdot 10^{-4}$
12	-	1972	1999	1999	$2.5 \cdot 10^{-6}$	$4.3 \cdot 10^{-4}$
13*	-	-	1079	-	$8.6 \cdot 10^{-8}$	$4.0 \cdot 10^{-4}$
14*	-	-	6599	-	$2.3 \cdot 10^{-6}$	$3.5 \cdot 10^{-4}$
15*	-	-	5899	-	$7.0 \cdot 10^{-6}$	$1.5 \cdot 10^{-4}$

that more orbits can be found with different seeds for other periods as well, but we have not investigated this. The example of period 4 illustrates that, unlike for a single Hénon map, the Biham–Wenzel method fails to detect all orbits from a single seed.

Even though our approach (labeled "ST" in Table 2) is general and does not rely on the special structure of the Hénon map, its efficiency far surpasses the full BW method and is comparable to the BW-r method. Except for periods 6 and 10, the ST method locates the same or larger number of orbits.[3]

Unlike the double rotor map, the CHM possesses very few periodic orbits for small p, particularly for odd values of p. Therefore, we found that the direct application of the detection strategy outlined at the beginning of section 4 would not allow us to complete the detection of even period orbits. Therefore, for even periods p we also used $p + 2$ as seeds and, in case of period 12, a few remaining orbits were located with seeds of period 15. We did not attempt to locate a maximum possible number of UPOs for $p > 12$. The numbers of such orbits (labeled with asterisks) are listed in Table 2 for completeness.

As with the double rotor map, we used the symmetry of the CHM to test the completeness of the detected sets of orbits. It is clear from the definition of the CHM that all its UPOs are related by the permutation symmetry (i.e., six permutations of indices j). The column labeled "Max" in Table 2 lists the maximum number of UPOs that we were able to find using all three methods and applying the permutation symmetry to find any UPOs that might have been missed. As can be seen in Table 2, only a few orbits remained undetected by the ST method.

Concluding this section, we would like to point out that the high efficiency of the proposed method is primarily due to the fact that each stabilizing transformation constructed based on the stability properties of the seed orbit substantially increases the basins of convergence of orbits stabilized by this transformation. This is apparent in a typical increase of the fraction of converged seeds with the increasing value of

[3]The precise reason for the failure of the ST method to detect all period-6 and period-10 orbits needs further investigation. We believe that the orbits that were not detected have uncharacteristically small convergence basins with any of the applied stabilizing transformations.

parameter β in (3). For example, when detecting period-10 orbits of CHM using period-12 orbits as seeds, the fraction of seeds that converge to periodic orbits grows from 25–30% for small β (essentially the Newton–Raphson method) to about 70% for the optimal value of β.

5. Discussion and conclusions. We have presented a new scheme for constructing stabilizing transformations which can be used to locate periodic orbits in chaotic maps with the iterative scheme given by (3). The scheme is based on the understanding of the relationship between the stabilizing transformations and the properties of eigenvalues and eigenvectors of the stability matrices of the periodic orbits. Of particular significance is the observation that only the unstable eigenvalues are important for determining the stabilizing transformations. Therefore, unlike the original set of transformations proposed by Schmelcher and Diakonos, which grows with the system dimensionality as $2^n n!$, our set has the size of at most 2^k, where k is the maximum number of unstable eigenvalues (i.e., the maximum dimension of the unstable manifold). It is also apparent that, while the SD set contains a large fraction of transformations that do not stabilize any UPOs of a given system, all of our transformations stabilize a significant subset of UPOs. The dependence of the number of transformations on the dimensionality of the unstable manifold rather than on the system dimensionality is especially important in cases when we study low-dimensional chaotic dynamics embedded in a high-dimensional phase space. This is often the case in systems obtained from time-space discretization of nonlinear partial differential equations (e.g., the Kuramoto–Sivashinsky equation). Application of the stabilizing transformations approach to such high-dimensional chaotic systems will be the subject of our future work.

The new transformations were tested on two systems: a kicked double rotor map and three symmetrically coupled Hénon maps. We aimed to achieve a plausibly complete detection of periodic orbits of low periods up to as high a period as was computationally feasible. In both cases our algorithm was able to detect large numbers of UPOs with a high degree of certainty that the sets of UPOs for each period were complete. We have used the symmetry of the systems to test the completeness of the detected sets. On the other hand, when the aim is to detect as many UPOs as possible without verifying the completeness, the symmetry of the system could be used to increase the efficiency of the detection of UPOs: Once an orbit is detected, additional orbits can be located by applying the symmetry transformations.

One apparent drawback of the new scheme is that a small set of UPOs needs to be available for the construction of the stabilizing transformation at the start of the detection process. With the systems studied so far, we had no problem detecting UPOs of low period using the standard Newton–Raphson method by setting $\beta = 0$ in (3). However, in systems where it is hard to detect even a single periodic orbit, it would be useful to be able to determine stabilizing transformations without the knowledge of UPOs. Since the stabilizing transformations depend mostly on the properties of the unstable subspace, and since the decomposition into stable and unstable subspaces can be defined at any, not just periodic, point on the chaotic set, it should be possible to estimate such properties and construct the stabilizing transformations without the knowledge of the UPOs. The decomposition could be done, for example, in a process similar to that used in the subspace iteration algorithm [17], and a set of stabilizing transformations, for example, $\mathcal{C}_{\mathrm{SD}}$, could then be applied only within the unstable subspace. The feasibility of such a construction will be the topic for future investigation.

Acknowledgment. The authors would like to thank Alexander Gorban for useful discussion and numerous suggestions that helped improve the quality of the manuscript.

REFERENCES

[1] O. Biham and W. Wenzel, *Characterization of unstable periodic orbits in chaotic attractors and repellers*, Phys. Rev. Lett., 63 (1989), pp. 819–822.
[2] E. M. Bollt, Y.-C. Lai, and C. Grebogi, *Coding, channel capacity, and noise resistance in communicating with chaos*, Phys. Rev. Lett., 79 (1997), pp. 3787–3790.
[3] P. Cvitanović, *Periodic orbits as the skeleton of classical and quantum chaos*, Phys. D, 51 (1991), pp. 138–151.
[4] R. L. Davidchack and Y.-C. Lai, *Efficient algorithm for detecting unstable periodic orbits in chaotic systems*, Phys. Rev. E (3), 60 (1999), pp. 6172–6175.
[5] R. L. Davidchack, Y.-C. Lai, E. M. Bollt, and M. Dhamala, *Estimating generating partitions of chaotic systems by unstable periodic orbits*, Phys. Rev. E (3), 61 (2000), pp. 1353–1356.
[6] R. L. Davidchack, Y.-C. Lai, A. Klebanoff, and E. M. Bollt, *Towards complete detection of unstable periodic orbits in chaotic systems*, Phys. Lett. A, 287 (2001), pp. 99–104.
[7] R. L. Devaney, *An Introduction to Chaotic Dynamical Systems*, 2nd ed., Addison-Wesley, Reading, MA, 1989.
[8] Z. Galias, *Interval methods for rigorous investigations of periodic orbits*, Internat. J. Bifur. Chaos Appl. Sci. Engrg., 11 (2001), pp. 2427–2450.
[9] M. C. Gutzwiller, *Chaos in Classical and Quantum Mechanics*, Springer-Verlag, New York, 1990.
[10] P. Halmos, *Finite Dimensional Vector Spaces*, 2nd ed., Van Nostrand, Princeton, NJ, 1958.
[11] K. T. Hansen, *Alternative method to find orbits in chaotic systems*, Phys. Rev. E (3), 52 (1995), pp. 2388–2391.
[12] R. A. Horn and C. R. Johnson, *Matrix Analysis*, Cambridge University Press, Cambridge, UK, 1985.
[13] R. A. Horn and C. R. Johnson, *Topics in Matrix Analysis*, Cambridge University Press, Cambridge, UK, 1994.
[14] J. E. Humphreys, *Reflection Groups and Coxeter Groups*, Cambridge University Press, Cambridge, UK, 1990.
[15] A. Klebanoff and E. M. Bollt, *Convergence analysis of Davidchack and Lai's algorithm for finding periodic orbits*, Chaos Solitons Fractals, 12 (2001), pp. 1305–1322.
[16] D. P. Lathrop and E. J. Kostelich, *Characterization of an experimental strange attractor by periodic orbits*, Phys. Rev. A (3), 40 (1989), pp. 4028–4031.
[17] K. Lust, D. Roose, A. Spence, and A. R. Champneys, *An adaptive Newton–Picard algorithm with subspace iteration for computing periodic solutions*, SIAM J. Sci. Comput., 19 (1998), pp. 1188–1209.
[18] J. R. Miller and J. A. Yorke, *Finding all periodic orbits of maps using Newton methods: Sizes of basins*, Phys. D, 135 (2000), pp. 195–211.
[19] E. Ott, *Chaos in Dynamical Systems*, Cambridge University Press, Cambridge, UK, 1993.
[20] E. Ott, C. Grebogi, and J. A. Yorke, *Controlling chaos*, Phys. Rev. Lett., 64 (1990), pp. 1196–1199.
[21] D. Pingel, P. Schmelcher, and F. K. Diakonos, *Stability transformation: A tool to solve nonlinear problems*, Phys. Rep., 400 (2004), pp. 67–148.
[22] J. Plumecoq and M. Lefranc, *From template analysis to generating partitions* I: *Periodic orbits, knots, and symbolic encodings*, Phys. D, 144 (2000), pp. 231–258.
[23] A. Politi and A. Torcini, *Towards a statistical mechanics of spatiotemporal chaos*, Phys. Rev. Lett., 69 (1992), pp. 3421–3424.
[24] F. J. Romeiras, C. Grebogi, E. Ott, and W. P. Dayawansa, *Controlling chaotic dynamical systems*, Phys. D, 58 (1992), pp. 165–192.
[25] P. Schmelcher and F. K. Diakonos, *Detecting unstable periodic orbits of chaotic dynamical systems*, Phys. Rev. Lett., 78 (1997), pp. 4733–4736.
[26] P. Schmelcher and F. K. Diakonos, *General approach to the localization of unstable periodic orbits in chaotic dynamical systems*, Phys. Rev. E (3), 57 (1998), pp. 2739–2746.

SIAM J. SCI. COMPUT.
Vol. 28, No. 4, pp. 1289–1300
© 2006 Society for Industrial and Applied Mathematics

ENHANCING EIGENVALUE APPROXIMATION BY GRADIENT RECOVERY*

AHMED NAGA†, ZHIMIN ZHANG‡, AND AIHUI ZHOU§

Abstract. The polynomial preserving recovery (PPR) is used to enhance the finite element eigenvalue approximation. Remarkable fourth order convergence is observed for linear elements under structured meshes as well as unstructured initial meshes (produced by the Delaunay triangulation) with the conventional bisection refinement.

Key words. finite element method, recovery, superconvergence, eigenvalue

AMS subject classifications. 65N30, 65N15, 65N12, 65D10, 74S05, 41A10, 41A25

DOI. 10.1137/050640588

1. Introduction.

Recovery techniques such as the Zienkiewicz–Zhu superconvergence patch recovery (SPR) have been widely used in the finite element software industry for a posteriori error estimates and adaptive remeshing [1, 2, 14, 15]. Recently, we have discovered a new application for recovery techniques, which might have been overlooked in previous literature.

In this article, we report some remarkable enhancement results for the finite element eigenvalue approximation. The enhancement is based on gradient recovery techniques, especially the newly developed polynomial preserving recovery (PPR) [6, 7, 12, 13]. We know that the optimal convergence rate of the eigenvalue approximation by linear element for the Laplace operator is $O(h^2)$. Our numerical results indicate that the enhanced eigenvalue approximation converges at a rate of $O(h^4)$ for structured meshes as well as unstructured initial meshes (produced by the Delaunay triangulation) with regular refinement.

2. Eigenvalue enhancement.

On a domain $\Omega \subset \mathbb{R}^2$ with Lipschitz boundary, we consider a model eigenvalue problem: Find $(u, \lambda) \in H_0^1(\Omega) \times \mathbb{R}$ with $\|u\|_{0,\Omega} = 1$, such that

$$a(u, v) = \int_\Omega \nabla w \cdot \nabla v = \lambda(u, v) = \lambda \int_\Omega uv \quad \forall v \in H_0^1(\Omega). \tag{2.1}$$

We know that (2.1) has a countable sequence of real eigenvalues $0 < \lambda_1 \le \lambda_2 \le \cdots \to \infty$ and that their corresponding eigenfunctions are $u_1, u_2, \ldots$, which can be assumed to satisfy $(u_i, u_j) = \delta_{ij}$ for all $i, j \ge 1$.

The finite element approximation for (2.1) reads as follows: Find $(u_h, \lambda_h) \in S_0^h(\Omega) \times \mathbb{R}$ with $\|u_h\|_{0,\Omega} = 1$, such that

$$a(u_h, v_h) = \lambda_h(u_h, v_h) \quad \forall v_h \in S_0^h(\Omega) = S^h(\Omega) \cap H_0^1(\Omega), \tag{2.2}$$

*Received by the editors September 18, 2005; accepted for publication (in revised form) February 16, 2006; published electronically July 31, 2006.
http://www.siam.org/journals/sisc/28-4/64058.html

†Applied Automation Technologies, Inc., Troy, MI 48083 (anaga@capps-edges.com).

‡Corresponding author. Department of Mathematics, Wayne State University, Detroit, MI 48202 (zzhang@math.wayne.edu). This author's work was supported in part by National Science Foundation grant DMS-0311807.

§Institute of Computational Mathematics and Scientific/Engineering Computing Academy of Mathematics and System Science, Chinese Academy of Sciences, Beijing 100080, China (azhou@lsec.cc.ac.cn).

where $S^h(\Omega)$ is the usual piecewise linear C^0 finite element space. It is well known that (2.2) has a finite sequence of eigenvalues $0 < \lambda_{1,h} \le \lambda_{2,h} \le \cdots \le \lambda_{n_h,h}$ and corresponding eigenfunctions $u_{1,h}, u_{2,h}, \ldots, u_{n_h,h}$, where $(u_{i,h}, u_{j,h}) = \delta_{ij}$ for all $(1 \le i, j \le n_h)$ and $n_h = \dim S_0^h(\Omega)$. In addition (see, e.g., [3, 4]),

$$\lambda_{i,h} \ge \lambda_i, \quad i = 1, 2, \ldots, n_h. \tag{2.3}$$

The following identity of eigenvalue and eigenfunction approximation (see, e.g., [3, 10]) is crucial for our method.

LEMMA 2.1. [illegible] (u, λ) [illegible] (2.1) [illegible] $w \in H_0^1(\Omega) \setminus \{0\}$ [illegible]

$$\frac{a(w,w)}{\|w\|_{0,\Omega}^2} - \lambda = \frac{a(w-u, w-u)}{\|w\|_{0,\Omega}^2} - \lambda \frac{\|w-u\|_{0,\Omega}^2}{\|w\|_{0,\Omega}^2}. \tag{2.4}$$

Setting $w = u_h$ in (2.4) yields

$$\lambda_h - \lambda = \|\nabla(u - u_h)\|_{0,\Omega}^2 - \lambda \|u - u_h\|_{0,\Omega}^2. \tag{2.5}$$

2.1. The polynomial preserving recovery (PPR). Following Zhang and Naga [13], we introduce a PPR operator $G_h : S^h \longrightarrow S^h \times S^h$, which produces a continuous piecewise linear gradient field and satisfies the following two properties:

(1) Polynomial preserving: There exists a constant C independent of u and h, such that

$$\|\nabla u - G_h u_I\|_{0,\Omega} \le C h^2 |u|_{3,\Omega},$$

where $u_I \in S^h$ is the piecewise linear interpolation of u.

(2) Boundedness: There exists a constant C independent of v and h, such that

$$\|G_h v\|_{0,\Omega} \le C |v|_{1,\Omega} \quad \forall v \in S^h.$$

We expect that the recovered gradient $\mathrm{G_h}\, u_h$ is "closer" to ∇u than ∇u_h is. Toward this end, we need some restriction on the mesh.

We say that two adjacent triangles (sharing a common edge) form an $O(h^{1+\alpha})$ $(\alpha > 0)$ approximate parallelogram if the lengths of any two opposite edges differ by only $O(h^{1+\alpha})$.

DEFINITION. [illegible] $\mathcal{T}_h$ [illegible] Condition α [illegible] $O(h^{1+\alpha})$ [illegible]

A common mesh refinement strategy is the [illegible], which decomposes one triangle into four congruent subtriangles by linking the three edge centers. An important property of the bisection is that it satisfies [illegible] α with $\alpha = \infty$.

We are able to establish the following theorem for the eigenfunction approximation. The proof will be omitted here since it is a straightforward mimic of proofs in [6, 9].

THEOREM 2.1. [illegible] $u \in W_\infty^3(\Omega) \cap H_0^1(\Omega)$ [illegible] $u_h \in S_0^h(\Omega)$ [illegible] (2.1) [illegible] (2.2) [illegible] G_h [illegible] $\mathcal{T}_h$ [illegible] Condition α [illegible] C [illegible] u [illegible] h [illegible]

$$\|\nabla u - G_h u_h\|_{0,\Omega} \le C h^{1+\beta} \|u\|_{3,\infty,\Omega}, \qquad \beta = \min(\alpha, 1). \tag{2.6}$$

. . . . . Although the theory needs some mesh condition, the recovery procedure itself can be applied to arbitrary meshes including anisotropic cases.

In what follows we also require that both u and a sequence of meshes $\mathcal{T}_h$ satisfy the following nondegeneracy property: There exists a constant $C > 0$ independent of h such that

$$\|\nabla(u - u_h)\|_{0,\Omega} \geq Ch. \tag{2.7}$$

As argued by Dörfler and Nochetto [5, sect. 4], this is not a very restrictive condition in practice; it is guaranteed, for instance, if $|D^2 u(x)| \geq C > 0$ for all x in a fixed region of Ω.

2.2. Asymptotically exact error estimates for $\lambda_h - \lambda$. An immediate benefit from the improved gradient based on the PPR is its accurate prediction of the error. We have the following theorem.

THEOREM 2.2. . . . . $\mathrm{G_h}\, u_h$ . . . . (2.6) . . . .

2.3. Eigenvalue enhancement. Now we propose an enhanced eigenvalue approximation based on the PPR-recovered gradient:

$$\lambda_h^* = \lambda_h - \eta_h^2. \tag{2.11}$$

Note that the cost of computing quantity $\eta_h = \|\mathrm{G_h}\, u_h - \nabla u_h\|_{0,\Omega}$ is very low. We refer to [8] for a general defect correction approach to improve the approximation accuracy of eigenvalues and [11] for other eigenvalue improvement techniques. The emphasis here is on the PPR under arbitrary initial meshes and superconvergence.

THEOREM 2.3. [illegible] $\mathrm{G_h}\, u_h$ [illegible] (2.6) [illegible] (2.7) [illegible]

$$|\lambda_h^* - \lambda| \le Ch^{2+\beta}. \tag{2.12}$$

[illegible]. From (2.8) and the definition of λ_h^*, we see that

$$-Ch^{\beta}\eta_h^2 \le \lambda_h^* - \lambda \le Ch^{\beta}\eta_h^2. \tag{2.13}$$

By the nondegeneracy condition (2.7) and the equivalence between η_h and $\|e_h\|_{0,\Omega}$ (2.9), we get $\eta_h = O(h)$. The conclusion follows by using $\eta_h = O(h)$ in (2.13). ∎

By (2.5), we see that the optimal rate for the linear finite element approximation of the eigenvalue is $O(h^2)$. Therefore, Theorem 2.3 is a superconvergence result.

Under uniform triangular mesh of the regular or chevron pattern, we can prove (2.6) with $\beta = 1$ for the linear finite element; see [13]. Then, by Theorem 2.3, we have

$$|\lambda_h^* - \lambda| \le Ch^3.$$

However, our numerical tests indicate

$$|\lambda_h^* - \lambda| \approx Ch^4$$

for all four patterns of uniform triangulation (see Figure 3.1) as well as uniform equilateral triangulation, and for unstructured initial meshes (generated by the Delaunay triangulation) with the bisection refinement. It seems that the error estimates in Theorems 2.2 and 2.3 are not optimal. Our conjecture is that instead of (2.9), we actually have

$$1 - Ch^{2\beta} \le \frac{\eta_h}{\|\nabla e_h\|_{0,\Omega}} \le 1 + Ch^{2\beta}. \tag{2.14}$$

As a consequence, we should have

$$\eta_h^2(1 - Ch^{2\beta}) \le \lambda_h - \lambda \le \eta_h^2(1 + Ch^{2\beta}) \tag{2.15}$$

in Theorem 2.2 and

$$|\lambda_h^* - \lambda| \le Ch^{2(1+\beta)} \tag{2.16}$$

in Theorem 2.3. With the bisection mesh refinement, we have $\alpha = \infty$, $\beta = 1$, and therefore, $|\lambda_h^* - \lambda| \le Ch^4$.

To better understand this ultraconvergence (superconvergence with degree 2) phenomenon, let us reexamine

$$\begin{aligned}\eta_h^2 &= \|\mathrm{G_h}\, u_h - \nabla u + \nabla e_h\|_{0,\Omega}^2 \\ &= \|\nabla e_h\|_{0,\Omega}^2 + \|\mathrm{G_h}\, u_h - \nabla u\|_{0,\Omega}^2 + 2(\nabla e_h, \mathrm{G_h}\, u_h - \nabla u).\end{aligned}$$

Note that

$$\|\nabla e_h\|_{0,\Omega}^2 \le Ch^4, \qquad \|\mathrm{G_h}\, u_h - \nabla u\|_{0,\Omega}^2 \le Ch^{2(1+\beta)}.$$

Using the Hölder inequality, we obtain only

$$\frac{|(\nabla e_h, \mathrm{G_h}\, u_h - \nabla u)|}{\|\nabla e_h\|_{0,\Omega}^2} \le Ch^{\beta},$$

which does not take into account the cancellation in the integral. The real bound should be

$$\frac{|(\nabla e_h, \mathrm{G_h}\, u_h - \nabla u)|}{\|\nabla e_h\|_{0,\Omega}^2} \le Ch^{2\beta}. \tag{2.17}$$

Unfortunately, we are not able to prove (2.17) even for uniform meshes. Nevertheless, we are able to verify it indirectly via numerical experience. Toward this end, we define a quantity

$$\omega_{1,h} = |(\mathrm{G_h}\, u_h - \nabla u, \mathrm{G_h}\, u_h - \nabla u_h)|\,. \tag{2.18}$$

Note that

$$\begin{aligned}(\nabla e_h, \mathrm{G_h}\, u_h - \nabla u) &= (\mathrm{G_h}\, u_h - \nabla u, \nabla u - \mathrm{G_h}\, u_h) + (\mathrm{G_h}\, u_h - \nabla u, \mathrm{G_h}\, u_h - \nabla u_h)\\ &= (\mathrm{G_h}\, u_h - \nabla u, \mathrm{G_h}\, u_h - \nabla u_h) + O(h^{2(1+\beta)}).\end{aligned}$$

We see that $\omega_{1,h}$ provides an indirect way to verify (2.17).

Another way to understand the role of $\omega_{1,h}$ in the ultraconvergence phenomenon of $\lambda_h^* - \lambda$ is via the identity

$$\begin{aligned}\lambda_h^* - \lambda &= \|\nabla u - \mathrm{G_h}\, u_h\|_{0,\Omega}^2 + 2(\nabla u - \mathrm{G_h}\, u_h, \mathrm{G_h}\, u_h - \nabla u_h) - \lambda\|e_h\|_{0,\Omega}^2\\ &= \|\nabla u - \mathrm{G_h}\, u_h\|_{0,\Omega}^2 \pm 2\omega_{1,h} - \lambda\|e_h\|_{0,\Omega}^2.\end{aligned} \tag{2.19}$$

If the recovered gradient enjoys superconvergence, then the term with the lowest convergence rate in (2.19) is the mixed dot product whose absolute value is $2\omega_{1,h}$. Indeed, this term controls the convergence rate of $\lambda_h^* - \lambda$, and sometimes, it cancels the first term $\|\nabla u - \mathrm{G_h}\, u_h\|_{0,\Omega}^2$ in a favorable way. We will see this in our numerical test examples.

In addition, we shall numerically examine another quantity,

$$\omega_{2,h} = \left|\|\nabla e_h\|_{0,\Omega} + \eta_h + \frac{\lambda\|e_h\|_{L_2(\Omega)}^2}{2\eta_h}\right| \left|\|\nabla e_h\|_{0,\Omega} - \eta_h - \frac{\lambda\|e_h\|_{L_2(\Omega)}^2}{2\eta_h}\right|. \tag{2.20}$$

Note that both $\|\nabla e_h\|_{0,\Omega}$ and η_h are of order $O(h)$. From our numerical test examples in the next section, we will see that η_h cancels the major part of $\|\nabla e_h\|_{0,\Omega}$. The role of $\omega_{2,h}$ can be seen from the following identity:

$$\begin{aligned}\lambda_h^* - \lambda &= \|\nabla e_h\|_{0,\Omega}^2 - \eta_h^2\left(1 + \frac{\lambda\|e_h\|_{0,\Omega}^2}{\eta_h^2}\right)\\ &= \left(\|\nabla e_h\|_{0,\Omega} + \eta_h\sqrt{1 + \frac{\lambda\|e_h\|_{0,\Omega}^2}{\eta_h^2}}\right)\left(\|\nabla e_h\|_{0,\Omega} - \eta_h\sqrt{1 + \frac{\lambda\|e_h\|_{0,\Omega}^2}{\eta_h^2}}\right).\end{aligned} \tag{2.21}$$

Using the first two terms of the asymptotic expansion

$$\sqrt{1+\frac{\lambda\|e_h\|_{0,\Omega}^2}{\eta_h^2}} = 1+\frac{\lambda\|e_h\|_{0,\Omega}^2}{2\eta_h^2} - \frac{\lambda^2\|e_h\|_{0,\Omega}^4}{4\eta_h^4} + \frac{3\lambda^3\|e_h\|_{0,\Omega}^6}{8\eta_h^6} + \cdots,$$

we have

$$\lambda_h^* - \lambda = \omega_{2,h} + \text{h.o.t.}$$

3. Numerical examples. In this section we numerically test the proposed eigenvalue enhancement procedure. The examples are based on the two-dimensional eigenvalue problem of (2.1). For comparison purposes, we compute the recovered gradient by both the PPR and Zienkiewicz–Zhu SPR [14].

Let λ denote the minimum eigenvalue and u the corresponding eigenfunction. As we mentioned before, if $\|\mathrm{G_h}\, u_h - \nabla u\|_{0,\Omega}$ has any superconvergence, then $\lambda_h^* - \lambda$ also has superconvergence. Indeed, this superconvergence is attributed to either $\omega_{1,h}$ or $\omega_{2,h}$ defined in (2.18) and (2.20), respectively.

Example 1. As our first example, we consider the eigenvalue problem (2.2) with $\Omega = (0,1)\times(0,1)$. In this case, $\lambda = 2\pi^2$, and its corresponding eigenfunction is $u = 2\sin(\pi x)\sin(\pi y)$. We consider both structured and unstructured meshes.

In order to generate structured meshes, we divide Ω into $m \times m$ subsquares and triangulate each subsquare to one of the patterns shown in Figure 3.1. In consecutive iterations, $m = 16, 32, 64, 128$ for the criss cross pattern, while $m = 8, 16, 32, 64$ for the other three patterns. The numerical results for each of the four patterns are shown in Figures 3.2–3.5.

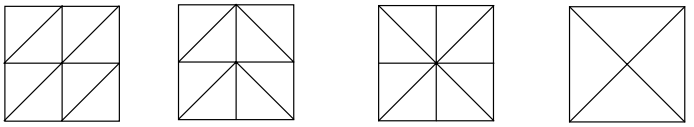
Let us have a closer look at Figure 3.2. First, the recovered gradient $\mathrm{G_h}\, u_h$ superconverges to ∇u. Accordingly, the convergence rate for $\omega_{1,h}$ is $O(h^3)$ if the Cauchy–Schwartz inequality is applied. However, this rate is $O(h^4)$ as depicted in Figure 3.2; i.e., $\omega_{1,h}$ is superconvergent. Consequently, all the terms in the identity for $\lambda_h^* - \lambda$ enjoy superconvergence. Hence, $\lambda_h^* - \lambda$ gets the ultraconvergence rate $O(h^4)$. Indeed, the same conclusion is true for the PPR in the other three cases.

As for the SPR, the situation is different. In the chevron pattern, the SPR-recovered gradient has no superconvergence [13]. Consequently, there is no superconvergence in $\lambda_h^* - \lambda$ although λ_h^* is more accurate than λ_h. In the Union Jack and the criss cross patterns, $\omega_{1,h}$ and consequently λ_h^*, do not gain from the superconvergence in the SPR-recovered gradient.

Next, we use the Delaunay triangulation (see Figure 3.6) to produce an unstructured initial mesh. In consecutive refinement, we construct a new mesh from the previous one by regular refinement, i.e., dividing each triangle into four congruent subtriangles by connecting its three edge centers. Figure 3.7 shows the convergence rates for various quantities of interest. Note that the recovered eigenvalue still has ultraconvergence. Surprisingly, the convergence rate is even better than the ones we have seen in the case of structured meshes, even though the recovered gradient does not have a full order recovery.

Example 2. So far, we have seen that $\omega_{1,h}$ and $\omega_{2,h}$ are almost the same. There are situations where the cancellations in $\omega_{2,h}$ are better, as we can see from the following example:

$$\begin{cases} -\triangle u = \lambda u & \text{in } \Omega = (0,1)^2, \\ u|_{x=0} = u|_{y=0} = u|_{y=1} = \partial_x u|_{x=1} = 0, \\ \|u\|_{L_2(\Omega)} = 1. \end{cases}$$



(a) The Regular pattern (b) The Chevron pattern (c) The Union Jack pattern (d) The Criss Cross pattern

FIG. 3.1. *The patterns used in constructing uniform meshes in* $\Omega = (0,1)^2$.

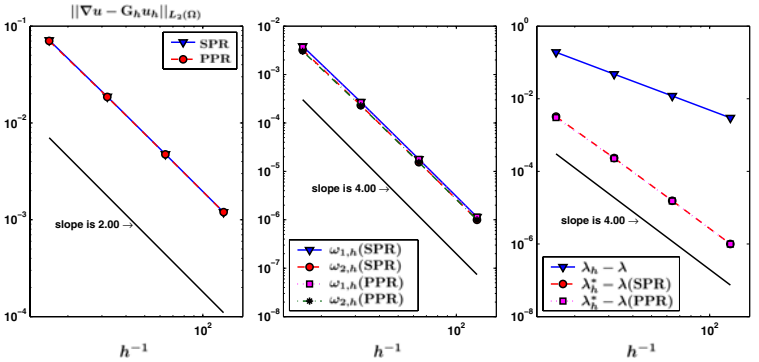


FIG. 3.2. *Example* 1 *results—the regular pattern case.*

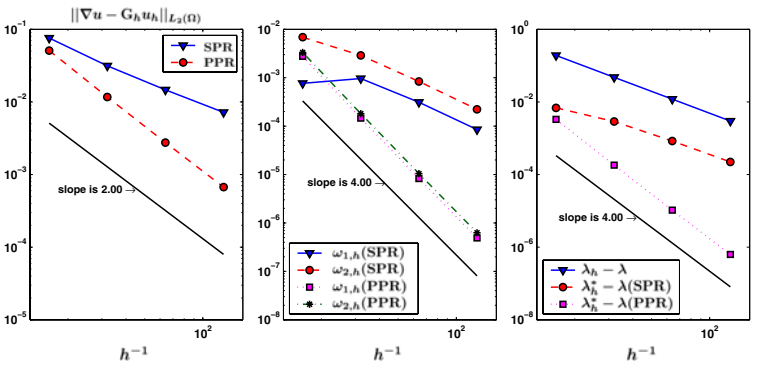


FIG. 3.3. *Example* 1 *results—the chevron pattern case.*

In this case,

$$\lambda = \frac{5\pi^2}{4}, \qquad u = 2\cos\left(\pi(x-1)/2\right)\sin \pi y.$$

Let us solve this problem using the sequence of meshes with an unstructured initial mesh employed in Example 1; see Figure 3.8 for the numerical results. As for the PPR, contrary to the SPR, the convergence rate of $\omega_{1,h}$ is less than the convergence rate of $\omega_{2,h}$ by at least 1. This means that η_h cancels not only the leading term in $\|\nabla e_h\|_{0,\Omega}$, but also the major part of $\frac{\lambda\|e_h\|_{0,\Omega}^2}{2\eta_h}$. This leads to a higher convergence rate in the recovered eigenvalue. Likewise, the dot product term in (2.19) cancels the first term in a favorable way since both terms are not as good as $O(h^4)$ and the enhanced eigenvalue converges at a rate $O(h^5)$!

3. In our previous two examples, the domain was a unit square. Let us

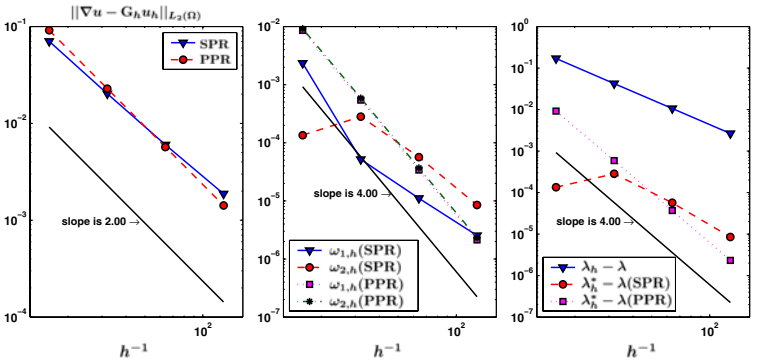


FIG. 3.4. *Example* 1 *results—the Union Jack pattern case.*

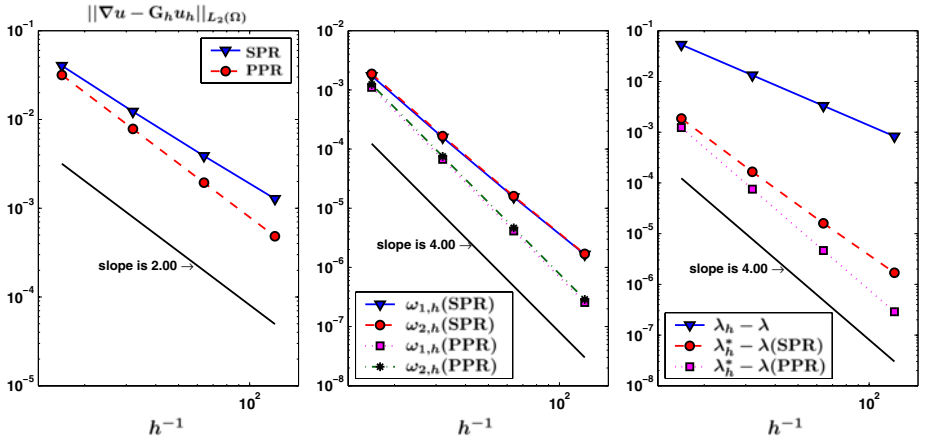


FIG. 3.5. *Example* 1 *results—the criss cross pattern case.*

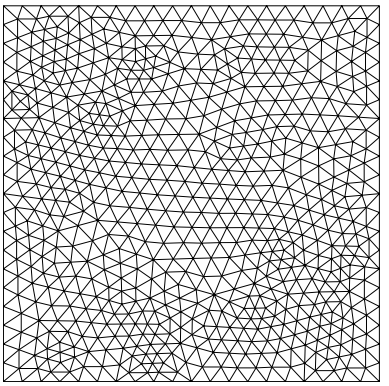


FIG. 3.6. *Example* 1*—the initial unstructured mesh case.*

change the domain to an equilateral triangle:

$$\Omega = (x, y) \in \mathbb{R}^2 : 0 < y < \sqrt{3}x, y > \sqrt{3}(1 - x).$$

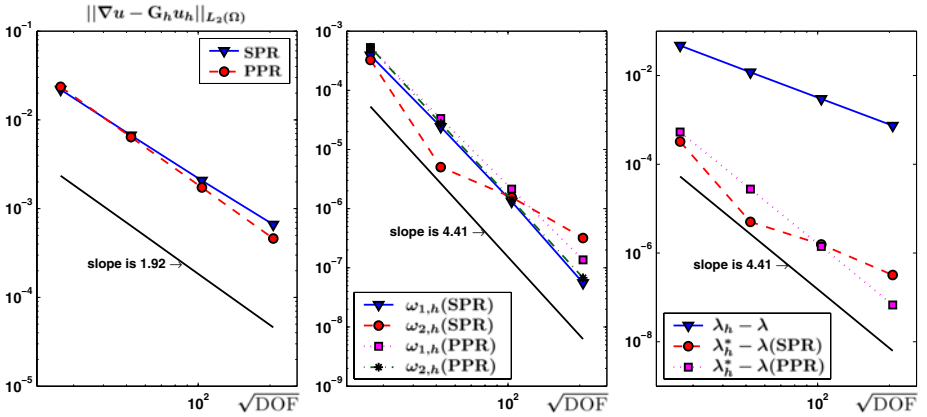


FIG. 3.7. *Example* 1 *results—the unstructured mesh case.*

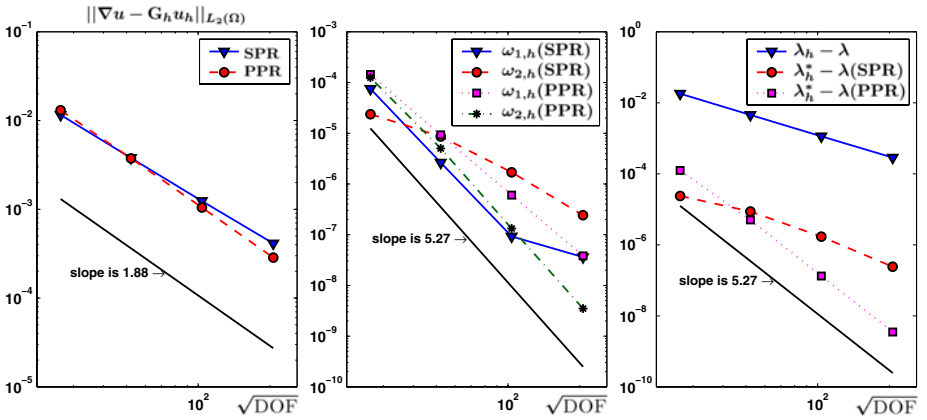


FIG. 3.8. *Example* 2 *results.*

With the zero boundary condition, $\lambda = 16\pi^2/3$ and

$$u = \frac{2\sqrt[4]{12}}{3}\left\{\sin\left(\frac{4\pi y}{\sqrt{3}}\right) + \sin\left(2\pi\left(x - \frac{y}{\sqrt{3}}\right)\right) + \sin\left(2\pi\left(1 - x - \frac{y}{\sqrt{3}}\right)\right)\right\}.$$

The structured meshes are generated by dividing Ω into $m(m+1)/2$ equilateral triangles with $m = 2^k$ for $k = 4, 5, 6, 7$. The initial mesh is shown in Figure 3.9. For unstructured initial mesh, we do as in Example 1 (see Figure 3.10.) The numerical results for both types of meshes are shown in Figures 3.11 and 3.12, respectively. Obviously, the results are similar to the ones in Example 1. Due to the high quality of the initially unstructured triangulation, there is not much difference between the convergence rates for uniform and unstructured meshes in this case.

EXAMPLE 4. In all previous examples, the eigenfunction u was analytic. Let us now consider the model problem (2.1) with

$$\Omega = \{(r, \theta) \in \mathbb{R}^2 : \ 0 < r < 1, \ \ 0 < \theta < 3\pi/2\},$$

in which case u has a corner singularity at $(0, 0)$. The smallest eigenvalue and its

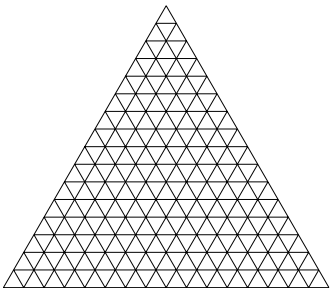


FIG. 3.9. *Example 3—the initial structured mesh.*

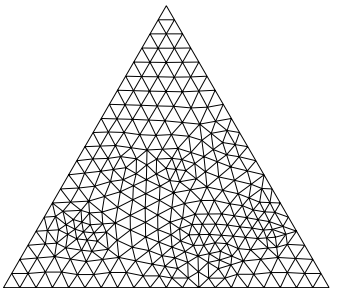


FIG. 3.10. *Example 3—the initial unstructured mesh.*

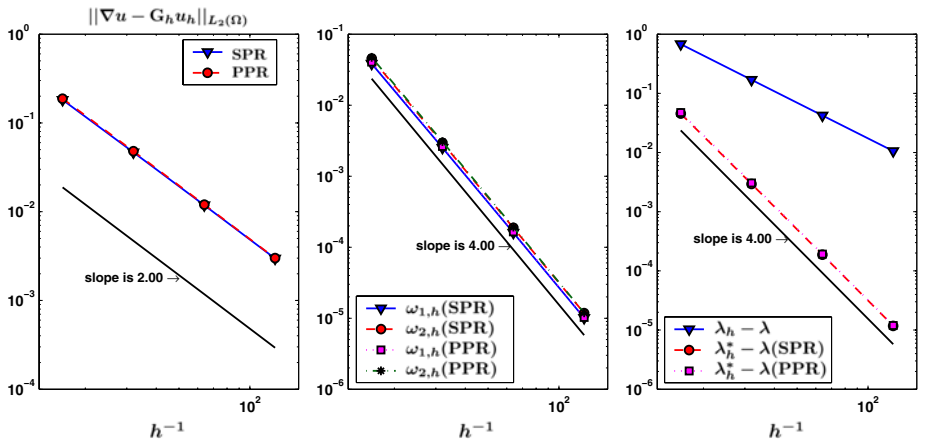


FIG. 3.11. *Example 3 results—the structured mesh case.*

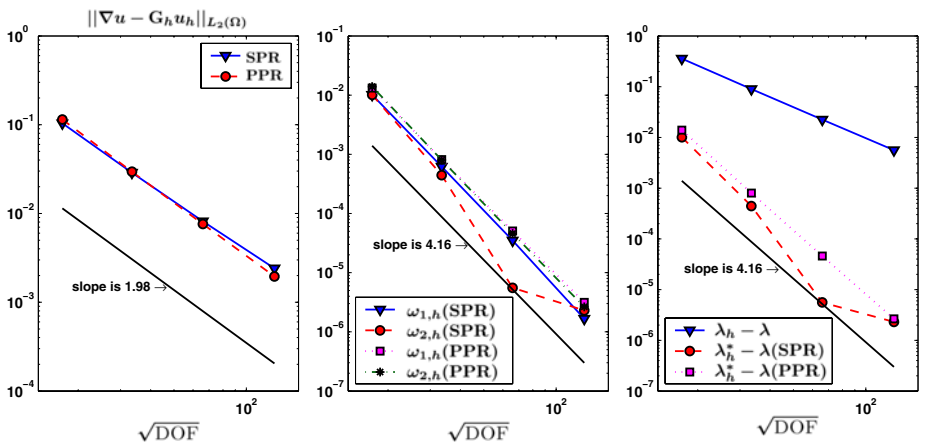


FIG. 3.12. *Example 3 results—the unstructured mesh case.*

eigenfunction for this problem are

$$u(r,\theta) = \frac{4J_{2/3}(\alpha r)}{\sqrt{6\pi}J_{5/3}(\alpha)}\sin(2\theta/3), \qquad \lambda_{\min} = \alpha^2, \qquad \alpha = \min\{x > 0 : J_{2/3}(x) = 0\},$$

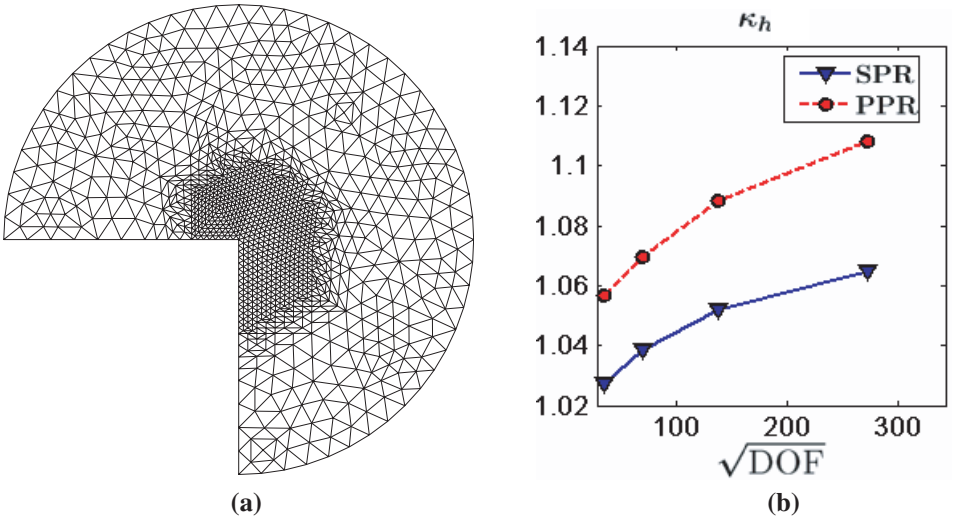


FIG. 3.13. *Example* 4*—the initial mesh* (a) *and the effectivity index* (b).

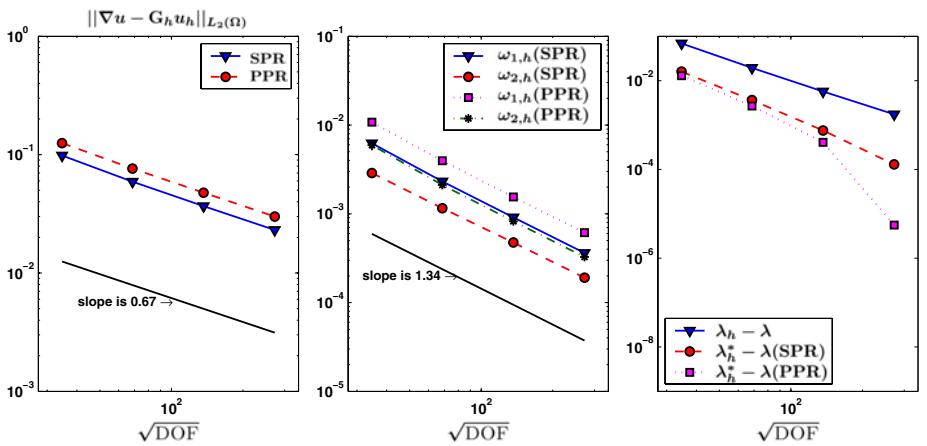


FIG. 3.14. *Example* 4 *results.*

where $J_{2/3}$ is the Bessel function of the first kind of order $2/3$. We start from an initial mesh shown in Figure 3.13(a). Note that this mesh is refined near $(0, 0)$ to avoid the pollution effect of the corner singularity. A sequence of meshes is produced by the regular refinement; i.e., by linking the three edge centers, each triangle is divided into four congruent subtriangles. From the numerical results in Figure 3.14, $\mathrm{G_h}\, u_h$ does not have superconvergence in both the PPR and the SPR. Nevertheless, η_h is still close to $\|\nabla e_h\|_{0,\Omega}$ (see the effectivity index $\kappa_h := \eta_h/\|\nabla e_h\|_{0,\Omega}$ in Figure 3.13(b)). From (2.10), the leading term in $\lambda_h^* - \lambda$ is $|\|\nabla e_h\|_{0,\Omega}^2 - \eta_h^2| = \|\nabla e_h\|_{0,\Omega}^2 |1 - \kappa_h^2|$. Since κ_h is very close to 1, the error in $\lambda_h^* - \lambda$ is a small fraction of the exact error $\lambda_h - \lambda$, which means that λ_h^* is more accurate than λ_h. This is true for both the PPR and the SPR.

4. Conclusion. We have shown that the PPR-recovered gradient $\mathrm{G_h}\, u_h$ can enhance the accuracy of λ_h if $\mathrm{G_h}\, u_h$ superconverges to ∇u (as we have seen in Examples 1, 2, and 3) or if the effectivity index of η_h is close to 1 (as we have seen in Example 4). Although the numerical results in this work are solely for the two-dimensional Poisson equation and the linear finite element method, the idea is nevertheless applicable to more general eigenvalue problems in the form

$$a(u, v) = \lambda b(u, v). \tag{4.1}$$

Here $a(\cdot, \cdot)$ and $b(\cdot, \cdot)$ are symmetric bilinear forms on some Hilbert space H and satisfy certain strong ellipticity and positivity conditions such that the following basic identity is valid (see [3, Lemma 9.1]):

$$\tilde{\lambda} - \lambda = \|u - w\|_a^2 - \lambda \|u - w\|_b^2,$$

where (u, λ) is an eigenpair of (4.1) with $\|u\|_b = 1$, $w \in H$ with $\|w\|_b = 1$, and $\tilde{\lambda} = a(w, w)$. In this setting, let $(\tilde{\lambda}, w) = (\lambda_h, u_h)$ be the finite element approximation of (u, λ). If we are able to improve the quantity involving u in the $\|\cdot\|_a$-norm via a recovery procedure, then $\|u - u_h\|_a^2$ may be substituted by a computable value to enhance the eigenvalue approximation. Further investigation along this line for the linear and the nonlinear problems is under way.

REFERENCES

[1] M. Ainsworth and J. T. Oden, *A Posteriori Error Estimation in Finite Element Analysis*, Wiley Interscience, New York, 2000.

[2] I. Babuška and T. Strouboulis, *The Finite Element Method and Its Reliability*, The Clarendon Press, Oxford University Press, New York, 2001.

[3] I. Babuška and J. E. Osborn, *Eigenvalue problems*, in Handbook of Numerical Analysis, Vol. II, Finite Element Methods (Part I), Handb. Numer. Anal. II, P. G. Ciarlet and J. L. Lions, eds., North–Holland, Amsterdam, 1991, pp. 641–787.

[4] F. Chatelin, *Spectral Approximations of Linear Operators*, Academic Press, New York, 1983.

[5] W. Dörfler and R. H. Nochetto, *Small data oscillation implies the saturation assumption*, Numer. Math., 91 (2002), pp. 1–12.

[6] A. Naga and Z. Zhang, *A posteriori error estimates based on the polynomial preserving recovery*, SIAM J. Numer. Anal., 42 (2004), pp. 1780–1800.

[7] A. Naga and Z. Zhang, *The polynomial-preserving recovery for higher order finite element methods in 2D and 3D*, Discrete Contin. Dyn. Syst. Ser. B, 5 (2005), pp. 769–798.

[8] L. Shen and A. Zhou, *A defect correction scheme for finite element eigenvalues with applications to quantum chemistry*, SIAM J. Sci. Comput., 28 (2006), pp. 321–338.

[9] J. Xu and Z. Zhang, *Analysis of recovery type a posteriori error estimators for mildly structured grids*, Math. Comp., 73 (2004), pp. 1139–1152.

[10] J. Xu and A. Zhou, *A two-grid discretization scheme for eigenvalue problems*, Math. Comp., 70 (2001), pp. 17–25.

[11] Y. Yang, *An Analysis of the Finite Element Method for Eigenvalue Problems*, Guizhou Press, Guiyang, China, 2004 (in Chinese).

[12] Z. Zhang, *Polynomial preserving gradient recovery and a posteriori estimate for bilinear element on irregular quadrilaterals*, Int. J. Numer. Anal. Model., 1 (2004), pp. 1–24.

[13] Z. Zhang and A. Naga, *A new finite element gradient recovery method: Superconvergence property*, SIAM J. Sci. Comput., 26 (2005), pp. 1192–1213.

[14] O. C. Zienkiewicz and J. Z. Zhu, *The superconvergent patch recovery and a posteriori error estimates.* I. *The recovery technique*, Internat. J. Numer. Methods Engrg., 33 (1992), pp. 1331–1364.

[15] O. C. Zienkiewicz and J. Z. Zhu, *The superconvergent patch recovery and a posteriori error estimates.* II. *Error estimates and adaptivity*, Internat. J. Numer. Methods Engrg., 33 (1992), pp. 1365–1382.

SIAM J. SCI. COMPUT.
Vol. 28, No. 4, pp. 1301–1317
© 2006 Society for Industrial and Applied Mathematics

CONTINUATION OF BIFURCATIONS IN PERIODIC DELAY-DIFFERENTIAL EQUATIONS USING CHARACTERISTIC MATRICES*

RÓBERT SZALAI†, GÁBOR STÉPÁN†, AND S. JOHN HOGAN‡

Abstract. In this paper we describe a method for continuing periodic solution bifurcations in periodic delay-differential equations. First, the notion of characteristic matrices of periodic orbits is introduced and equivalence with the monodromy operator is demonstrated. An alternative formulation of the characteristic matrix is given, which can be computed efficiently. Defining systems of bifurcations are constructed in a standard way, including the characteristic matrix and its derivatives. For following bifurcation curves in two parameters, the pseudo-arclength method is used combined with Newton iteration. Two test examples (an interrupted machining model and a traffic model with driver reaction time) conclude the paper. The algorithm has been implemented in the software tool PDDE-CONT.

Key words. delay-differential equations, periodic solutions, bifurcations, continuation

AMS subject classifications. 37M20, 34K13, 34K18

DOI. 10.1137/040618709

1. Introduction.

Continuation methods have widespread use in scientific applications described by ordinary differential equations (ODEs)(see, e.g., [6, 9, 21]). These methods are used to investigate invariant structures, such as fixed points and periodic orbits of dynamical systems over a range of parameter values. Usually only special or trivial solutions are available analytically and direct simulations only converge to stable (or unstable, if time reversed) solutions. Numerical continuation methods provide an ideal tool to find bifurcations of these solutions even after they become of saddle type. Similarly, bifurcations can also be continued by allowing additional parameters to vary.

However, when delay-differential equations (DDEs) [7, 17, 18] are considered, the resulting infinite-dimensional system presents far greater numerical challenges. Available continuation methods for bifurcations in DDEs are essentially limited to fixed points. The software DDE-BIFTOOL, developed by Engelborghs, Luzyania, and Roose [13], is capable of continuing bifurcations of fixed points. It can also continue periodic orbits, but not their bifurcations. To compute fixed point bifurcations, Engelborghs, Luzyania, and Roose [13] used the characteristic matrix [17], which can be computed analytically when the partial derivatives of the vector field are known. They exploited the fact that this matrix is singular at bifurcation points. In this paper we use a similar approach, but for periodic orbits.

The paper is organized as follows. In section 2, we introduce the characteristic matrix for periodic orbits. We derive its equivalence with the monodromy operator

*Received by the editors November 10, 2004; accepted for publication (in revised form) February 17, 2006; published electronically July 31, 2006.
http://www.siam.org/journals/sisc/28-4/61870.html

†Department of Applied Mechanics, Budapest University of Technology and Economics, H-1521, Budapest, Hungary (szalai@mm.bme.hu, stepan@mm.bme.hu). The work of the first author was supported by the Hungarian Eötvös scholarship and the Fulbright Scholarship. The work of the second author was supported by the Hungarian National Science Foundation under grant OTKA T043368.

‡Department of Engineering Mathematics, University of Bristol, BS8 1TR, Bristol, UK (s.j.hogan @bristol.ac.uk).

and show how its computational cost can be reduced to that of the solution of one system of linear equations. In section 3, continuation of periodic orbits is discussed using the pseudo-arclength method. Continuation of bifurcations is discussed in section 4. In section 5, we give implementation details of the algorithm. In section 6, two examples, an interrupted machining model and a traffic model, are analyzed.

2. Characteristic matrices. It is well known [7, 18] that linear autonomous delay equations admit the construction of a characteristic matrix. Thus, the task of finding the spectra of the infinite-dimensional problem is reduced to finding the roots of a nonlinear function, which is just the determinant of the characteristic matrix. Jordan chains [7] of the characteristic matrix at a spectral point can be used to obtain (generalized) eigenvectors of the infinitesimal generator.

For periodic DDEs, there is no such general characteristic matrix, except for special cases. For example, where the delay is an integer multiple of the period length and the partial derivatives of the vector field with respect to the delayed arguments are invertible, Verduyn Lunel [22] constructed a characteristic matrix. This case can also be found in [18], where an equivalence with the monodromy operator is derived (as in the autonomous case) based on the theory presented in [20].

Unfortunately, many important practical problems, such as our first test example of interrupted machining (see section 6.1), have noninvertible partial derivatives, so the case in [22] is not applicable. This difficulty can be overcome using Floquet theory [17], and a characteristic matrix can be constructed for integer delays [25]. Although this latter problem does not give equivalence with the monodromy operator, it serves as a starting point of our more general case.

In what follows, we generalize the results of Kaashoek and Verduyn Lunel [20] to periodic delay equations. Let x be a continuous function $x : I \subset \mathbb{R} \to \mathbb{R}^n$ and let $h > 0$ be a real number. Define $x_t \in C([-h, 0], \mathbb{R}^n)$ by $x_t(\theta) = x(t + \theta)$ and consider the following DDE:

$$x'(t) = f(t, x_t), \tag{2.1}$$

where $f : \mathbb{R}^n \times C([-h, 0], \mathbb{R}^n) \to \mathbb{R}^n$ is of class C^N, $N \geq 2$ in the second variable, and the prime denotes time-derivative. Further, assume that f is periodic in the first variable; thus $f(t, .) = f(t + \omega, .)$ with $\omega \geq h$. Consider a ω-periodic solution $v(t)$ of (2.1). Since f is smooth, we can linearize (2.1) at v in the form

$$x'(t) = \int_0^h d_\theta \eta^v(t, \theta) x(t - \theta), \tag{2.2}$$

where η^v is a matrix-valued function that is of bounded variation in θ and is associated with the periodic solution v. We can represent η uniquely if we take the space NBV, which consists of functions of bounded variations normalized such that $\eta(\theta) = 0$ for $\theta \in (-\infty, 0]$, right continuous on $(0, h)$ and constant on $[h, \infty)$. According to the basic theory of delay equations, (2.2) has a unique (forward) solution if we specify an initial function with $x_s = \phi$, $s \in \mathbb{R}$ fixed. The linearized equation can be written equivalently as

$$x'(t) = \int_0^{t-s} d_\theta \eta^v(t, \theta) x(t - \theta) + \int_{t-s}^{\infty} d_\theta \eta^v(t, \theta) \phi(t - s - \theta), \tag{2.3}$$

where the first term includes merely the unknown function, and the second term includes the initial function. Since the initial function is in the equation now, we can solve it uniquely by specifying an initial value $x(s) = \phi(0)$ only.

In what follows, we will consider a bigger space $X = L^\infty([-\omega, 0], \mathbb{R}^n)$, where $C([-\omega, 0], \mathbb{R}^n)$ is a closed subspace in X. This choice of X guarantees that there is a one-to-one relationship between solutions of (2.2) and the solutions given by our extended monodromy operator (defined below) by maintaining the compatibility with the duality framework for delay equations developed by Clement et al. [2, 3] (see also Diekmann et al. [7]). Let us define the operator

$$(\mathcal{A}(v,s)\varphi)(\theta) = \varphi'(\theta) - \int_0^{\omega+\theta} d_\vartheta \eta^v(s+\theta,\vartheta)\varphi(\theta-\vartheta) \tag{2.4}$$

with domain of definition $D(\mathcal{A}(v,s)) = \mathrm{Lip}([-\omega, 0], \mathbb{R}^n)$ and

$$(\mathcal{B}(v,s)\varphi)(\theta) = \int_{\omega+\theta}^{\omega} d_\vartheta \eta^v(s+\theta,\vartheta)\varphi(\omega+\theta-\vartheta) \tag{2.5}$$

on X. Note that $\overline{D(\mathcal{A}(v,s))} = C([-\omega, 0], \mathbb{R}^n)$ and $\mathcal{A}(v,s)$ has an n-dimensional kernel since, as we mentioned, we can solve $\mathcal{A}(v,s)\varphi = 0$ by specifying $\varphi(-\omega) = c$ for any $c \in \mathbb{R}^n$. To obtain the monodromy operator, we solve the equation $\mathcal{A}(v,s)\psi = \mathcal{B}(v,s)\varphi$ (which is just a reformulation of (2.3)) with the boundary condition $\psi(-\omega) = \varphi(0)$. Thus the monodromy operator becomes

$$U(s+\omega, s)\varphi = \psi. \tag{2.6}$$

The boundary condition can be eliminated by introducing the (even) bigger space $\hat{X} = \mathbb{R}^n \times X$, which includes the initial function ϕ and its initial value $\phi(0)$ as well. For ease of notation we use the following boundary value operators:

$$L : X \to \mathbb{R}^n,\ L\varphi = \varphi(-\omega),$$

$$M : X \to \mathbb{R}^n,\ M\varphi = \varphi(0).$$

These definitions make sense, since we consider only functions in the domain of $\mathcal{A}(v,s)$. Let

$$\hat{\mathcal{A}}(v,s) = \begin{pmatrix} 0 & L \\ 0 & \mathcal{A}(v,s) \end{pmatrix}$$

and

$$D(\hat{\mathcal{A}}(v,s)) = \{(c,\varphi) \in \hat{X} \mid \varphi \in D(\mathcal{A}(v,s)),\ c = M\varphi\}.$$

With this definition, it is easy to check that $\hat{\mathcal{A}}(v,s)$ is one-to-one and onto. $\hat{\mathcal{B}}(v,s)$ is defined similarly as

$$\hat{\mathcal{B}}(v,s) = \begin{pmatrix} I & 0 \\ 0 & \mathcal{B}(v,s) \end{pmatrix}.$$

The extended monodromy operator becomes $\hat{U}(s+\omega, s) = \hat{\mathcal{A}}^{-1}(v,s)\hat{\mathcal{B}}(v,s)$. Note that

$$\hat{\mathcal{A}}(v,s)\begin{pmatrix} \psi(0) \\ \psi \end{pmatrix} = \hat{\mathcal{B}}(v,s)\begin{pmatrix} \varphi(0) \\ \varphi \end{pmatrix} \Leftrightarrow \psi(-\omega) = \varphi(0), \quad \mathcal{A}\psi = \mathcal{B}\varphi$$

gives (2.6).

It can easily be proved that the eigenvalues of $\hat{U}(s+\omega,s)$ and $U(s+\omega,s)$ are the same (see [20, Proposition 1.1.]). Therefore we can equivalently analyze the spectrum

$$\sigma(\hat{U}(s+\omega,s)) := \mathbb{C}\backslash\{\lambda\in\mathbb{C}\,|\,(\lambda I-\hat{U}(s+\omega,s))^{-1}\text{ is a bounded linear operator}\}.$$

In case of delay equations, $\hat{U}(s+\omega,s)$ is compact: Thus, its spectrum consists of eigenvalues which have a cluster point at 0. Since we are not interested in 0, we can define $\mu=1/\lambda$ and analyze the equivalent problem

$$\lambda I-\hat{U}(s+\omega,s)=\frac{1}{\mu}\hat{\mathcal{A}}^{-1}(v,s)(\hat{\mathcal{A}}(v,s)-\mu\hat{\mathcal{B}}(v,s))$$

or, for the sake of simplicity, $\hat{\mathcal{A}}(v,s)-\mu\hat{\mathcal{B}}(v,s)$. In this latter case,

$$\sigma(\hat{U}(s+\omega,s))=\mathbb{C}\backslash\left\{\frac{1}{\mu}\in\mathbb{C}\,|\,(\hat{\mathcal{A}}(v,s)-\mu\hat{\mathcal{B}}(v,s))^{-1}\text{ is a bounded linear operator}\right\}.$$

From now on, we suppress the initial time s, the periodic solution v, or both in the notation of $\mathcal{A}(v,s)$, $\hat{\mathcal{A}}(v,s)$, $\mathcal{B}(v,s)$, and $\hat{\mathcal{B}}(v,s)$.

2.1. Equivalence. Kaashoek and Verduyn Lunel [20] derived the equivalence of unbounded linear operators, including infinitesimal generators of evolutionary systems and a characteristic matrix $\Delta(z)$, with some spectral parameter z under quite general assumptions. Here, we show how general periodic DDEs can fit into their framework, and extend the characteristic matrix constructions for equations with integer delays in [18, section 8.3].

In accordance with the previous section, we infer that operator $\mathcal{A}$ has the following properties:

1. $\mathcal{N}:=\ker(\mathcal{A})$ is finite-dimensional, $\dim(\mathcal{N})=n$, and $\mathcal{N}\neq\{0\}$.
2. The operator $\mathcal{A}$ has a restriction $\mathcal{A}_0: X\to X$ such that
   (a) $D(\mathcal{A})=\mathcal{N}\oplus D(\mathcal{A}_0)$,
   (b) $\Omega:=\{\mu\in\mathbb{C}\,|\,(\mu\mathcal{B}-\mathcal{A}_0)^{-1}\text{ is a bounded operator on }X\}\neq\emptyset$.

Also, denote the isomorphism from $\mathbb{R}^n$ onto $\mathcal{N}$ by $\jmath$. In what follows, we will use the graph norm $\|x\|_{\hat{\mathcal{A}}}=\|x\|+\|\hat{\mathcal{A}}x\|$ on $D(\hat{\mathcal{A}})$. Using this norm, $\hat{\mathcal{A}}$ becomes bounded and $\hat{X}_A:=D(\hat{\mathcal{A}})$ becomes a closed space (see Hille and Phillips [19]).

THEOREM 2.1.

$$F(\mu)(\hat{\mathcal{A}}-\mu\hat{\mathcal{B}})E(\mu)=\begin{pmatrix}\Delta(\mu)&0\\0&I_X\end{pmatrix},\quad \mu\in\Omega, \tag{2.7}$$

$E:\Omega\to\mathcal{L}(\hat{X},\hat{X}_A)$ $F:\Omega\to\mathcal{L}(\hat{X}_A,\hat{X})$

$$\Delta(\mu)=(\mu M-L)\left(I-(\mu\mathcal{B}-\mathcal{A}_0)^{-1}\mu\mathcal{B}\right)\jmath,\qquad \mu\in\Omega. \tag{2.8}$$

$$\begin{aligned}
E(\mu)\begin{pmatrix}c\\ \varphi\end{pmatrix}&=\begin{pmatrix}M(Q(\mu)\jmath c+(\mu\mathcal{B}-\mathcal{A}_0)^{-1}\varphi)\\ Q(\mu)\jmath c+(\mu\mathcal{B}-\mathcal{A}_0)^{-1}\varphi\end{pmatrix},\\
E^{-1}(\mu)\begin{pmatrix}M\psi\\ \psi\end{pmatrix}&=\begin{pmatrix}\jmath^{-1}Q(0)\psi\\ (\mu\mathcal{B}-\mathcal{A})\psi\end{pmatrix},\\
F(\mu)\begin{pmatrix}c\\ \varphi\end{pmatrix}&=\begin{pmatrix}c-(\mu M-L)(\mu\mathcal{B}-\mathcal{A}_0)^{-1}\varphi\\ \varphi\end{pmatrix},\\
F^{-1}(\mu)\begin{pmatrix}c\\ \varphi\end{pmatrix}&=\begin{pmatrix}c+(\mu M-L)(\mu\mathcal{B}-\mathcal{A}_0)^{-1}\varphi\\ \varphi\end{pmatrix},
\end{aligned}$$

$$Q(\mu)\varphi = \varphi - (\mu\mathcal{B} - \mathcal{A}_0)^{-1}(\mu\mathcal{B} - \mathcal{A})\varphi.$$

. The proof is similar to the one in [20, Theorem 3.1.], and therefore it is omitted. □

The proof is based on the observation that for all $\mu \in \Omega$, $(\mu\mathcal{B}-\mathcal{A})$ has an n-dimensional kernel and that the projection onto $\ker(\mu\mathcal{B}-\mathcal{A})$ is $Q(\mu)$. Since $\ker Q(\mu)$ is independent of μ, $Q(\mu_1) = Q(\mu_1)Q(\mu_2)$ and therefore $Q(\mu)\jmath$ is an isomorphism from $\mathbb{R}^n$ to $\ker(\mu\mathcal{B}-\mathcal{A})$. Applying $(MQ(\mu), Q(\mu))^T$ to $\mu\hat{\mathcal{B}} - \hat{\mathcal{A}}$, we find that the first equation is the characteristic matrix and the second equation is identically zero.

We need a characteristic matrix around geometrically simple inverse characteristic multipliers, so we make the following conjecture.

CONJECTURE 2.2. $(\mathcal{A}_0 - \mu\mathcal{B})$

(i) μ^{-1}

(ii) μ^{-1} ”

In continuation the inverse of $\mathcal{A}_0 - \mu\mathcal{B}$ is numerically computed using LU decomposition, so that the above conjecture can be checked by monitoring the smallest diagonal element of the U factor. In our examples in section 6 this minimal diagonal element always stayed away from zero.

First we expand $(\mathcal{A} - \mu\mathcal{B})\varphi = \psi$, $L\varphi = c$ using (2.4) and (2.5) into

$$\dot\varphi(\theta) = \int_0^{\omega+\theta} \mathrm{d}_\vartheta\eta(s+\theta,\vartheta)\varphi(\theta-\vartheta) + \mu\int_{\omega+\theta}^{\omega} \mathrm{d}_\vartheta\eta(s+\theta,\vartheta)\varphi(\omega+\theta-\vartheta) - \psi(\theta),$$

which contains our problem, when $L\varphi = 0$. After integration by parts and changing integration limits, we get that

$$\dot\varphi(\theta) = \eta(s+\theta,\omega+\theta)c + \int_{-\omega}^{\theta} \eta(s+\theta,\theta-\vartheta)\dot\varphi(\vartheta)\mathrm{d}\vartheta$$
$$+ \mu\left(\eta(s+\theta,\omega)\varphi(\theta) - \eta(s+\theta,\omega+\theta)\varphi(0) + \int_{\theta}^{0} \eta(s+\theta,\omega+\theta-\vartheta)\dot\varphi(\vartheta)\mathrm{d}\vartheta\right) - \psi(\theta).$$

Denote $\phi = \dot\varphi$; then the equation becomes

(2.9)
$$\phi(\theta) = \int_{-\omega}^{\theta} h_{s,\mu}(\theta,\vartheta)\phi(\vartheta)\mathrm{d}\vartheta + \int_{\theta}^{0} k_{s,\mu}(\theta,\vartheta)\phi(\vartheta)\mathrm{d}\vartheta + (\eta(s+\theta,\omega+\theta)+\eta(s-\omega,\omega))c - \psi(\theta),$$

where

$$h_{s,\mu}(\theta,\vartheta) = \eta(s+\theta,\theta-\vartheta) + \mu\eta(s+\theta,\omega) - \mu\eta(s+\theta,\omega+\theta),$$

$$k_{s,\mu}(\theta,\vartheta) = \mu\eta(s+\theta,\omega+\theta-\vartheta) - \mu\eta(s+\theta,\omega+\theta).$$

For the sake of simplicity define

$$(S\phi)(\theta) = \int_{-\omega}^{\theta} h_{s,\mu}(\theta,\vartheta)\phi(\vartheta)\mathrm{d}\vartheta + \int_{\theta}^{0} k_{s,\mu}(\theta,\vartheta)\phi(\vartheta)\mathrm{d}\vartheta,$$

which is a Fredholm operator. Hence $\ker(I-S)$ has the same dimension as $\ker(I-S)^*$, which implies that $(I-S)$ is onto if and only if it is one-to-one. From (2.9) it is clear that $(\mathcal{A}-\mu\mathcal{B})$ has at least an n-dimensional kernel. It is also clear that if the kernel has exactly n dimensions, then there is a unique ϕ satisfying (2.9) for $c=0$, which implies that

$$\varphi(\theta)=\int_{-\omega}^{\theta}\phi(\vartheta)\mathrm{d}\vartheta.$$

Hence it is enough to prove that $\dim\ker(\mathcal{A}-\mu\mathcal{B})=n$ under our assumptions.

When μ is not a spectrum point of $(\hat{\mathcal{A}}-\mu\hat{\mathcal{B}})$ we prove that $(\mathcal{A}_0-\mu\mathcal{B})$ is one-to-one by contradiction. Assume that $\dim\ker(\mathcal{A}-\mu\mathcal{B})=m$, $m>n$. This implies the existence of an isomorphism $p:\mathbb{R}^m\to\ker(\mathcal{A}-\mu\mathcal{B})$. In this case $(L-\mu M)p$ has an at least $(m-n)$-dimensional kernel, which contradicts our assumption that $(\hat{\mathcal{A}}-\mu\hat{\mathcal{B}})$ is one-to-one and proves that $\dim\ker(\mathcal{A}-\mu\mathcal{B})=n$.

At a geometrically simple inverse characteristic multiplier μ, by definition, $(\hat{\mathcal{A}}-\mu\hat{\mathcal{B}})$ has a one-dimensional kernel spanned by $(M\varphi_0,\varphi_0)$. In this case we either have $\dim\ker(\mathcal{A}-\mu\mathcal{B})=n$ or $n+1$ by a similar argument as above. We have to prove that the latter case does not occur generically. If $\dim\ker(\mathcal{A}-\mu\mathcal{B})=n+1$, we then have the isomorphism $p:\mathbb{R}^{n+1}\to\ker(\mathcal{A}-\mu\mathcal{B})$ so that $\dim\ker(L-\mu M)p=1$. However, this is not generic, because for any operator, say $F:X\to\mathbb{R}^n$, $\dim\ker Fp\geqslant 1$. In order to fix this we need $\dim\ker(\mathcal{A}-\mu\mathcal{B})=n$ and this must hold for almost every periodic orbit with a geometrically simple singularity of a parameter-dependent system within a parameter interval. If it is true, the same reasoning applies as above, and we infer that $(\mathcal{A}_0-\mu\mathcal{B})$ is one-to-one and onto.

In numerical applications, computing (2.8) is not efficient. Instead, we use the isomorphism $R(\mu):\mathbb{R}^n\to\ker(\mu\mathcal{B}-\mathcal{A})$,

$$R(\mu)c=\begin{pmatrix}0 & I\end{pmatrix}\begin{pmatrix}0 & -L\\ 0 & \mu\mathcal{B}-\mathcal{A}\end{pmatrix}^{-1}\begin{pmatrix}c\\ 0\end{pmatrix},$$

which is well defined if

$$D\left(\begin{pmatrix}0 & -L\\ 0 & \mu\mathcal{B}-\mathcal{A}\end{pmatrix}\right)=D(\hat{\mathcal{A}})$$

in light of Conjecture 2.2. In this case, the equivalent characteristic matrix becomes

$$\tilde{\Delta}(\mu)=(\mu M-L)R(\mu).$$

3. Continuation of periodic orbits. Define the following nonlinear operator:

$$G:\ X\times\mathbb{R}^p\to X,\quad G(\varphi,\lambda)(\theta)=x'(\theta)-f(\theta,\imath_\theta(\varphi),\lambda),$$

where $t\in[-T,0]$, $(\imath_t(\varphi))(\theta)=\varphi((t+\theta)\bmod\omega)$. The domain of G is $D(G(.,\lambda))=\{\varphi\in\mathrm{Lip}([-\omega,0],\mathbb{R}^n)|\,\varphi(-\omega)=\varphi(0)\}$. We can extend this operator in the same way as in the previous section:

$$\hat{G}((c,\varphi),\lambda)=\begin{pmatrix}L\varphi-c\\ G(\varphi,\lambda)\end{pmatrix},\quad(c,\varphi)\in D(\hat{G}(.,\lambda))=D(\hat{\mathcal{A}}).$$

Now we have the nonlinear equation for the periodic solutions of (2.1) in the form

$$\hat{G}((c,\varphi),\lambda)=\begin{pmatrix}0\\ 0\end{pmatrix}. \tag{3.1}$$

Let $(c_0, \varphi_0) = u_0$ be a solution of (3.1) at λ_0. Then the Frechet derivative of (3.1) is $\hat{\mathcal{A}}(\varphi_0) - \hat{\mathcal{B}}(\varphi_0)$. By the equivalence (2.7) we infer that $\hat{\mathcal{A}}(\varphi_0) - \hat{\mathcal{B}}(\varphi_0)$ is one-to-one and onto at a regular solution, when there is no $+1$ multiplier of the monodromy operator ($\det \Delta(1) \neq 0$). In this case, the solution u_0 can be continued according to the implicit function theorem.

3.1. Pseudo-arclength continuation. In our computations, we use the pseudo-arclength method to continue branches beyond limit points [10]. Here, we will use the following bilinear form:

$$\langle\, ,\, \rangle : \hat{X}_A \times \hat{X}_A \to \mathbb{R}, \quad \langle u, v\rangle = (Mv)^* Mu + \frac{1}{\omega}\int_{-\omega}^{0} u^*(\theta)v(\theta)d\theta,$$

where the * denotes the (conjugate) transpose. Let us denote a branch of solutions by $(u, \lambda) : I \subset \mathbb{R} \to \hat{X} \times \mathbb{R}$ parametrized with the arclength γ with respect to the norm $\|(u,\lambda)\| = \sqrt{\langle u, u\rangle} + |\lambda|$. Consider a solution $(u_0, \lambda_0) = (u(0), \lambda(0))$, which we want to continue in λ. Since u_0 is a regular solution, we can obtain the tangents $\dot{u}_0$, $\dot{\lambda}_0$ from the equation

$$(\hat{\mathcal{A}}(u_0) - \hat{\mathcal{B}}(u_0))\dot{u}_0 + D_\lambda \hat{G}(u_0, \lambda)\dot{\lambda}_0 = 0.$$

Further, we normalize the tangents such that $\|(\dot{u}_0, \dot{\lambda}_0)\| = 1$. With these tangents we predict the next solution point on the branch of solutions $u_1^{(0)} = d\gamma\, \dot{u}_0$, $\lambda_1^{(0)} = d\gamma\, \dot{\lambda}_0$, where $d\gamma$ is the step size. Newton iteration of the correction step proceeds by solving

$$\begin{pmatrix} \hat{\mathcal{A}}(u_1^{(\nu)}) - \hat{\mathcal{B}}(u_1^{(\nu)}) & D_\lambda \hat{G}(u_1^{(\nu)}, \lambda_1^{(\nu)}) \\ \langle \dot{u}_0, \, . \, \rangle & \dot{\lambda}_0 \end{pmatrix} \begin{pmatrix} \Delta u \\ \Delta\lambda \end{pmatrix} = \begin{pmatrix} -\hat{G}(u_1^{(\nu)}, \lambda_1^{(\nu)}) \\ d\gamma - \langle \dot{u}_0, u_1^{(\nu)} - u_0\rangle - (\lambda_1^{(\nu)} - \lambda_0)\dot{\lambda}_0 \end{pmatrix} \tag{3.2}$$

and updating $u_1^{(\nu+1)} = u_1^{(\nu)} + \Delta u$, $\lambda_1^{(\nu+1)} = \lambda_1^{(\nu)} + \Delta\lambda$. According to bordering theorems (see, e.g., [10]), the coefficient operator in (3.2) is also one-to-one and onto, since $\hat{\mathcal{A}}(u_0) - \hat{\mathcal{B}}(u_0)$ is one-to-one and onto and $\dot{\lambda}_0 \neq 0$ at a regular solution. The same is true in the case of a quadratic fold when $D_\lambda \hat{G}(u_0, \lambda_0) \notin \mathcal{R}(\hat{\mathcal{A}}(u_0) - \hat{\mathcal{B}}(u_0))$.

4. Continuation of bifurcations. In this section we reformulate classical continuation theorems of codimension-one bifurcations for periodic DDEs. Since we have the equivalence with the finite-dimensional characteristic matrix, these classical theorems can be applied almost directly. In order to compute the Jacobians, we will need derivatives of the characteristic matrix with respect to u, λ, and the spectral parameter μ. For the sake of simplicity (and computational efficiency) we use $\tilde{\Delta}(\mu)$. In general, the derivative of the inverse of an operator-valued function A is

$$(D_x A^{-1}(x))(y) = -A^{-1}(x)(D_x A(x))(A^{-1}(x), y). \tag{4.1}$$

Using this relation

$$\begin{aligned} D_u\tilde{\Delta}(\mu)c = &-(\mu M - L)\begin{pmatrix} 0 & -L \\ 0 & \mu\mathcal{B} - \mathcal{A} \end{pmatrix}^{-1} \\ &\times D_u \begin{pmatrix} 0 & -L \\ 0 & \mu\mathcal{B}(u) - \mathcal{A}(u) \end{pmatrix} \left(\begin{pmatrix} 0 & -L \\ 0 & \mu\mathcal{B} - \mathcal{A} \end{pmatrix}^{-1} \begin{pmatrix} c \\ 0 \end{pmatrix}, \, . \, \right), \end{aligned}$$

$$D_\lambda\tilde{\Delta}(\mu)c = -(\mu M - L)\begin{pmatrix} 0 & -L \\ 0 & \mu\mathcal{B}-\mathcal{A}\end{pmatrix}^{-1} \times D_\lambda\begin{pmatrix} 0 & -L \\ 0 & \mu\mathcal{B}(u)-\mathcal{A}(u)\end{pmatrix}\begin{pmatrix} 0 & -L \\ 0 & \mu\mathcal{B}-\mathcal{A}\end{pmatrix}^{-1}\begin{pmatrix} c \\ 0\end{pmatrix},$$

and

$$D_\mu\tilde{\Delta}(\mu)c = M\,R(\mu)c - (\mu M - L)\begin{pmatrix} 0 & -L \\ 0 & \mu\mathcal{B}-\mathcal{A}\end{pmatrix}^{-1} \times \begin{pmatrix} 0 & 0 \\ 0 & \mathcal{B}(u)\end{pmatrix}\begin{pmatrix} 0 & -L \\ 0 & \mu\mathcal{B}-\mathcal{A}\end{pmatrix}^{-1}\begin{pmatrix} c \\ 0\end{pmatrix}. \tag{4.2}$$

Note that in the above formulas, it is always the same operator that is inverted.

4.1. Fold bifurcation. In case of a fold bifurcation there is a $+1$ eigenvalue of the monodromy operator, which is equivalent to $\det\Delta(1) = 0$. This implies that we have unit vectors p and q, ($p^*p = 1$, $q^*q = 1$) such that $p^*\Delta(1) = 0$ and $\Delta(1)q = 0$. The kernel of $\hat{\mathcal{A}}-\hat{\mathcal{B}}$ and $(\hat{\mathcal{A}}-\hat{\mathcal{B}})^*$ is spanned by $\varphi = E(1)(q,0)^T$ and $\psi = F^*(1)(p,0)^T$, respectively. The defining system is (see, e.g., Beyn et al. [1])

$$\begin{aligned}\hat{G}(u,\lambda) &= 0,\\ \tilde{\Delta}(1)q &= 0,\\ q_0^*q - 1 &= 0.\end{aligned}$$

The Jacobian with respect to (u,q,λ) is

$$\begin{pmatrix} \hat{\mathcal{A}}(u_0)-\hat{\mathcal{B}}(u_0) & 0 & D_\lambda\hat{G}(u_0,\lambda_0) \\ D_u\tilde{\Delta}(1)q_0 & \tilde{\Delta}(1) & D_\lambda\tilde{\Delta}(1)q_0 \\ 0 & q_0^* & 0\end{pmatrix}. \tag{4.3}$$

PROPOSITION 4.1. [illegible] (4.3) [illegible]. The proof is the same as for algebraic systems and is therefore omitted (see Doedel, Keller, and Kernévez [10, section 2.3.]). □

4.2. Period doubling. Similarly, the defining system is

$$\begin{aligned}\hat{G}(u,\lambda) &= 0,\\ \tilde{\Delta}(-1)q &= 0,\\ q_0^*q - 1 &= 0,\end{aligned}$$

and the Jacobian with respect to (u,q,λ) yields

$$\begin{pmatrix} \hat{\mathcal{A}}(u_0)-\hat{\mathcal{B}}(u_0) & 0 & D_\lambda\hat{G}(u_0,\lambda_0) \\ D_u\tilde{\Delta}(-1)q_0 & \tilde{\Delta}(-1) & D_\lambda\tilde{\Delta}(-1)q_0 \\ 0 & q_0^* & 0\end{pmatrix}. \tag{4.4}$$

4.3. Neimark–Sacker bifurcation. The defining system is

$$\begin{aligned}\hat{G}(u,\lambda) &= 0,\\ \tilde{\Delta}(e^{i\alpha})q &= 0,\\ q_0^*q - 1 &= 0,\end{aligned}$$

and its Jacobian with respect to (u, q, α, λ) becomes

$$\begin{pmatrix} \hat{\mathcal{A}}(u_0) - \hat{\mathcal{B}}(u_0) & 0 & 0 & D_\lambda \hat{G}(u_0, \lambda_0) \\ D_u \tilde{\Delta}(e^{i\alpha_0}) q_0 & \tilde{\Delta}(e^{i\alpha_0}) & D_\alpha \tilde{\Delta}(e^{i\alpha_0}) q_0 & D_\lambda \tilde{\Delta}(e^{i\alpha_0}) q_0 \\ 0 & q_0^* & 0 & 0 \end{pmatrix}. \tag{4.5}$$

Here we use complex values in the last two rows of the Jacobian, which can be expanded to reals, and then use the same argument to prove the regularity as in [10, section 2.3.].

4.4. Remark on the autonomous case. The presented method can easily be extended for autonomous equations by adding a phase condition to the defining systems and the period to the list of unknown variables [12]. In the case of a fold bifurcation in autonomous systems, the characteristic multiplier $+1$ has algebraic multiplicity two; therefore, it is not sufficient to use $\Delta(1)q$, but by the equivalence of Jordan chains [20],

$$\begin{aligned} \tilde{\Delta}(1) q_1 &= 0, \\ D_\mu \tilde{\Delta}(1) q_1 + \tilde{\Delta}(1) q_2 &= 0. \end{aligned} \tag{4.6}$$

The size of this system of equations can be reduced by computing q_1 and $D_\mu \Delta(1) q_1$. The eigenfunction ϕ_1 of the trivial multiplier 1 is always the derivative of the solution, which equals the right-hand side of the equation $\phi_1(\theta) = f(\iota_\theta(u_0), \lambda)$. Further, from the analysis above, we infer that $q_1 = M\phi_1 = f(\iota_{(0)}(u_0), \lambda)$. On the other hand, $D_\mu \Delta(1) q_1$ can be computed according to (4.2) as

$$D_\mu \tilde{\Delta}(\mu) q_1 = f(i_{(0)}(u), \lambda) - M \begin{pmatrix} 0 & -L \\ 0 & \mathcal{B} - \mathcal{A} \end{pmatrix}^{-1} \begin{pmatrix} 0 \\ \mathcal{B} f(i_{(.)}(u), \lambda) \end{pmatrix}.$$

In general, (4.6) defines a one-dimensional kernel space at a regular solution, which corresponds to q_1. However, if there is an algebraically double $+1$ multiplier, our system (4.6) has a two-dimensional kernel space. To eliminate the trivial kernel we use bordering, which reduces (4.6) to

$$\begin{pmatrix} D_\mu \tilde{\Delta}(\mu) q_1 & \tilde{\Delta}(\mu) \\ 0 & q_1^* \end{pmatrix} \tilde{q}_2 = 0,$$

where $\tilde{q}_2$ is a new variable, which contains q_2 from its second element. We remark that in [8], test functionals for folds in ODEs are constructed in a similar way.

5. Implementation. In this section, we summarize the numerical methods used in our continuation software that uses orthogonal collocation [5] to discretize the operators $\mathcal{A}$ and $\mathcal{B}$ of equations with discrete fixed delays. Thus, (2.1) can be written as

$$x'(\tilde{t}) = \tilde{f}(\tilde{t}, x(\tilde{t} - \tilde{\tau}_1), \ldots, x(\tilde{t} - \tilde{\tau}_m)),$$

where $\tilde{f} : \mathbb{T}^1 \times \mathbb{R}^n \times \cdots \times \mathbb{R}^n \to \mathbb{R}^n$ and $0 \le \tilde{\tau}_1 < \tilde{\tau}_2 < \cdots < \tilde{\tau}_m \le \omega$. By rescaling the time with $\tilde{t} = \omega t$ and defining $f(t, x_1, \ldots, x_m) = \tilde{f}(\omega t, x_1, \ldots, x_m)$, our equation becomes

$$x'(t) = \omega f(t, x(t - \tau_1), \ldots, x(t - \tau_m)).$$

Its variational system at v is

$$x'(t) = \sum_{i=1}^{m} D_{x_i} f(t, v(t-\tau_1), \dots, v(t-\tau_m)) x(t-\tau_i).$$

Also, assume that the initial time $s = 0$. Let A_i, $B_i : X \times [-1,0] \to \mathbb{R}^{n\times n}$ be defined by

$$A_i(v,\theta) = \omega H(1+\theta-\tau_i) D_{x_i} f(\theta, v((\theta-\tau_1) \bmod 1), \dots, v((\theta-\tau_m) \bmod 1)),$$
$$B_i(v,\theta) = \omega H(\tau_i-1-\theta) D_{x_i} f(\theta, v((\theta-\tau_1) \bmod 1), \dots, v((\theta-\tau_m) \bmod 1)),$$

where mod maps to $[-1,0]$ and H is the Heaviside function. Then (2.4) and (2.5) become

$$(\mathcal{A}(v)\varphi)(\theta) = \varphi'(\theta) - \sum_{i=1}^{m} A_i(\theta)\varphi(\theta-\tau_i),$$
$$(\mathcal{B}(v)\varphi)(\theta) = \sum_{i=1}^{m} B_i(\theta)\varphi(1+\theta-\tau_i).$$

5.1. Collocation. Here, we summarize the main steps of the method; its detailed discussion can be found in [12]. We approximate the solution φ on $\mathfrak{n}$ intervals by degree $\mathfrak{m}$ polynomials. The method requires the space X to be $C^{\mathfrak{m}}([-1,0],\mathbb{R}^n)$, and consequently $D(\mathcal{A}(v)) = C^{\mathfrak{m}+1}([-1,0],\mathbb{R}^n)$ and f is also $C^{\mathfrak{m}}$-smooth in t. The mesh is represented by the intervals $[\theta_i, \theta_{i+1}]$, $i = 0, \dots, \mathfrak{n}-1$ with $-1 = \theta_0 < \theta_1 < \cdots < \theta_{\mathfrak{n}} = 0$. Also denote the length of an interval by $h_i := \theta_{i+1} - \theta_i$. The approximate solution can be written on the ith subinterval as

$$\tilde{\varphi}(\theta) = \sum_{j=0}^{\mathfrak{m}} \varphi(\theta_{i+\frac{j}{\mathfrak{m}}}) P_{i,j}(\theta),$$

where

$$P_{i,j}(\theta) = \prod_{r=0, r\neq j}^{\mathfrak{m}} \frac{\theta - \theta_{i+\frac{r}{\mathfrak{m}}}}{\theta_{i+\frac{j}{\mathfrak{m}}} - \theta_{i+\frac{r}{\mathfrak{m}}}}$$

and $\theta_{i+\frac{j}{\mathfrak{m}}} := \theta_i + j\frac{h_i}{\mathfrak{m}}$. Let c_i, $i = 1, \dots, \mathfrak{m}-1$, be the roots of the degree $\mathfrak{m}$ Legendre polynomial scaled from $[-1,1]$ to $[0,1]$. Then define the collocation points by $c_{j,i} = \theta_j + h_j c_i$. The discretized version of operator $\mathcal{A}$ becomes

$$(\mathcal{A}_d(v)\tilde{\varphi})(c_{j,i}) = \tilde{\varphi}'(c_{j,i}) - \sum_{i=1}^{m} A_i(c_{j,i})\tilde{\varphi}(c_{j,i}-\tau_i). \tag{5.1}$$

Operator $\mathcal{B}$ is discretized similarly. Note that (5.1) is an operator from $\mathbb{R}^{(\mathfrak{m}\times\mathfrak{n}-1)\times n}$ to $\mathbb{R}^{\mathfrak{m}\times\mathfrak{n}\times n}$, so it has (at least) an n-dimesional kernel just as its abstract counterpart (2.4) has. The discretized version of $\hat{\mathcal{A}}$ acting on $(M\varphi, \varphi)$ can be represented by a square matrix by placing the $-L\varphi$ vector into the first n rows, which is of full rank at a regular solution v according to [11, Theorem 3.1] when $|h| = \max_{i=0}^{n-1} h_i$ is sufficiently small. Moreover, if we choose the space $X = C^{\mathfrak{m}}([-1,0],\mathbb{R}^n)$ as described above, then there exist a $\delta > 0$ and $C > 0$ such that the solutions of $\hat{\mathcal{A}}\psi = \hat{\mathcal{B}}\varphi$ and $\hat{\mathcal{A}}_d\tilde{\psi} = \hat{\mathcal{B}}_d\tilde{\varphi}$ satisfy

$$\max_{\theta\in[-1,0]} |\psi(\theta) - \tilde{\psi}(\theta)| \leq C|h|^{\mathfrak{m}} \tag{5.2}$$

if $|h| \in (0, \delta]$ (see Engelborghs and Doedel [11, Theorem 3.3]). Although superconvergence is lost for delay equations, some improvement can be achieved over (5.2) by using modified collocation schemes (see [11, section 4] and [12]).

5.2. Solving bordered linear systems. Equations (3.2), (4.3), (4.4), and (4.5) contain the operator $\hat{\mathcal{A}} - \hat{\mathcal{B}}$, functionals such as $\langle \dot{u}_0, . \rangle$ on $\hat{X}$, vectors such as $D_\lambda \hat{G}(u_0, \lambda_0)$ in $\hat{X}$, and also the finite-dimensional characteristic matrix. From a computational point of view, it is better to factorize the discretized counterparts of these components separately rather than copying them into one big matrix. Usually, the discretized infinite-dimensional part has a special sparsity structure, which can be exploited in computations. Other reasons for bordering methods might be the reuse of code and avoiding the frequent copying of large data structures. In our code, we used methods developed by Govaerts [14, 15] and Govaerts and Pryce [16]. These methods have the advantage over classical bordering methods [10, 1] in that they do not require the calculation of the kernel of singular matrices for the computation and can be used with arbitrary stable solvers. This feature makes it possible to use the same routine for sparse (e.g., $\hat{\mathcal{A}}_d - \hat{\mathcal{B}}_d$ with discrete delays) and dense (e.g., $\hat{\mathcal{A}}_d - \hat{\mathcal{B}}_d$ with distributed delays) matrices, leaving space for future improvements.

The method of mixed block elimination (BEM) [14] is used in factorizing the discretized counterpart of the Jacobian in (3.2). For solving equations of the form

$$\begin{pmatrix} A & b \\ c^* & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \alpha \\ \beta \end{pmatrix},$$

the method requires a solution with A and one with A^*. Note that for this solution A is factorized only once. For the bigger problem

$$\begin{pmatrix} A & 0 & b \\ C & D & e \\ f^* & g^* & h \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} \alpha \\ \beta \\ \delta \end{pmatrix},$$

which appears in (4.3) and (4.4), we apply the generalized mixed block elimination (GMBE) [14] technique, which requires one solution with D^T and two BEM steps with A and D, respectively. In case of systems augmented with wider borders, BEMW [14] is used, which is the iterative form of the recursive version of BEM. The wider borders occur in (4.5) and (4.3), and also in (4.4) when they are further augmented with derivatives and tangent vectors in the two-parameter continuation context similarly to (3.2). These kinds of equations can be solved using GMBE, but BEM must be replaced with BEMW.

Factorization of sparse matrices was carried out using UMFPACK,[1] which implements the unsymmetric multifrontal method of direct LU factorization (see Davies [4]). For (nearly) optimal performance we construct the matrices $\hat{\mathcal{A}}_d - \hat{\mathcal{B}}_d$ directly in compressed row form instead of the default compressed column form. After factorization, we interchange the solution methods for the matrix and its transpose to get the correct results. For dense matrices, we use the standard LAPACK routine GESVX.

6. Examples. Our first example is a forced system which models an interrupted machining process. In this example, when considering period-one solutions, the delay equals the period. However, this is not the case for period-two solutions, where

[1] We remark that this package is incorporated into MATLAB as well.

there are two delays, the smallest being half of the period length; thus the characteristic matrix definition of Verduyn Lunel [22] is not applicable and we need our general construction. The second example is a traffic model, which demonstrates that our algorithm can handle arbitrary (varying) delays. This second example, being autonomous, also demonstrates the fold continuation in our remark in section 4.4.

6.1. Interrupted machining. Here, we analyze bifurcations of period-two orbits in an interrupted machining model. To this end, we consider an extended version of the equation of motion in [25] in the form

(6.1)
$$\begin{aligned} x''(t)+2\zeta x'(t)+x(t) = {} & wg(t)H\left(1+x(t-2T)-x(t-T)\right)F_c\left(1+x(t-T)-x(t)\right) \\ & + wg(t)H\left(1+x(t-T)-x(t-2T)\right)F_c\left(2+x(t-2T)-x(t)\right), \end{aligned}$$

where H is the Heaviside function,

$$g(t) = \begin{cases} 0 & \text{if } \exists\, k:\ kT \le t < (k+1-\rho)T \\ 1 & \text{if } \exists\, k:\ (k-\rho)T \le t < kT \end{cases}, \quad k \in \mathbb{Z},$$

and

$$F_c(x) = H(x)\left(\frac{120}{59}x - \frac{48}{59}x^2 + \frac{35}{177}x^3\right).$$

This cutting force function F_c is a polynomial approximation of the common three-quarter rule $F_c(x) \approx 4/3\, x^{3/4}$ (see [27]). Also, $\zeta = 0.015$ is the relative damping, and $\rho = 0.1$ is the ratio between the period length T and the time spent with cutting. Equation (6.1) has a unique T-periodic solution, which is described by the ODE

$$x''(t) + 2\zeta x'(t) + x(t) = wg(t)F_c(1).$$

In a quite large neighborhood of this periodic solution, (6.1) simplifies to

$$x''(t) + 2\zeta x'(t) + x(t) = wg(t)F_c\left(1+x(t-T)-x(t)\right), \tag{6.2}$$

while the second line of (6.1) is turned on if the tool did not cut the workpiece in the previous period. Its variational equation around the T-periodic solution is

$$x''(t) + 2\zeta x'(t) + x(t) = wg(t)\left(x(t-T)-x(t)\right).$$

The stability of the above equation was analyzed analytically in [25] in great detail. The stability chart is shown in Figure 1. We analyze the lowest lens at $T \approx 15$. The bifurcation diagram at $T = 14.75$ can be seen in Figure 2. The first period doubling bifurcation at $w \approx 0.6629$ is subcritical. The arising unstable period-two orbit folds back and then loses its stability through a torus bifurcation. The other period doubling bifurcation is supercritical. On the stable branch there is a torus bifurcation window. This branch folds back, too, and becomes unstable.

The bifurcations in Figure 2 are continued in two parameters using our method. First, the stability boundaries of (6.2) are obtained, which coincides with the semi-analytical results of [25]. The period-T orbits are also checked analytically. In order to avoid difficulties arising from the nonsmoothness of (6.1), we substitute the Heaviside function $H(x)$ by $(1+\tanh(C\,x))/2$ with $C = 30$. Choosing bigger values for C is not reasonable because it only improves the results within the line thickness but makes

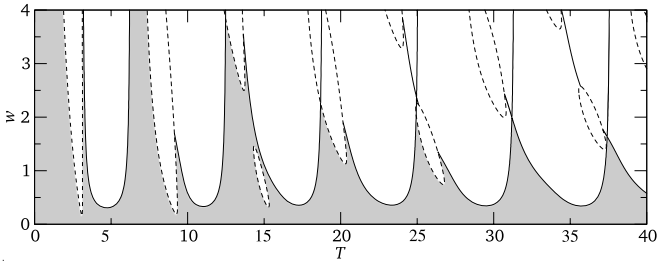


FIG. 1. *Stability chart for the interrupted machining model showing period doubling boundaries (dashed lines) and Neimark–Sacker boundaries (continuous lines). In the gray region the T-periodic solution is stable ($\zeta = 0.015$).*

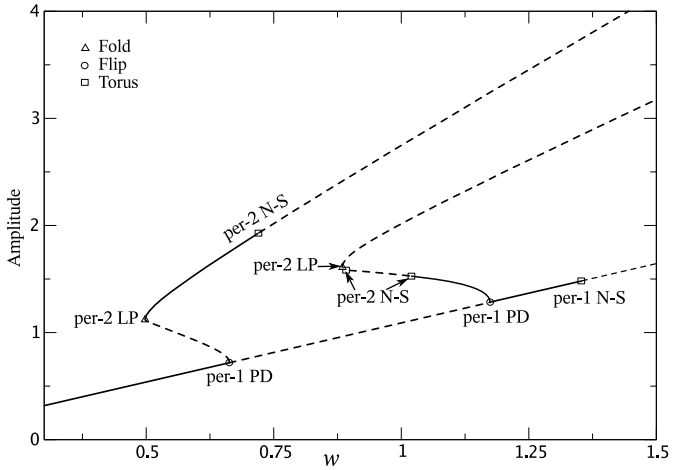


FIG. 2. *This bifurcation diagram shows the period-two orbits emanating from the period-one solution at the stability boundary in the vertical cross section of a lens at $T = 14.75$ in Figure 1. Continuous lines denote stable solutions, while dashed lines correspond to unstable solutions.*

the computation more difficult due to derivatives tending to infinity. The period-two orbits arising at the lens in the space of two parameters extended with the solution amplitude form a highly distorted tube-like object. Figure 3 shows the computed bifurcation curves. The lower continuous curve corresponds to the fold bifurcation of orbits arising at the lower part of the lens. The other continuous curve shows the fold of the upper solution branch in Figure 2. It can be seen that there are two cusp points on the bifurcation curve which, in addition, connects to the period doubling boundary of period-one solutions separating sub- and supercritical period doubling bifurcations. The dash-dot lens near to the upper fold curve is an unstable island on the supercritically bifurcated family of period-two solutions, while the other dash-dot line corresponds to the stability loss of motion on the other branch of orbits.

6.2. Traffic model with driver reaction time. Here, we focus on the fold bifurcations of periodic orbits in the autonomous traffic model described in Orosz, Wilson, and Krauskopf [23]. We consider N_c cars on a circular track of length L, so

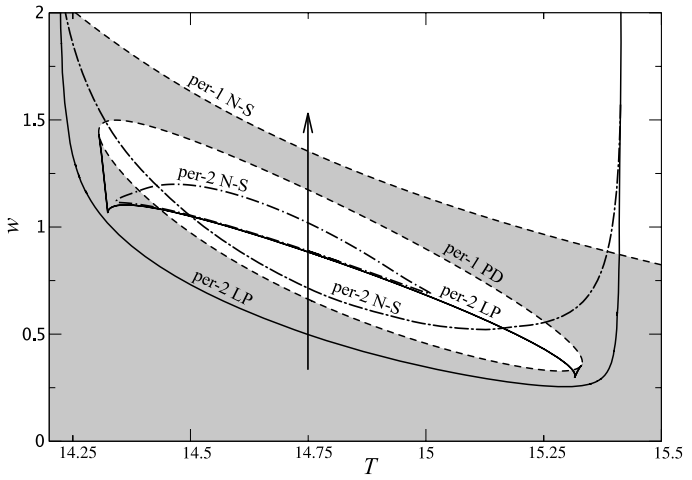


FIG. 3. *Bifurcations of period-two orbits. Dashed lines correspond to the stability of period-two solutions (see Figure 1). Continuous lines are fold bifurcation curves, and dash-dot lines are Neimark–Sacker stability curves of period-two solutions.*

that the distances between the cars (h_i) add up to L:

$$\sum_i h_i(t) = L. \tag{6.3}$$

In addition, these distances vary with the speed difference between two adjacent cars:

$$\dot{h}_i(t) = v_{i+1}(t) - v_i(t), \tag{6.4}$$

where v_i denotes the speed of the ith car. The drivers increase or decrease their speed according to their current speed and the headway before them. However, they cannot react immediately; thus a delayed value of the headway is considered in the model:

$$\dot{v}_i(t) = \beta[V(h_i(t-\tau)) - v_i(t)], \tag{6.5}$$

where β is the sensitivity of the drivers and V is the so-called optimal velocity function defined as

$$V(h) = \begin{cases} 0, & 0 \le h \le 1, \\ v^0 \frac{(h-1)^3}{1+(h-1)^3}, & h > 1. \end{cases}$$

The steady state solution of the model corresponds to equal headways and equal car velocities; that is,

$$h_i^* = L/N, \quad v_i^* = V(h_i^*),$$

where N_c is the number of cars on the track.

From now on we focus on the case $N_c = 3$; thus our equations become

$$\begin{aligned}
\dot{h}_1(t) &= v_2(t) - v_1(t), \\
\dot{h}_2(t) &= v_3(t) - v_2(t), \\
\dot{v}_1(t) &= \beta[V(h_1(t-\tau)) - v_1(t)], \\
\dot{v}_2(t) &= \beta[V(h_2(t-\tau)) - v_2(t)], \\
\dot{v}_3(t) &= \beta[V(L - h_1(t-\tau) - h_2(t-\tau)) - v_3(t)],
\end{aligned}$$

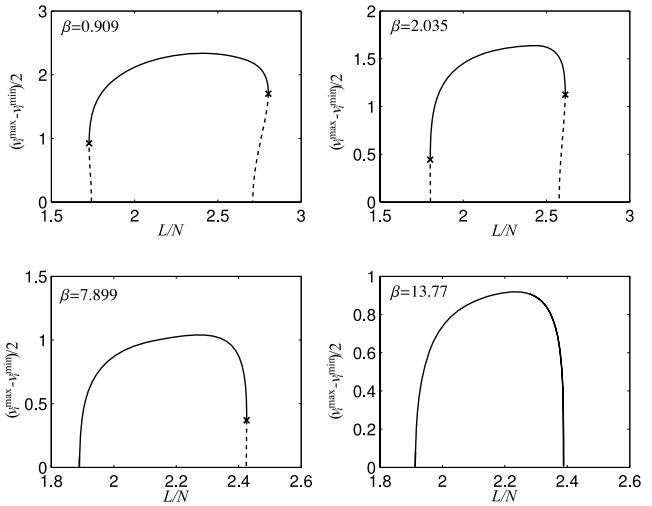


FIG. 4. *Branches of periodic orbits in the traffic model. Dashed curves represent unstable solutions, while continuous curves denote stable solutions. The parameters are* $v^0 = 0.8$ *and* $\tau = 1.0$.

using (6.3). The fixed points can undergo a Hopf bifurcation. Although it is not computationally efficient, we computed the Hopf bifurcation curve using our method, considering the fixed point as a periodic solution (see Figure 5). After computing the critical eigenvectors of the characteristic matrix we constructed a linearly approximate periodic solution around the fixed point

$$h_i(t) = \epsilon \mathrm{Re}(\mathrm{e}^{i\alpha t} h_i^q),$$
$$v_i(t) = \epsilon \mathrm{Re}(\mathrm{e}^{i\alpha t} v_i^q),$$

where ϵ is a sufficiently small real number, α is in the argument of the critical characteristic multiplier (see (4.5)), and h_i^q, v_i^q are the respective components of the kernel vector q of $\Delta(\mathrm{e}^{i\alpha})$. The computed branches of periodic solutions for different values of β can be seen in Figure 4. These periodic orbits undergo fold bifurcations, whose locations can be continued in two parameters using our method. These fold curves are represented by continuous lines in Figure 5, while the Hopf bifurcation of the fixed point is denoted by the dashed lines. It can be seen that the two types of bifurcations meet in a degenerate Hopf point, where the monodromy operator has triple $+1$ eigenvalues including the trivial one.

7. Discussion. In our presentation the characteristic matrix definition was given for equations with delays smaller than or equal to the period. However, our definition can be easily extended to any bounded delays such as the special case of integer delays in [22]. This extended algorithm is already implemented in the software PDDE-CONT [24].

Although the full equivalence of a characteristic matrix and the monodromy operator at spectral points remains to be proven, the correctness of our conjecture can be checked numerically with minimal effort. For discrete delays τ_i that are rationally related to the period, e.g., $\tau_i = r_i/p$, $r_i, p \in \mathbb{N}$, one can construct an $(n \times p)$-dimensional characteristic matrix that is equivalent to the monodromy operator, since the operator in Conjecture 2.2 can be inverted by solving ODEs. Hence for discrete delays that are sufficiently close to rational delays, a characteristic matrix can be constructed,

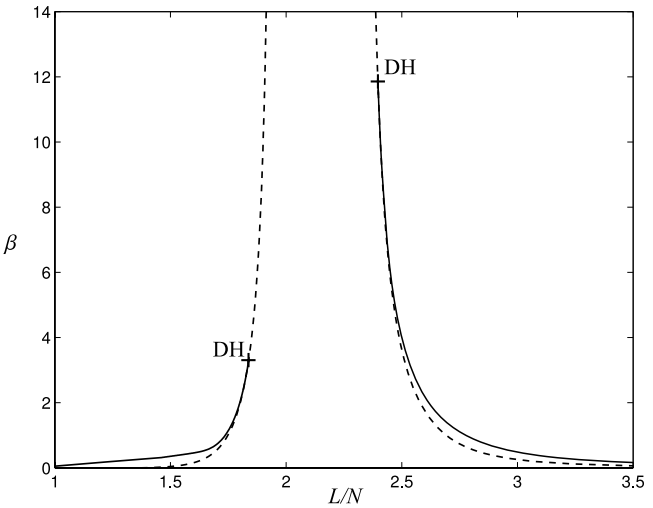


FIG. 5. *Stability chart. Between the two dashed curves denoting Hopf bifurcations, the fixed point is unstable. The arising periodic orbits fold back along the continuous curve. The fold curves meet the Hopf bifurcation curve in the degenerate Hopf points denoted by "+" signs ($v^0 = 0.8$, $\tau = 1.0$).*

because the resolvent set of an operator is open and the operator is norm continuous with respect to the delays.

The equivalence with the monodromy operator also gives the opportunity to compute center manifolds and normal forms for periodic equations in special cases where the characteristic matrix is analytically available as was demonstrated in [26].

8. Conclusions. Periodic DDEs frequently arise in applications that describe phenomena in engineering, biology, and other disciplines. In this paper we have constructed a characteristic matrix for general periodic DDEs in order to use it in the numerical continuation of bifurcations. We also have presented the equivalence of the characteristic matrix with the monodromy operator, which makes regularity proofs of defining systems of equations similar to the proofs in the case of finite-dimensional algebraic systems. We have implemented the algorithm in software written in C++ [24], which uses collocation to discretize the infinite-dimensional operators. Bordering techniques were also used in the solution of linear systems extended with the characteristic matrix, some derivatives, and tangent vectors. Finally, two examples, an interrupted machining model and a traffic model, were analyzed.

Acknowledgments. The authors thank Bernd Krauskopf, Kirk Green, Gábor Orosz, and Barnabás Garay for helpful discussions.

REFERENCES

[1] W. J. BEYN, A. CHAMPNEYS, E. J. DOEDEL, W. GOVAERTS, B. SANDSTEDE, AND Y. A. KUZNETSOV, *Numerical continuation and computation of normal forms*, in Handbook of Dynamical Systems, B. Fiedler, ed., North–Holland, Amsterdam, 2002, pp. 149–219.

[2] P. CLEMENT, O. DIEKMANN, M. GYLLENBERG, H. J. A. M. HEIMANS, AND H. R. THIEME, *Perturbation theory for dual semigroups.* I. *The sun-reflexive case*, Math. Ann., 277 (1987), pp. 709–725.

[3] P. Clement, O. Diekmann, M. Gyllenberg, H. J. A. M. Heimans, and H. R. Thieme, *Perturbation theory for dual semigroups.* II. *Time-dependent perturbations in the sun-reflexive case*, Proc. Roy. Soc. Edinburgh Sect. A, 109 (1988), pp. 145–172.

[4] T. A. Davies, *UMFPACK Version* 4.1 *User Guide*, Technical report, Department of Computer and Information Science and Engineering, University of Florida, Gainesville, FL, 2003; available online from http://www.cise.ufl.edu/research/sparse/umfpack/.

[5] C. de Boor and B. Swartz, *Collocation at Gaussian points*, SIAM J. Numer. Anal., 10 (1973), pp. 582–606.

[6] A. Dhooge, W. Govaerts, and Y. Kuznetsov, MATCONT*: A* MATLAB *package for numerical bifurcation analysis of ODEs*, ACM Trans. Math. Software, 29 (2003), pp. 141–164.

[7] O. Diekmann, S. A. van Gils, S. M. Verduyn Lunel, and H. O. Walther, *Delay Equations*, Springer-Verlag, New York, 1995.

[8] E. J. Doedel, W. Govaerts, and YU. A. Kuznetsov, *Computation of periodic solution bifurcations in ODEs using bordered systems*, SIAM J. Numer. Anal., 41 (2003), pp. 401–435.

[9] E. J. Doedel, A. R. Champneys, T. F. Fairgrieve, Y. A. Kuznetsov, B. Sandstede, and X.-J. Wang, *AUTO*97: *Continuation and Bifurcation Software for Ordinary Differential Equations*, Technical report, Department of Computer Science, Concordia University, Montreal, 1997; available via anonymous ftp from ftp.cs.concordia.ca from the directory pub/doedel/auto.

[10] E. J. Doedel, H. B. Keller, J. P. Kernévez, *Numerical analysis and control of bifurcation Problems.* I. *Bifurcation in finite dimensions*, Internat. J. Bifur. Chaos Appl. Sci. Engrg., 1 (1991), pp. 493–520.

[11] K. Engelborghs and E. J. Doedel, *Stability of piecewise polynomial collocation for computing periodic solutions of delay differential equations*, Numer. Math., 91 (2002), pp. 627–648.

[12] K. Engelborghs, T. Luzyanina, K. J. In 't Hout, and D. Roose, *Collocation methods for the computation of periodic solutions of delay differential equations*, SIAM J. Sci. Comput., 22 (2000), pp. 1593–1609.

[13] K. Engelborghs, T. Luzyanina, and D. Roose, *Numerical bifurcation analysis of delay differential equations using DDE-BIFTOOL*, ACM Trans. Math. Software, 28 (2002), pp. 1–21.

[14] W. Govaerts, *Bordered augmented linear systems in numerical continuation and bifurcation*, Numer. Math., 58 (1990), pp. 353–368.

[15] W. Govaerts, *Solution of bordered singular systems in numerical continuation and bifurcation*, J. Comput. Appl. Math., 50 (1994), pp. 339–347.

[16] W. Govaerts and J. D. Pryce, *Mixed block elimination for linear systems with wider borders*, IMA J. Numer. Anal., 13 (1993), pp. 161–180.

[17] J. K. Hale, *Theory of Functional Differential Equations*, Springer-Verlag, New York, 1977.

[18] J. K. Hale and S. M. Verduyn Lunel, *Introduction to Functional Differential Equations*, Springer-Verlag, New York, 1993.

[19] E. Hille and R. S. Phillips, *Functional Analysis and Semigroups*, AMS, Providence, RI, 1957.

[20] M. A. Kaashoek and S. M. Verduyn Lunel, *Characteristic matrices and spectral properties of evolutionary systems*, Trans. Amer. Math. Soc., 334 (1992), pp. 479–517.

[21] Y. A. Kuznetsov, *CONTENT—Integrated Environment for Analysis of Dynamical Systems: Tutorial*, Rapport de Recherche UPMA-98-224, Ecole Normale Superieure de Lyon, Lyon, France, 1998.

[22] S. M. Verduyn Lunel, *Small solutions and completeness for linear functional differential equations*, in Oscillations and Dynamics in Delay Equations, Contemp. Math. 129, AMS, Providence, RI, 1992, pp. 127–152.

[23] G. Orosz, R. E. Wilson, and B. Krauskopf, *Global bifurcation investigation of an optimal velocity traffic model with driver reaction time*, Phys. Rev. E (3), 70 (2004), article 026207.

[24] R. Szalai, *PDDE-CONT: A Continuation and Bifurcation Software for Delay-Differential Equations*, Technical report, Budapest University of Technology and Economics, Budapest, Hungary, 2005; available online from http://www.mm.bme.hu/~szalai/pdde/.

[25] R. Szalai and G. Stépán, *Stability boundaries of high-speed milling corresponding to period doubling are essentially closed curves*, in Proceedings of IMECE2003, ASME International Mechanical Engineering Congress and R&D Expo, 2003, pp. 1–6 (CD-ROM).

[26] R. Szalai and G. Stépán, *Period Doubling Bifurcation and Center Manifold Reduction in a Time-Periodic and Time-Delayed Model of Machining*, preprint, 2006; available online from http://www.mm.bme.hu/~szalai/publications.html.

[27] J. Tlusty, *Manufacturing Process and Equipment*, Prentice–Hall, Englewood Cliffs, NJ, 2000.

SIAM J. SCI. COMPUT.
Vol. 28, No. 4, pp. 1318–1334
© 2006 Society for Industrial and Applied Mathematics

PARALLEL SUPPORT SET SEARCHES FOR MESHFREE METHODS*

C. CARTWRIGHT†, S. OLIVEIRA‡, AND D. E. STEWART§

Abstract. We describe the implementation of a parallel algorithm that solves a computational geometry problem arising in meshfree methods. We solve the following problem: Given a collection of N d-rectangles S_i and Q points x_k in $\mathbb{R}^d$, compute $I_k = \{i \mid x_k \in S_i\}$. This task is necessary for computing values of the meshfree basis functions $\Psi_i(x_k)$ and can be used to efficiently compute the mass and stiffness matrices for the meshfree Galerkin method.

Key words. meshfree methods, computational geometry, quadtrees

AMS subject classifications. Primary, 65D18; Secondary, 68W10, 65N30

DOI. 10.1137/S1064827502414321

1. Introduction.

Recently there has been a growing interest in developing meshfree methods for solving partial differential equations (PDEs) as an alternative to finite element methods (FEMs). Meshfree methods in general require more complicated geometric tasks than are associated with FEMs. Efficient computational geometry algorithms are necessary if meshfree methods are to be viable.

In this article we look at the solution of a particular computational geometry problem arising from meshfree Galerkin methods, a class of meshfree methods. In section 2 we introduce meshfree methods and the geometric task that they require for efficient implementation. In section 3 we discuss quadtree data structures and how we use them to solve our particular problem. Then in section 4 we discuss how to parallelize the sequential algorithm described in section 3, considering the issues of data distribution which can be rather irregular in applications. Parallel performance estimates are derived in section 5, followed in section 6 by a discussion of a partitioning algorithm we use to balance the computational load. Finally in section 8 we give numerical results that illustrate the scalability of our algorithm and improvements we are able to achieve through attempting to control the load balance among processors.

We use standard asymptotic notation: $f(n) = O(g(n))$ means that there is a C and an n_0 where $n \geq n_0$ implies $|f(n)| \leq C\,|g(n)|$. Furthermore, $f(n) = \Omega(g(n))$ means that $g(n) = O(f(n))$; $f(n) = \Theta(g(n))$ means that $f(n) = O(g(n))$ and $g(n) = O(f(n))$.

2. Geometric problems arising in meshfree methods.

Some of the limitations of the traditional FEMs for solving PDEs are the computationally intensive tasks of generating meshes with good quality and of regenerating meshes when necessary. Recently there have been several methods, known as meshfree methods, that do not require constructions of grids (meshes) and are being developed as an alternative

*Received by the editors September 12, 2002; accepted for publication (in revised form) February 20, 2006; published electronically July 31, 2006. This research was supported in part by NSF grants DMS-9874015 and DMS-0213305.

http://www.siam.org/journals/sisc/28-4/41432.html

†Lawrence Technological University, Department of Mathematics and Computer Science, 21000 West Ten Mile Road, Southfield, MI 48075-1058 (cartwright@ltu.edu).

‡Department of Computer Science, University of Iowa, Iowa City, IA 52242 (oliveira@cs.uiowa.edu).

§Department of Mathematics, University of Iowa, Iowa City, IA 52242 (dstewart@math.uiowa.edu).

to FEMs that for large deformation problems may perform better than traditional FEMs. There are a number of types of meshfree methods, such as [illegible] (PUFEMs), [illegible] (RKPMs), [illegible] (EFG), [illegible], and [illegible] (MLS). [illegible]. An overview of meshfree methods can be found in [1] and more recently in [5]. In this article we will focus on computational geometry techniques that are applicable to meshfree Galerkin methods.

In meshfree Galerkin methods, a set of basis functions Ψ_i, $i = 1, 2, \ldots, N$, is defined, and we seek an approximate solution $u^h = \sum_{i=1}^{N} u_i^h \Psi_i(x)$. To solve a PDE such as

$$-\nabla \cdot A(x)\nabla u + c(x)u = f$$

in Ω with Neumann boundary conditions, we construct a linear system of equations to solve for the u_i's, $\sum_{j=1}^{N} K_{ij} u_j^h = f_i$ for $i = 1, 2, \ldots, N$, where

$$K_{ij} = \int_{\Omega} (\nabla \Psi_i(x)^T A(x) \nabla \Psi_j(x) + c(x)\Psi_i(x)\Psi_j(x)) dV(x). \tag{2.1}$$

In all meshfree methods, one starts with an a priori given set of [illegible] Φ_i on Ω, which are typically smooth and nonnegative and have compact support. Usually these kernel functions are obtained from a single reference function using "centers" x_i and "radii" a_i: $\Phi_i(x) = \Phi((x - x_i)/a_i)$. The reference function $\Phi(z)$ is commonly a tensor product of B-spline functions $\Phi(z) = B(z_1)\, B(z_2) \cdots B(z_d)$, where B is a standard cubic spline function with nodes at -1, $-1/2$, 0, $+1/2$, $+1$, with support $[-1, +1]$. Alternative choices include $\Phi(z) = B(\|z\|_2)$ and radial basis functions.

From these kernel functions, meshfree methods construct basis functions that have better approximation properties. Partition of unity methods (PUMs) usually set

$$\Psi_i(x) = \frac{\Phi_i(x)}{\sum_{j=1}^{N} \Phi_j(x)},$$

which are called Shepard functions in the meshfree literature [10]. These basis functions are then "enriched" by adding additional basis functions $\Psi_i(x)\, v_j(x - x_i)$, where v_j belongs to a small set of functions, typically polynomials. For example, $v_j(z) = z_j$, $j = 1, 2, \ldots, d$, so that any linear function can be exactly represented by basis functions. Alternatively, RKPMs construct basis functions in order to be able to reproduce a certain space of functions, typically low order polynomials. Suppose we wish to reproduce all affine functions so that if f is an affine function, $f(x) = \sum_{i=1}^{N} \Psi_i(x)\, f(x_i)$. Then we set $H(z) = [1, z_1, \ldots, z_d]^T$ and construct the RKPM basis functions by the rule

$$\Psi_i(x) = H(0)^T M(x)^{-1} H(x - x_i)\, \Phi_i(x),$$

where $M(x) = \sum_{i=1}^{N} H(x - x_i) H(x - x_i)^T\, \Phi_i(x)$.

It should be noted that unless there is some kind of cancellation, the supports of the basis functions are exactly the supports of the kernel functions, which are typically rectangles or balls in $\mathbb{R}^d$. An illustration of how some of the support sets might appear is shown in Figure 1. Also note that in order to compute $\Psi_i(x)$ in either the PUMs or

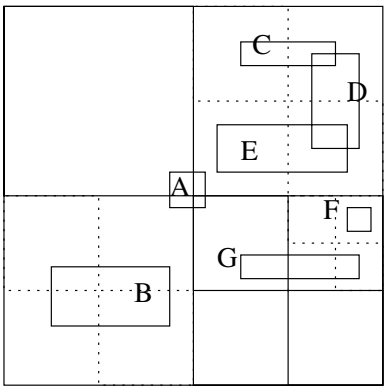


FIG. 1. *Sample collection of rectangles. For clarity we show only a few rectangles; normally the rectangles cover* Ω.

RKPMs it is necessary to compute the values of $\Phi_j(x)$ for ,, j. In practice, for fixed x, $\Phi_j(x) \neq 0$ for only a small set of values of j. Thus, in order to be computationally efficient, given x, we would like to know the set $I(x) = \{\, j \mid x \in \operatorname{supp} \Phi_j \,\}$ in order to compute $\Psi_i(x)$ for any and all i. Derivatives can be obtained by simply differentiating the formulas given for Ψ_i and substituting the values and derivatives of the kernel functions.

The only meshfree methods presented in the literature which do not require finding $\{\, i \mid x \in \operatorname{supp} \Phi_i \,\}$ are the methods of Klaas and Shephard [6] and Griebel and Schweitzer [3] which construct the basis functions in a way based on quadtrees or octrees where the support sets covering a given point can be determined by looking at neighboring rectangles in the octree. This can be done using pointers in the quadtree/octree nodes to neighboring nodes and ensuring that the levels of these neighboring nodes differ by at most one. To make this technique work, however, Klaas and Shephard make some restrictive assumptions about the "radii" a_i of the support sets used. The width of a support set, assuming that the center is exactly a center of a quadtree or octree rectangle, must be between the width of that octree rectangle and $1\frac{1}{2}$ times the width of that octree rectangle. Modifications of their algorithm could increase this ratio to any number less than 2 with a potentially large increase in the methods complexity, but a ratio of 2 or higher renders their method inapplicable. This could become a serious obstacle to adaptive methods. The method of Griebel and Schweitzer is more flexible, but it too restricts the placement of support sets in order to enable fast neighborhood searches. Even so, for a general set of quadrature points, the method of Griebel and Schweitzer still requires a search of the quadrature points for each nonzero entry of the stiffness matrix. By contrast, the method described below has no such restrictions.

Not only can we use our algorithm for the task of computing $\Psi_i(x)$ for a given $x \in \Omega$, but we can also use it to efficiently assemble stiffness matrices using quadratures:

$$K_{ij} \approx \sum_{i=1}^{Q} [\nabla \Psi_i(y_k)^T A(y_k) \nabla \Psi_j(y_k) + c(y_k) \Psi_i(y_k) \Psi_j(y_k)]\, \omega_k.$$

This can be done via the following algorithm that loops first over the quadrature points. Note that $S_i = \operatorname{supp} \Psi_i$:

for $k = 1, \ldots, Q$

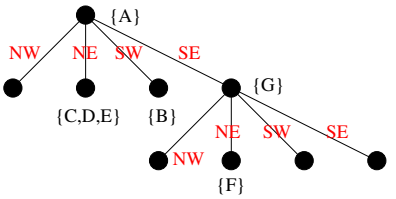


FIG. 2. *MX-CIF quadtree for rectangles in Figure* 1.

$$
\begin{aligned}
&I_k \leftarrow \{\, i \mid y_k \in S_i \,\} \\
&\text{for } i,\, j \in I_k \\
&\qquad K_{ij} \leftarrow K_{ij} + [\nabla \Psi_i(y_k)^T A(y_k) \nabla \Psi_j(y_k) + c(y_k)\Psi_i(y_k)\Psi_j(y_k)]\, \omega_k.
\end{aligned}
$$

The task of computing I_k may of course be solved by exhaustively testing each y_k with every S_i to see which S_i cover y_k. This would be an inefficient strategy, since there would be many "wasted tests" where a support set did not contain a certain point. The use of quadtrees is effective in reducing the number of wasted tests and thus producing an efficient algorithm. The sequential algorithm of Han, Oliveira, and Stewart [4], which uses quadtrees to compute I_k, is discussed in section 3. The remainder of the article is devoted to the parallelization of that algorithm and issues of parallel performance, especially speedup and load balancing.

3. Quadtrees. A quadtree is a tree data structure where each node has four children. An octree is a tree in which each node has eight children. Each of these structures is a generalization of binary trees. They may be further generalized to trees in R^d in which each node has 2^d children. For solving 2-dimensional problems, we use region-based quadtrees, in which each node of the quadtree is a rectangle. The root of the quadtree is the entire domain (or enclosing rectangle) over which the PDE is solved, and the children of a node are the 4 subrectangles which form the parent rectangle.

3.1. MX-CIF quadtrees. In the sequential algorithm of Han, Oliveira, and Stewart, the expanded MX-CIF quadtree data structure (developed by Abel and Smith, and named by Samet) is used [9]. An MX-CIF quadtree is an area-based quadtree in which each set S (circle or rectangle) is associated with the quadtree node corresponding to the smallest block (rectangle) that contains S in its entirety. For the support sets shown in Figure 1, the MX-CIF quadtree is shown in Figure 2. For this application, each point must be tested against all sets linked to the nodes in the path from the root node to the leaf node containing the point. The farther down the tree we go, the fewer candidate points there will be to test. To reduce the number of tests necessary, we would like to link the sets down as far in the quadtree as possible. For our application, the MX-CIF quadtree would not handle appropriately small rectangles that do not fit nicely inside blocks, e.g., the set A in Figure 1. These sets would be tested against a larger number of points than is really necessary. To help fix this problem, the [illegible] MX-CIF quadtree is used. For each set S, instead of linking it to the smallest enclosing block B, if $diam(S) < \frac{1}{2} width(B)$ we look at the children of B, B_i and split S into the parts $S'_i = S \cap B_i$. We then find the smallest enclosing blocks, D_i, of S'_i, repeating the procedure for all D_i where $diam(S'_i) < \frac{1}{2} width(D_i)$. Use of the [illegible] quadtree will ensure that the smaller the set is, the further down the quadtree it will be stored, and fewer tests will be

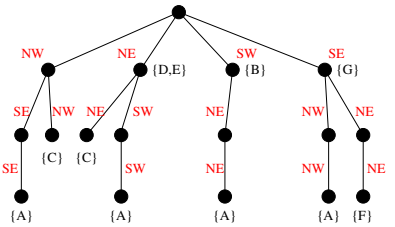


FIG. 3. *Expanded MX-CIF quadtree for rectangles in Figure* 1.

required. For the support sets shown in Figure 1, the expanded MX-CIF quadtree is shown in Figure 3.

The sequential algorithm of Han, Oliveira, and Stewart works in two stages. Stage 1 consists of building the expanded MX-CIF quadtree, which for this application the authors describe as a "splitting" algorithm (since sets that overlap blocks will be split into smaller sets). In stage 2, each point is tested to find which sets cover that point using the quadtree structure.

Note that for any additional quadrature points that are determined later, we merely need to rerun stage 2 with the new quadrature points. Thus this method is also applicable to adaptive quadrature methods.

3.2. Adaptive methods and the quadtree algorithm. While handling adaptivity is not the main purpose of this paper, we would like to make some comments about how the quadtree algorithm can handle adaptive changes in the collection of support sets $\{ S_i \mid i = 1, 2, \ldots, N \}$ and the set of quadrature points $\{ x_k \mid k = 1, 2, \ldots, Q \}$. Typically, adaptive algorithms change a small number of support sets, perhaps replacing one support set S_i with a small number of new support sets $S'_{i_1}, S'_{i_2}, \ldots, S'_{i_p}$. Similarly, an adaptive algorithm would remove a small subset of the set of quadrature points and add a set of new quadrature points.

The process of adding a new support set S'_j to an MX-CIF quadtree is the same as that used for building the initial MX-CIF quadtree. Deletion of a support set S_i can also be accomplished easily by removing any links to S_i and then deallocating any "dead" subtrees of the quadtree that contain no links to support sets. Alternatively, the two operations could be separated: The removal of links could be done as a "batch" operation, and then a "pruning" operation could be applied to the entire tree to remove all "dead" subtrees. To make this more efficient, each node of the quadtree can have a "modified" bit to indicate if it, or any descendant of it, has been modified. This can be done efficiently by setting the "modified" bit of any node accessed by an addition operation; the deletion of a support set would involve unlinking the support set, then setting the "modified" bits of the linking quadtree node and its ancestor nodes. These bits would be reset after a pruning operation or after updates have been completed.

Adjusting the sets $I_k = I(x_k)$ could be done by reprocessing all of the quadrature nodes. This expense can be reduced by using a "modified" bit as described in the previous paragraph: For each quadrature point x_k, go down the tree as required by the standard algorithm, checking the linked support sets, but stop when the "modified" bit is not set. The new list of support sets I'_k then consists of all the support sets found before reaching the unmodified subtree, plus all support sets in the original list I_k linked at a depth greater than or equal to the depth of the root of the unmodified subtree. New quadrature points will have to be processed using the standard algorithm.

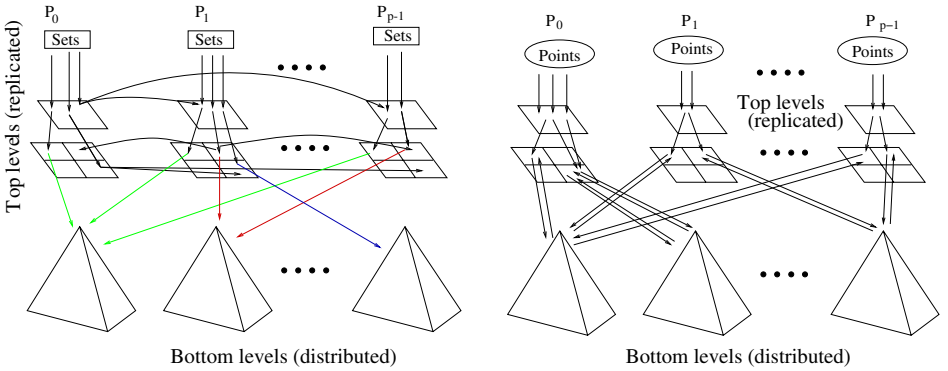
4. Parallelizing the algorithm. We assume that there is an algorithm which takes a collection of support sets and a data structure defining the region of integration, and computes the set of quadrature points and weights. These algorithms typically generate quadrature points that depend locally on the collection of support sets nearby. For parallel implementations, this often translates into quadrature points being generated on processors that are close to the processors that "own" support sets that contain them.

However, in the development of our algorithm, we do not assume such locality. First, this gives our algorithm greater flexibility. Different algorithms may well result in different levels or kinds of locality, or may occasionally violate a specific condition on how near quadrature points are to the support sets that contain them in the network of processors. Second, our algorithm can make use of this locality, as long as the all-to-all communication routines can make use of the locality of the communication pattern (such as avoiding communication altogether when source and destination processors are the same).

4.1. Problem of data distribution. A simple approach to parallelizing the algorithm for finding sets which cover a point would be to evenly distribute the set data. Then the sets on local processors could be linked to nodes in the quadtree. While this approach requires little space on each processor (only part of the quadtree structure is stored on each processor), this has a disadvantage in that $\frac{n-1}{n}$ of the points would have to be located on all the processors, meaning that there is parallelism only in regard to set data but not to point data. Alternatively, it is possible to create an algorithm which would distribute all of the points but would require that all of the sets and the quadtree structure be duplicated on all of the processors. This would require a lot of space on each processor and would also lack any degree of parallelism with respect to set data. Achieving parallelism in both the set and point data, while also distributing the quadtree structure as much as possible, requires a more complicated parallel algorithm.

4.2. Solution: Use *two-level* quadtree structures. Our solution was to divide the quadtree into two parts: one for the top levels, levels 0 (the root) up to level L (the user chooses L), and another structure for the bottom levels (all nodes at level L or lower in the tree). Each processor contains the entire top level structure, while the bottom level structure is distributed among all processors. Each processor is assigned a branch (one or more) of the tree, i.e., a node(s) at level L and all descendant nodes. This algorithm balances the costs and benefits of the simpler approaches mentioned in the previous section. The benefit is that both the set and point data may be initially evenly distributed, increasing parallelism, while the cost of the data overhead required may be minimized by choosing L appropriately, since only the top L levels of the quadtree need to replicated among all processors. This is illustrated in Figure 4.

4.3. Description of the steps in our parallel algorithm. In parallelizing stage 1 of the sequential algorithm, we use the splitting algorithm to process the support sets. After the sets have been evenly distributed among all processors, each processor calls the splitting algorithm on its collection of sets. If a set is to be linked into nodes at the top L levels of the quadtree, this set's data will need to be broadcast to all other processors and must be linked to the top quadtree structure in all processors. This is necessary to allow the equal distribution of points, since any processor may have a point that is located within some rectangle at a node in any



(a) Constructing the quadtree

(b) Testing the quadrature points

FIG. 4. *Two-level parallel quadtree data structures.*

level of the quadtree. If a set is to be linked to a rectangle in the bottom levels of the quadtree, the set data is sent to the appropriate processor assigned that rectangle, and is then linked in at the appropriate node(s). Recall that it is possible for a set to be linked to more than one subrectangle in possibly different levels, so that a support set may be linked to both the top and bottom levels, and thus a set's data may be contained on more than one processor. This is illustrated by Figure 4(a).

For stage 2 to run in parallel, each processor first checks all of its sets linked to the top level against all of its assigned points (which are initially evenly distributed among processors) to see which sets cover which points. Then an exchange of points is made so that all points are sent to the processor which contains the rectangles (the "roots" of the bottom level quadtrees) in which those points are located. Finally the received points are checked against the sets linked to the bottom levels of the quadtree to see which sets cover which points. This is illustrated by Figure 4(b).

Locality of support sets and quadrature nodes can be utilized by solving the stable marriage problem for matching bottom-level quadtrees and processors, where the desirability of a match of a bottom-level quadtree and a processor is given by the number of support sets in the bottom-level quadtree that are "owned" by the given processor. Cheaper alternatives to solving the full stable marriage problem can include assigning bottom-level quadtrees to the (unmatched) processor that owns the largest number of support sets in the bottom-level quadtree. Note that either of these techniques can be applied [illegible]. Note that solving the full stable marriage problem would take $O(p^2)$ in the worst case, but only $O(p \log p)$ in the average case. The cheaper methods require only $O(p)$ time.

5. Theoretical estimates. For the following estimates the following notation is used: d is the dimension of the problem (usually $d = 2$ or 3), $\epsilon = \min_i \operatorname{diam}(S_i)$, N is the number of support sets, p is the number of processors, Q is the number of (quadrature) points, n_t is the number of S_i's linked to top levels in the quadtree, and n_b is the number of S_i's linked to the bottom levels of the quadtree. Note that $n_b + n_t = N$.

5.1. Parallel computation time. The average computation time for stage 1 of the algorithm (to process the support sets, assuming they are uniformly distributed) may be estimated as follows.

The computations should scale for the sets linked to the bottom level of the quadtree. Since the quadtree structure has depth $O(\log(1/\epsilon))$, the computation time should be

$$O\left(\left(\frac{n_b}{p}\right)\log(1/\epsilon)\right).$$

There will be some overhead for the top level sets since all must be on each processor. However, if the sets are fairly small, most sets should be linked to the lower levels and not to the top levels, and so the algorithm should be fairly scalable for sets in which there are not too many large sets.

The total computation time from stage 1 is then

$$O\left(\left(\frac{n_b}{p}\right)\log(1/\epsilon) + n_t\right).$$

The average computation time required for stage 2 (to test which points are covered by which sets) can be computed as follows: Since the points should be evenly distributed and the quadtree has depth $O(\log(1/\epsilon))$, the points should be tested on average

$$O\left(\frac{Q}{p}\left[\log\left(\frac{1}{\epsilon}\right) + \mathbb{E}(\mathit{covering\#})\right]\right)$$

times, where $\mathbb{E}(\mathit{covering\#})$ is the expected number of sets covering each point. Stage 2 should be completely scalable, except for some small $O(1)$ space overhead.

5.2. Communication time. The average case communication time (under the case of uniformly generated sets and points) may be estimated as follows: There are a fixed number of communications, 14, each with a start-up time of t_s.

Assuming that t_{all} is the time to send one word (or byte) of information via an all-to-all exchange (the only type used in this algorithm), the 14 communications have roughly the following totals of information exchanged:

- 2 `Allreduce`s of total length 2^{dL} each,
- 4 `Alltoall`s of total length p each,
- 7 `Alltoallv`s, 4 of length n_b and 3 of length Q,
- 1 `Allgatherv` of length n_t.

Combining these we get a total (average case) estimate of the communication time of

$$14t_s + t_{all}[2(2^{dL}) + 4p + 3Q + 4n_b + n_t].$$

In most cases Q will be much larger than the other variables affecting the communication, so the number of points should be what affects the communication time the greatest.

Finally, there will be some time due to latency. Since we decided on synchronous communication and using only all-to-all communication, for each of the 14 communications, all processors fill the entire buffers with data before sending any data. So the waiting time between computations and communication is directly proportional to the load imbalance. Thus load balancing is important for controlling both computation costs and latency.

6. Partitioning for load balance. For applications over regular domains with uniform distributions of support sets and quadrature points, the computational load should already be approximately balanced. Each branch of the bottom-level quadtrees should require roughly the same amount of work, and the best approach is to assign the same number of branches to each processor; i.e., p should divide evenly into 2^{dL}. For irregularly shaped domains or where the distribution of points is not uniform, the work required on each branch of the bottom level may vary significantly. Our a priori estimate for the amount of work done on each processor is the number of sets linked to the bottom-level branches. This is a good predictor for the actual load since the work depends on the area of the sets, and the splitting algorithm helps to ensure that sets in the same level of the tree have similar size areas. We seek to partition the branches among the processors such that the number of sets assigned to each processor is roughly equal.

We do this by minimizing the sum of the number of sets linked to the branches assigned to any single processor (i.e., we minimize the maximum load on any single processor).

Specifically, the problem we seek a solution for may be stated as follows: Given a sequence of N nonnegative integers $a_1, a_2, \ldots, a_N$, let $\mathcal{B} := \{1, 2, \ldots, N\}$:

$$\text{minimize } \max_k \sum_{i \in B_k} a_i \quad \text{such that} \quad \{ B_k \mid k = 1, \ldots, p \} \text{ is a partition of } \mathcal{B}.$$

Here a_i is the number of sets assigned to the bottom levels of branch i, and B_k consists of the branches assigned to processor k.

Note that this problem is the minimum makespan scheduling problem described by Vazirani [12] and is known to be NP-hard. We need only an approximate solution to this discrete optimization problem since this problem is based only on an estimate of the amount of computational work. We implemented the following heuristic algorithm of Graham [2]:

Sort the elements so that (after relabeling) $a_1 \geq a_2 \geq \cdots \geq a_N$
$B_k \leftarrow \emptyset$ for $k = 1, 2, \ldots, p$
$m_k \leftarrow 0$ for $k = 1, 2, \ldots, p$
For $i = 1$ to N
    $k \leftarrow \text{argmin}_l \, m_l$
    $m_k \leftarrow m_k + a_i$
    $B_k \leftarrow B_k \cup \{i\}$.

Graham was able to prove that this algorithm produces a value of $\max_k \sum_{i \in B_k} a_i$ within a factor of 4/3 of the optimum [2].

Upon termination of the algorithm we will have a partition of the set $S = \cup_{k=1}^{p} B_k$, and $m_k = \sum_{i \in B_k} a_i$ are the loads on each processor. A list of branches to be assigned to each processor is also produced in the algorithm implemented for this application.

Increasing L increases the number of possible processors, since if there are L levels in the top level, there are 2^{dL} leaf rectangles of the top level, and each leaf may be assigned to only one processor; i.e., the number of processors is limited by 2^{dL}. Without load balancing, exactly one branch is assigned to each processor, and so we must have exactly $p = 2^{dL}$. Load balancing allows us to choose the number of processors flexibly, allowing any $p \leq 2^{dL}$.

Increasing L will improve load balancing up to a point. Since the number of branches to be assigned is 2^{dL}, exponentially more branches are created as L is increased. Even using our heuristic, finding a solution to the discrete optimization

problem may outweigh the savings gained by increasing L further. Increasing L also will increase the number of sets linked to the top level, which will reduce some of the parallelism in regard to the set data.

Locality of support sets and quadrature points can also be exploited in the load-balancing step. A simple approach is to first use the above load-balancing heuristic to obtain balanced groups of bottom-level quadtrees and then match the groups to processors using the same criteria as described at the end of section 4.3. An alternative would be to combine load balancing (the number of support sets on each processor should be roughly the same) and locality (bottom-level quadtrees should be assigned to processors which contain the majority of their support sets) into a single objective function and then apply a greedy algorithm to assign the bottom-level quadtrees to processors. Note that the optimal way to assign bottom-level quadtrees will depend on the ratio of speed of computation to speed of communication: Relatively large communication times would make locality more important than balancing. The objective function used to assign bottom-level quadtrees to processors would need to take this into account.

7. Adaptivity in parallel. As noted in section 3.2, we would like to make some comments regarding how to handle adaptive methods efficiently, although this is not a main purpose of this paper. The "modified" bit can be incorporated into quadtree nodes in the parallel case as for the sequential case. Parallel additions of support sets can be done as for constructing the original quadtree, except that quadtree nodes traversed in the addition algorithm would have the "modified" bit set. Parallel deletions of support sets should be initiated by the processor "owning" the support set. Provided each processor keeps a record of which bottom-level quadtree(s) is linked to the deleted support set, nodes in the top-level quadtree can be marked as "modified." Then a message would need to be sent to the processor owning the corresponding bottom-level quadtree, indicating that this support set should be deleted. The processor containing the bottom-level quadtree can then apply the sequential deletion algorithm: Remove the link and set the "modified" bits of the linking node and its ancestors in the bottom-level quadtree. Collections of these operations can be combined to reduce overhead. Top-level nodes which have the modified bit set must be communicated to all processors. Since finding all modified top-level quadtree nodes is essentially a union operation, it can be performed in $O(\log p)$ communication steps with $\leq 2^{dL}$ items to communicate in each step. Since adaptation typically occurs in small regions, the amount of data that needs to be communicated could be considerably less than this, although 2^{dL} is typically modest (say, $d = 2$ and $L = 3$ gives $2^{dL} = 64$).

Rebalancing may eventually prove necessary. This would require the broadcast of 2^{dL} data items in order to determine a new assignment of data items. The algorithm to do this should take into account the current assignment of bottom-level quadtrees to processors in order to minimize unnecessary communication.

Processing the old set of quadrature points can be done using the modified bits of the quadtree nodes as noted in section 3.2. In particular, note that if traversing the top-level quadtree for a given quadrature point stops at a node where the "modified" bit is not set, that quadrature point does not need to be communicated to the processor containing the associated bottom-level quadtree. This, combined with the techniques for using locality, can greatly reduce the amount of communication needed. New quadrature points would have to be processed as normal.

8. Numerical results. We implemented our algorithm in parallel in C using MPI, the Message Passing Interface. Helpful references are the texts of Pacheco [8] and Snir et al. [11]. We ran the test cases on the Xeon cluster (Tungsten) at the National Center for Supercomputing Applications at the University of Illinois, Urbana-Champaign. This is a distributed memory cluster-type parallel computer.

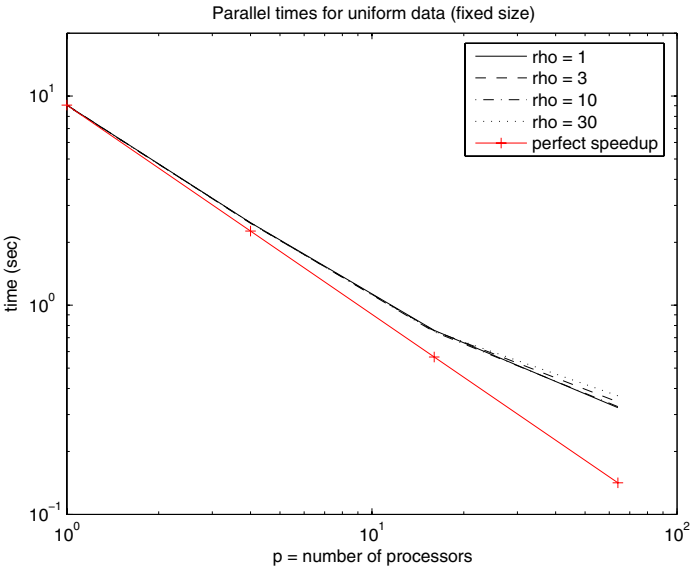
The test data we used was randomly generated. There were two main ways in which this data was generated. The first was uniform generation on support sets and test points on the unit square $[0,1] \times [0,1]$. The second had support sets and test points generated over a circle with a wedge removed from it, creating a reentrant corner. The density of support sets and points was chosen to have an $r^{-\alpha}$ distribution where r is the distance from the reentrant corner. This nonuniform distribution was chosen to have no support sets or points in the southeast quadrant of the unit square.

Two kinds of tests were carried out with this data. One was with a fixed problem size to test speedup. The other consisted of scalability tests where the number of support sets and the number of test points increased proportionally to the number of processors. In keeping with the application to meshfree methods, the support sets generated became smaller as the number of support sets increased so as to keep the average covering number ρ_0 (the expected number of support sets covering a random point) approximately constant. Tests were done with different values for ρ_0. Since the data generation was done using pseudorandom number generators, we repeated the problems with different seed values in order to estimate the effect of randomness on the results. Parallel times were obtained using the MPI routine `MPI_Wtime`. This avoided including the program loading and setup overhead.

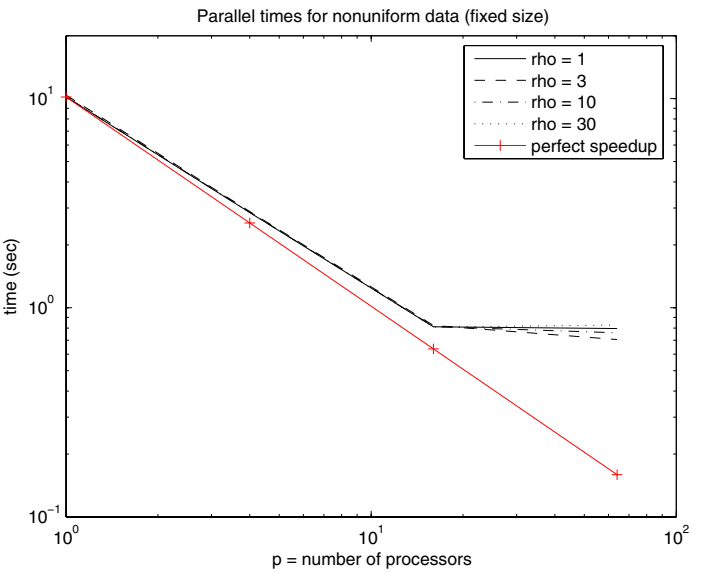
8.1. Data generation. For the uniform data generation, center points for the support sets and test points were generated uniformly on the unit square $[0,1] \times [0,1]$. All the generated support sets were square but with a width W that was chosen uniformly from an interval $[0, w_0]$, with w_0 chosen to give the desired average covering number ρ_0. To do this we ensured that $N\,\mathbb{E}(A) = \rho_0\,A_0$, where A is the (random) area of the generated support set, $\mathbb{E}$ is the expectation operator, and A_0 is the area of the region covered (that is, the unit square; thus $A_0 = 1$). Since $A = 4W^2$, some simple calculations give the parameter $w_0 = \sqrt{3\rho_0/(4N)}$ needed for the generation of the support sets. The Unix pseudorandom number generator `drand48` was used for data generation.

For the nonuniform data generation, an exponent $\alpha > 0$ is chosen where the probability density for the support set centers and the test points were both $\rho(r) := C_\alpha\, r^{-\alpha}$, where $0 < r \le 1$ is the radius from the reentrant corner. The constant C_α is chosen so that the total probability is one; that is, $\int_0^{r_{\max}} (3\pi/2)\, r\, \rho(r)\, dr = 1$. Again the support sets generated are squares with width W; thus the area is $A = 4W^2$. The distribution of W is chosen to depend on r so that $N\,\mathbb{E}(4W^2)\,\rho(r) = \rho_0$. The corner is set to be at (0.45, 0.55); the value of $r_{\max} = 0.4$. Unlike the uniform case, W is determined directly from r, instead of involving yet another (pseudo)random number. Since r is already a random quantity, we did not think that there was any need to have additional randomness in the size of the support set.

Note that nearby support sets and quadrature points generated by this procedure are usually not generated on the same or nearby processors. That is, there is no locality of support sets and quadrature points to exploit. In a meshfree system which does generate nearby support sets and quadrature points on the same or nearby processors, the techniques for exploiting locality described at the end of sections 4.3 and 6 should improve the performance beyond what is presented here.



(a)



(b)

FIG. 5. *Parallel times for fixed problem size (speed-up tests):* (a) *uniform data,* (b) *nonuniform data.*

8.2. Speed-up tests. The simplest tests we performed were speed-up tests. In these tests we fix the size of the problem and increase the number of processors up to some upper limit. For our tests, we used 1, 4, 16, and 64 processors. We used 1.28×10^6 support sets and 1.92×10^6 test points.

The times for these runs are shown in Figure 5.

As can be seen in Figure 5(a) and (b), the speedup is good up to about 16 proces-

sors, but there is reduced improvement in going to 64 processors with uniform data and no improvement in going to 64 processors with nonuniform data. For any problem with a fixed data size there is a limit to how much improvement can be gained from parallelism.

The best speedup for nonuniform data is about 13.2 for 16 processors, giving a parallel efficiency of about 82%. For uniform data the best speedup obtained is about 26.5 for 64 processors, giving a parallel efficiency of only about 41%. If we stopped at 16 processors with the uniform data we would get a speedup of 12.0, giving a parallel efficiency of 75%.

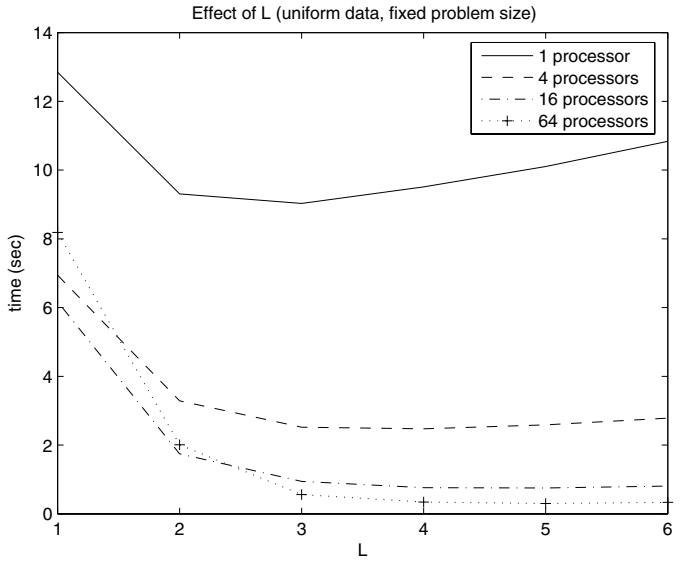
These speedups were obtained using the times for the optimum value of L. A question that naturally arises is how the choice of L affects the times; theoretically this should depend solely on the size of the support sets, and does not depend on the size of the problem or the number of processors. This can be seen in the times for all the values of L used ($L = 1, 2, \ldots, 6$) as shown in Figure 6.

Indeed, we see that for $L \geq 3$ for uniform data ($L \geq 4$ for nonuniform data) with $p \geq 4$ the difference in the times is negligible. Note that $L = 3$ means that the top-level tree has $4^3 = 64$ leaves, which is sufficient for good load balance for uniform data even for 64 processors. For nonuniform data with 64 processors we need more leaves than processors for good load balance; thus $L = 4$ giving $4^4 = 256$ leaves is sufficient for good load balance.

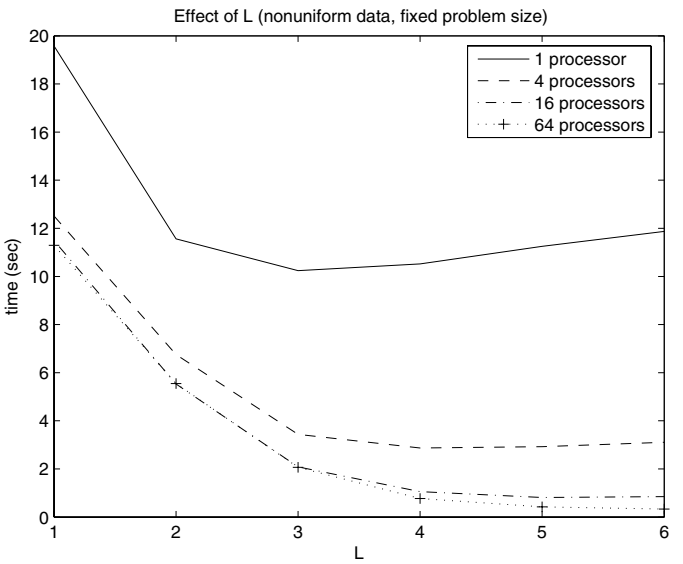
To more clearly demonstrate the effect of L on the load balance, see Figure 7, which shows the ratio of the maximum load to the average load per processor in terms of the number of support sets handled by each processor. Note that for 64 processors with $L = 3$ (which gives 64 leaves for the top-level tree) the ratio is still about five for nonuniform data; this means that without load balancing the algorithm would take roughly five times as long on 64 processors.

8.3. Scalability tests. The main reason for using parallel computers is not for solving a fixed problem faster but for solving larger problems in a short time. To test our algorithm for this, we carried out scalability tests, where the size of the data set increased proportionally to the number of processors, starting with 20,000 support sets and 30,000 test points for one processor. Since the size of the support sets decreases as the problem size becomes larger, the depth of the expanded MX-CIF quadtree must increase along with it. As a result, the time for the sequential algorithm is approximately $O(N \log N)$, where N is the number of support sets. This means that if the number of processors p is proportional to N, ideally the time for the parallel computation should go like $\beta_1 + \beta_2 \log(p)$, ignoring nonscalable communications and computations.

MPI supports sparse communication through `MPI_AlltoAllv`; however, the receiver must know in advance how many items to receive from each originating processor to use this routine. The time for each processor to get this information is $\Theta(p)$ because these numbers must be transmitted by `MPI_AlltoAll`, which assumes a communication pattern. The $\Theta(p)$ time per processor implies a total parallel cost of at least $\Omega(p^2)$ for this step. To be scalable the total parallel cost would have to be $O(p \log(p))$, since the sequential cost is $O(N \log(N))$, and the efficiency would not go to zero as $p \to \infty$. Note that this is an MPI implementation restriction. For large p, and most pairs of processors, the numbers of items to transmit would be zero; if MPI had a version of `MPI_AlltoAllv` that only required specifying the nonzero numbers to be transmitted and received, then this nonscalable communication step would be avoided.



(a)



(b)

FIG. 6. *Dependence of parallel times on L:* (a) *uniform data,* (b) *nonuniform data.*

In this sense the MPI model is at a disadvantage relative to the shared-memory framework (even if it is implemented on a distributed memory machine): In the shared-memory framework a processor can request an individual data item from a remote processor, without the remote processor knowing in advance that it has to receive a request from that processor. In the MPI framework the remote processor has to know in advance that it has to send the data item to the original processor.

It might be useful to have [illegible] all-to-all communication routines whereby sparse

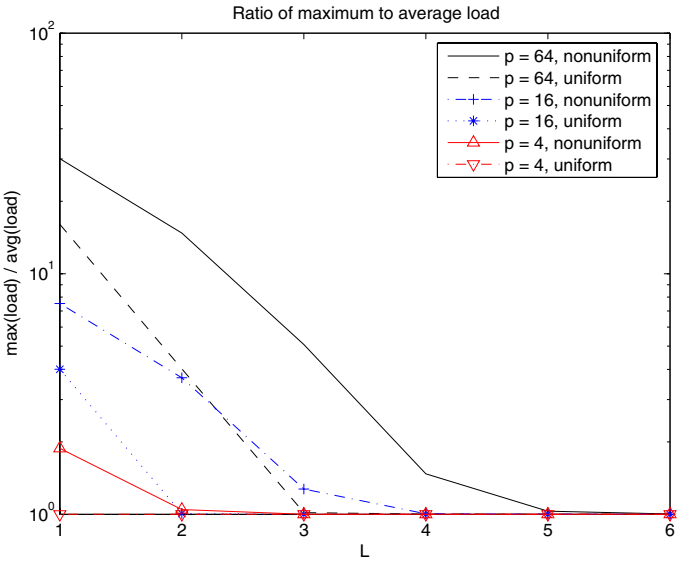


FIG. 7. *Effect of L on ratio of maximum load to average load.*

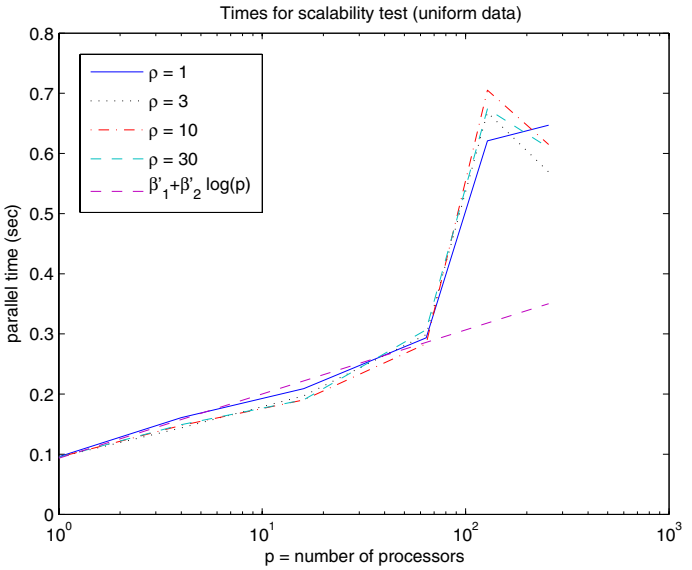
communication patterns can be efficiently handled. The MPI-2 standard [7] includes the concept of [illegible] and [illegible] of processors, which can be used to communicate between prespecified groups of processors. This can provide some similar functionality. However, these do not provide ad hoc sparse communication patterns. Such a communication routine is conceivable and should be efficiently implementable.

In spite of this nonscalability, the algorithm performs well in practice up to 64 or even 128 processors, as can be seen in Figure 8. In these tests we used $L = \log_4(p) + 2$ to ensure good load balance. The graph in Figure 8 looks a little odd with the times for 128 processors being greater than the times for 256 processors. This is because 128 not being a power of 4, we used $L = 6$, the same value as for 256 processors. The additional overhead in this case appears to have contributed to larger than expected times for 128 processors with nonuniform data.

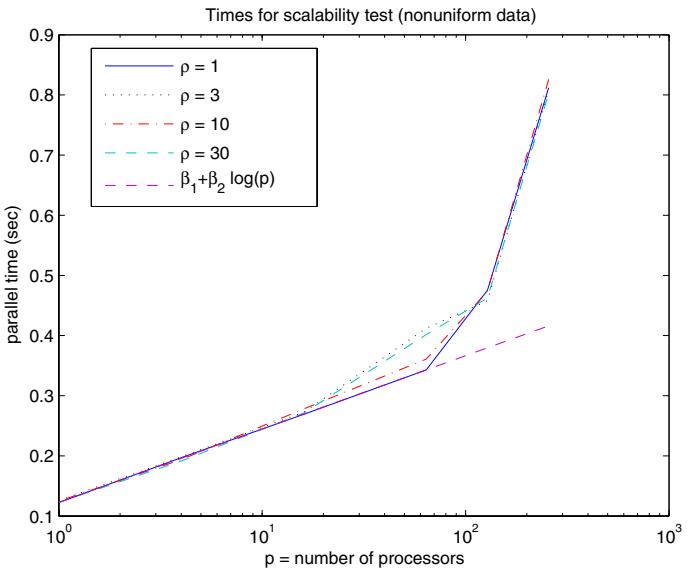
Also note that alongside the plots of the actual times, there are also plots of the form $\beta_1 + \beta_2 \log(p)$ obtained by linear regression on the minimum times for 1, 4, and 16 processors.

9. Conclusions. We have presented a parallel algorithm for dealing with a problem in computational geometry which is relevant for meshfree methods in PDEs. While this algorithm is not scalable as $p \to \infty$, the bottlenecks preventing scalability could be avoided by allowing true sparse all-to-all communication in message passing libraries. In spite of this difficulty, the algorithm appears to work in a scalable fashion with up to 64 processors.

Acknowledgments. We would like to thank the National Center for Supercomputing Applications (NCSA) at the University of Illinois at Urbana-Champaign for providing access, CPU time, and assistance on the Xeon cluster which was used for the computations reported here. A previous draft of this paper included results from Origin2000 machines at NCSA. We would also like to thank the referees for their suggestions for improvements to the paper.



(a)



(b)

FIG. 8. *Times for scalability tests:* (a) *uniform data,* (b) *nonuniform data.*

REFERENCES

[1] T. BELYTSCHKO, Y. KRONGAUZ, D. ORGAN, M. FLEMING, AND P. KRYSL, *Meshless methods: An overview and recent developments*, Comput. Methods Appl. Mech. Engrng., 139 (1996), pp. 3–47.

[2] R. L. GRAHAM, *Bounds on multiprocessing timing anomalies*, SIAM J. Appl. Math., 17 (1969), pp. 416–429.

[3] M. Griebel and M. A. Schweitzer, *A particle-partition of unity method—Part* II: *Efficient cover construction and reliable integration*, SIAM J. Sci. Comput., 23 (2002), pp. 1655–1682.
[4] X. Han, S. Oliveira, and D. Stewart, *Finding sets covering a point with application to mesh-free Galerkin methods*, SIAM J. Comput., 30 (2000), pp. 1368–1383.
[5] A. Huerta, T. Belytchko, S. Fernández-Méndez, and T. Rabczuk, *Meshfree methods*, in Encyclopedia of Computational Mechanics, Vol. 1, John Wiley & Sons, New York, 2004, pp. 279–309.
[6] O. Klaas and M. S. Shephard, *Automatic generation of octree-based three-dimensional discretizations for partition of unity methods*, Comput. Mech., 25 (2000), pp. 296–304.
[7] Message Passing Interface Forum, *MPI-2: Extensions to the Message-Passing Interface*, http://www.mpi-forum.org/docs/docs.html (1997).
[8] P. Pacheco, *Parallel Programming with MPI*, Morgan Kaufmann, San Francisco, 1997.
[9] H. Samet, *The Design and Analysis of Spatial Data Structures*, Ser. Comput. Sci., Addison–Wesley, Reading, MA, 1990.
[10] D. Shepard, *A two-dimensional function for irregularly spaced data*, in Proceedings of the 23rd National Conference of the ACM, 1968, pp. 517–524.
[11] M. Snir, S. Otto, S. Huss-Lederman, D. Walker, and J. Dongarra, *MPI: The Complete Reference*, MIT Press, Boston, 1996.
[12] V. V. Vazirani, *Approximation Algorithms*, Springer-Verlag, Berlin, 2001.

SIAM J. SCI. COMPUT.
Vol. 28, No. 4, pp. 1335–1360
© 2006 Society for Industrial and Applied Mathematics

HIGH-RESOLUTION FINITE VOLUME METHODS FOR DUSTY GAS JETS AND PLUMES*

MARICA PELANTI† AND RANDALL J. LEVEQUE†

Abstract. We consider a model for dusty gas flow that consists of the compressible Euler equations for the gas coupled to a similar (but pressureless) system of equations for the mass, momentum, and energy of the dust. These sets of equations are coupled via drag terms and heat transfer. A high-resolution wave-propagation algorithm is used to solve the equations numerically. The one-dimensional algorithm is shown to give agreement with a shock tube test problem in the literature. The two-dimensional algorithm has been applied to model explosive volcanic eruptions in which an axisymmetric jet of hot dusty gas is injected into the atmosphere and the expected behavior is observed at two different vent velocities. The methodology described here, with extensions to three dimensions and adaptive mesh refinement, is being used for more detailed studies of volcanic jet processes.

Key words. finite volume methods, high-resolution methods, volcanic flows, dusty gas, plumes, jets, shocks

AMS subject classifications. 65M06, 76T15

DOI. 10.1137/050635018

1. Introduction.

We study compressible gas dynamics coupled with a suspended particulate phase having small volume fraction but possibly large density per unit volume relative to the gas. We refer to the particulate phase as "dust" (though in some contexts it could represent liquid droplets) and to the mixture as a "dusty gas." Dusty gas flows arise in many applications, from industrial processes to geophysical flows. One particular application, which was the original motivation for our work, is the study of volcanic ash plumes and pyroclastic flows. The numerical approach developed and tested here should be directly applicable in other contexts as well.

We ignore viscosity in the present work and use the compressible Euler equations to model the conservation of mass, momentum, and energy in the gas phase, using an ideal gas equation of state. The particulate phase is modeled by a similar system of equations for the conservation of mass, momentum, and energy of the dust. We ignore interparticle collisions and assume there is no pressure component in the momentum equations, using the so-called pressureless dust or sticky particle equations for this flow (see section 2). These two sets of conservation laws are coupled together through source terms that model drag and heat transfer between the phases.

In the absence of coupling terms, we would have two decoupled sets of hyperbolic conservation laws. The pressureless dust equations are nonstrictly hyperbolic (since the sound speed is zero in the absence of pressure) and have a degenerate structure in which delta shocks can arise, shocks at which delta function singularities form in this mathematical idealization. The source term coupling smooths out singularities in the solution. The speed of sound in the dusty gas is lower than the sound speed of the pure gas phase and can be substantially lower at high dust densities if there

*Received by the editors July 2, 2005; accepted for publication (in revised form) January 18, 2006; published electronically August 7, 2006. This work was supported in part by DOE grant DE-FC02-01ER25474 and NSF grant DMS-0106511.
http://www.siam.org/journals/sisc/28-4/63501.html
†Department of Applied Mathematics, University of Washington, Box 352420, Seattle, WA 98195-2420 (pelanti@amath.washington.edu, rjl@amath.washington.edu).

is sufficient coupling, i.e., if the relaxation times of the momentum and heat transfer processes are sufficiently fast (see section 2.2). One effect of this is that dusty gas jets can be supersonic at relatively low velocities, and thus the shock wave structures typically seen in supersonic jets are often observed at lower velocities than might be expected. In particular, jets of ash in volcanic eruptions can exhibit such structures; see, e.g., [22].

The effect of gravity on both the gas and dust phases can also be included in the model when needed, as, for example, in the case of volcanic plumes, where gravity plays a substantial role. One question of particular interest in this context is whether an eruption column rises high into the atmosphere as a "Plinian column" and slowly disperses downwind, or whether the column collapses with the heavier dust phase flowing along the surface. This depends on the balance between the downward force of gravity and the upward momentum and buoyancy of the hot dense jet.

Applications to volcanic jets will be explored in more detail in a forthcoming paper [13] in which numerical results obtained with our approach will be compared with results from a similar model and different numerical method developed in the work of Neri, Esposti Ongaro, and co-workers [34, 36]. In the present paper we present the details of our numerical algorithm and several test problems.

The model is presented in more detail in section 2 and the numerical algorithm is developed in section 3. We use the wave-propagation algorithm as implemented in the CLAWPACK software [26]. This is a high-resolution finite volume algorithm for the hyperbolic conservation law portion of the equations, based on solving Riemann problems at each cell interface at every time step. We use the "f-wave formulation" [2, 29] of the algorithm in which the flux difference between neighboring cells is decomposed into waves that are then used to update the cell averages in each cell. This formulation is preferable for the pressureless dust portion of the hyperbolic system, where we use the algorithm developed in [30]. This formulation also allows us to incorporate the source term due to gravity in the gas dynamics portion of the hyperbolic system into the Riemann solver in such a way that hydrostatic balance in the atmosphere is well maintained in the absence of flow or dust (with the pressure gradient balancing the gravitational force on the gas). This is discussed in more detail in section 6, where gravity is introduced. The gravitational force in the dust equations and the source terms modeling interphase drag and heat transfer are all handled by a fractional step procedure, as described in section 5.

Our goal is to keep the physical model as simple as possible while still allowing simulation of some fundamental features of hot dilute dusty flows in the presence of gravity. In particular, we ignore viscosity within the gas phase and the effects of turbulence. In modeling the structure of dusty gas jets as considered here, this is a reasonable approximation.

Ultimately we hope that this approach will also be useful for modeling some aspects of volcanic eruptions in three dimensions over large domains to better understand hazards associated with active volcanos. For this it is desirable to have a simple model that is applicable on mapped grids that follow realistic terrain and to which adaptive mesh refinement can also be applied. The present model has been implemented on quadrilateral and hexahedral grids, and adaptive mesh refinement has been applied using both the AMRCLAW package included in CLAWPACK and the recently developed CHOMBO-CLAW package [5].

Extensive work has been done on dusty gas and related flows. The formulation that we adopt to describe the dynamics of this two-phase flow under the above assumptions follows especially the work of Harlow and Amsden [17], Sainsaulieu

[40, 42, 41], and Saito [43]. In particular, the physical and numerical model in [17] has been widely used in the field of geophysics to study the particle-laden flows ejected during volcanic eruptions, e.g., [46, 44, 10, 32, 36, 9]. A technique to obtain the macroscopic system of the presented model is the procedure described, e.g., in [19, 11], and applied by Sainsaulieu in [40], which consists in averaging the equations governing each single phase at the microscopic scale.

2. The equations for a dusty gas. We consider a two-phase flow composed of a gaseous carrier phase and a dispersed phase that can consist of either solid particles (dust) or liquid droplets, though we concentrate on the case of dust. In the following, the subscripts g and d will refer to the gas and the dust (or droplets), respectively. Each phase is modeled as a continuum described by macroscopic quantities, with the gas phase being compressible and the dispersed phase incompressible at a microscopic level (with microscopic density $\rho_d =$ constant within each particle or droplet). Moreover, we assume the carried phase is dilute; that is, its volume fraction $\vartheta_d \ll 1$.

2.1. One-dimensional equations. We first consider the one-dimensional equations in slightly more generality than we ultimately use computationally in order to discuss some of the physical and mathematical issues. In section 3 we discuss the numerical algorithms used in the context of the one-dimensional equations. In section 4 we present the relevant portion of these equations in three dimensions and discuss the multidimensional generalization of the numerical algorithm. Initially we ignore gravity and consider one-dimensional flow in the x-direction. In section 6 we discuss the manner in which gravity is incorporated into the algorithm.

In one space dimension the equations take the form

$$\frac{\partial}{\partial t}(\vartheta_g \rho_g) + \frac{\partial}{\partial x}(\vartheta_g \rho_g u_g) = 0, \tag{1a}$$

$$\frac{\partial}{\partial t}(\vartheta_g \rho_g u_g) + \frac{\partial}{\partial x}\left(\vartheta_g \rho_g u_g^2 + \vartheta_g p_g\right) = p_g \frac{\partial \vartheta_g}{\partial x} - D(u_g - u_d), \tag{1b}$$

$$\begin{aligned}\frac{\partial}{\partial t}(\vartheta_g \rho_g e_g) + \frac{\partial}{\partial x}((\vartheta_g \rho_g e_g + \vartheta_g p_g) u_g) \\ = -p_g \frac{\partial}{\partial x}((1-\vartheta_g) u_d) - D(u_g - u_d) u_d - \mathcal{Q}(T_g - T_d),\end{aligned} \tag{1c}$$

$$\frac{\partial}{\partial t}(\vartheta_d \rho_d) + \frac{\partial}{\partial x}(\vartheta_d \rho_d u_d) = 0, \tag{1d}$$

$$\frac{\partial}{\partial t}(\vartheta_d \rho_d u_d) + \frac{\partial}{\partial x}\left(\vartheta_d \rho_d u_d^2 + \varphi\right) = -\vartheta_d \frac{\partial p_g}{\partial x} + D(u_g - u_d), \tag{1e}$$

$$\begin{aligned}\frac{\partial}{\partial t}(\vartheta_d \rho_d e_d) + \frac{\partial}{\partial x}((\vartheta_d \rho_d e_d + \varphi) u_d) \\ = -\vartheta_d u_d \frac{\partial p_g}{\partial x} + D(u_g - u_d) u_d + \mathcal{Q}(T_g - T_d).\end{aligned} \tag{1f}$$

Here ρ_g, ρ_d are the microscopic material densities, ϑ_g, ϑ_d the volume fractions, u_g, u_d the velocities along the x-axis, e_g, e_d the specific total energies, and T_g, T_d the temperatures. Moreover, p_g denotes the gas pressure and φ denotes a pressure correction

for the dispersed phase that will be set to zero below. Finally, D and $\mathcal{Q}$ express a drag and heat transfer function, respectively. To complete the description we need closure relations and specific expressions for D and $\mathcal{Q}$.

The specific total energies are related to the specific internal energies ε_g, ε_d through

$$e_g = \varepsilon_g + \frac{1}{2}|u_g|^2 \quad \text{and} \quad e_d = \varepsilon_d + \frac{1}{2}|u_d|^2. \tag{2}$$

The gaseous phase is assumed to follow the ideal polytropic gas thermodynamic relations

$$p_g = (\gamma - 1)\rho_g \varepsilon_g, \quad \gamma = \text{constant}, \tag{3a}$$

$$\varepsilon_g = c_{vg} T_g, \quad c_{vg} = \text{constant}. \tag{3b}$$

The pressure φ associated to the dispersed phase is small in comparison to the gas pressure p_g, and considering negligible interaction between particles (droplets), it can be assumed to be zero, as in [17, 43]. However, a nonzero φ plays an important role from a mathematical point of view, since the homogeneous equations of the dispersed phase lose strict hyperbolicity under the hypothesis $\varphi \equiv 0$. Following [41], here a nonzero φ is viewed as a pressure correction that allows one to maintain the strict hyperbolicity of the dispersed phase system. We will consider φ a function of the macroscopic density only:

$$\varphi = \varphi(\beta), \quad \beta \equiv \vartheta_d \rho_d. \tag{4}$$

For instance, we can define φ based on the pressure law of the isothermal flow equations:

$$\varphi(\beta) = a_d^2 \beta, \tag{5}$$

with a_d small. In [40, 41], where in particular the suspended phase is considered made of droplets, φ is expressed in the form

$$\varphi = \varphi_0 \vartheta_d^{\,\delta} = \varphi_0 \left(\frac{\beta}{\rho_d}\right)^{\delta}, \tag{6}$$

where φ_0 is a constant proportional to the rest pressure of the gas flow on the droplets and $\delta = \frac{4}{3}$. Note that mathematically we recover (5) from (6) taking $\varphi_0 = \rho_d a_d^2$ and $\delta = 1$.

The specific internal energy of the dispersed phase is related to the temperature through a constant specific heat c_{vd}:

$$\varepsilon_d = c_{vd} T_d, \quad c_{vd} = \text{constant}. \tag{7}$$

System (1) is closed by the equations (2), (3), (4), (7), together with the algebraic constraint $\vartheta_g + \vartheta_d = 1$, and definitions for the drag and heat transfer functions D and $\mathcal{Q}$.

The drag function D has the form (e.g., [31])

$$D = \frac{3}{4} C_{\mathrm{d}} \frac{\beta \rho}{\rho_d d} |u_g - u_d|, \tag{8}$$

where d is the dust particle diameter and $C_{\rm d}$ is the drag coefficient, which we express as in [10]:

$$C_{\rm d} = \begin{cases} \frac{24}{Re}\left(1 + 0.15Re^{0.687}\right) & \text{if } Re < 1000, \\ 0.44 & \text{if } Re \geq 1000. \end{cases} \tag{9}$$

Above $Re = \frac{\rho\, d|u_g - u_d|}{\mu}$ is the Reynolds number and μ the dynamic viscosity of the gas.

The heat transfer function is (see, e.g., [23])

$$\mathcal{Q} = \frac{Nu\, 6\kappa_g \beta}{\rho_d d^2}, \tag{10}$$

where $Nu = 2 + 0.65Re^{1/2}Pr^{1/3}$ is the Nusselt number. Here $Pr = \frac{c_{pg}\mu}{\kappa_g}$ is the Prandtl number, κ_g the gas thermal conductivity, and c_{pg} the gas specific heat at constant pressure. In general, μ and κ_g will be assumed constant.

The momentum equations (1b), (1e) and the energy equations (1c), (1f) include nonconservative terms modeling exchange of momentum and energy related to the gas pressure gradient, namely $p_g \frac{\partial \vartheta_g}{\partial x}$, $p_g \frac{\partial}{\partial x}\left((1-\vartheta_g)u_d\right)$, $\vartheta_d \frac{\partial p_g}{\partial x}$, and $\vartheta_d \frac{\partial p_g}{\partial x} u_d$. Under the hypothesis $\vartheta_d \ll 1$, these are small and give a weak coupling. They can be neglected, as in [43]. These nonconservative terms pose a mathematical difficulty when we wish to allow for discontinuous solutions of the two-phase system. An extensive mathematical analysis of the nonconservative system (1) without drag and heat transfer terms has been performed in [42, 41] by Sainsaulieu, who gives a definition of shock wave solutions for the two-phase nonconservative system and derives explicit approximate jump conditions for these discontinuous solutions, showing that they are perturbations of order $\varepsilon = \frac{1}{\rho_d} \ll 1$ of the decoupled jump conditions of the homogeneous system. Other work on nonconservative terms in nonlinear problems can be found, for example, in [1, 4, 8, 25].

In this work we drop these nonconservative terms, so that only drag and heat conduction source terms appear on the right-hand side of equations (1). We can then simplify the notation by introducing

$$\rho = \vartheta_g \rho_g, \quad p = \vartheta_g p_g, \quad E = \vartheta_g \rho_g e_g, \quad \beta = \vartheta_d \rho_d, \quad \Omega = \vartheta_d \rho_d e_d. \tag{11}$$

The reduced equations can then be written as

$$q_t + f(q)_x = \psi(q), \tag{12}$$

where q is the solution vector, which we partition into q_g, containing the mass, momentum, and energy of the gas, and q_d, containing these quantities for the dust. Then (12) is partitioned as

$$\frac{\partial}{\partial t}\begin{bmatrix} q_g \\ q_d \end{bmatrix} + \frac{\partial}{\partial x}\begin{bmatrix} f_g(q_g) \\ f_d(q_d) \end{bmatrix} = \begin{bmatrix} \psi_g(q) \\ \psi_d(q) \end{bmatrix}, \tag{13}$$

with

$$q_g = \begin{bmatrix} \rho \\ \rho u_g \\ E \end{bmatrix}, \quad f_g(q_g) = \begin{bmatrix} \rho u_g \\ \rho u_g^2 + p \\ (E+p)u_g \end{bmatrix}, \quad \psi_g = \begin{bmatrix} 0 \\ -D(u_g - u_d) \\ -D(u_g - u_d)u_d - \mathcal{Q}(T_g - T_d) \end{bmatrix}, \tag{14a}$$

(14b)
$$q_d = \begin{bmatrix} \beta \\ \beta u_d \\ \Omega \end{bmatrix}, \quad f_d(q_d) = \begin{bmatrix} \beta u_d \\ \beta u_d^2 + \varphi \\ (\Omega + \varphi) u_g \end{bmatrix}, \quad \psi_d = \begin{bmatrix} 0 \\ D(u_g - u_d) \\ D(u_g - u_d) u_d + \mathcal{Q}(T_g - T_d) \end{bmatrix}.$$

Numerically we solve system (13) using a fractional step procedure in which we alternate between a time step on the homogeneous hyperbolic system $q_t + f(q)_x = 0$ and a time step on the source terms, the system of ODEs $q_t = \psi(q)$ in each grid cell. (The source term for gravity in the gas will be treated differently in section 6.)

Notice that q_g and q_d are coupled only through the source terms and so the hyperbolic part decouples into two smaller systems for the gas and dust separately. For the gas phase,

$$\frac{\partial q_g}{\partial t} + \frac{\partial f_g(q_g)}{\partial x} = 0, \tag{15}$$

we have the standard Euler equations for gas dynamics for an ideal gas. This system has three distinct characteristic speeds u_g and $u_g \pm c_g$ given by the eigenvalues of the Jacobian matrix $f_g'(q_g)$, where c_g is the sound speed of the gas, and $c_g = \sqrt{\gamma p_g/\rho}$ for an ideal gas.

The equations for the dust phase are similar:

$$\frac{\partial q_d}{\partial t} + \frac{\partial f_d(q_d)}{\partial x} = 0. \tag{16}$$

If we assume that the pressure correction φ depends only on β (e.g., (5)), then the energy equation simply models advection of energy at velocity u_d. This equation is coupled to the rest of the system, however, when the source terms are added due to energy transfer between the phases.

Dropping the energy equation from the hyperbolic system for the dust, we obtain the reduced system

$$\frac{\partial}{\partial t} \begin{bmatrix} \beta \\ \beta u_d \end{bmatrix} + \frac{\partial}{\partial x} \begin{bmatrix} \beta u_d \\ \beta u_d^2 + \varphi \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \tag{17}$$

The Jacobian matrix for this sytem has eigenvalues $u_d \pm c_d$, where $c_d = \sqrt{\varphi'(\beta)}$. For example, if we use the isothermal relation (5), $\varphi(\beta) = a_d^2 \beta$, then $c_d = a_d$. If we ignore interparticle collisions, then we want to consider the "pressureless" limit $a_d \to 0$, in which case the system (17) becomes nonstrictly hyperbolic with both eigenvalues equal to u_d, and the system (16) has all three eigenvalues equal to u_d. The Jacobian matrix lacks a complete set of eigenvectors in this case, and weak solutions to the equations can contain delta shocks—shock waves at which a delta function singularity of accumulated dust also arises [30].

The term "pressureless" is something of a misnomer when the limiting process described above is really intended, as we are considering the limit where interparticle collision effects play a role, but only when the density becomes very large. When the particulate phase is sufficiently dilute, the details of this interaction are unimportant and we can set $a_d = 0$. The drag coupling of these equations to the gas dynamics equations might be expected to yield a regularization effect preventing the formation of delta function singularities in the coupled system. We do not know of any theoretical work in this direction and believe that it would be an interesting topic to pursue.

Numerically we do not observe delta shocks in the coupled system, although numerical viscosity is also playing a role, and at any rate exact singularities could not arise in the numerical solution. More importantly, in practice the dust densities do not grow to levels that cause any difficulties near shocks.

2.2. The mixture speed of sound. Gas flows carrying a particulate phase may exhibit a sound speed that is much lower than the sound speed of the pure gas. Here we informally derive an expression for the speed of sound of the mixture of gas and particles, based on the model (1) with $\varphi \equiv 0$, under the assumption that momentum and energy exchange between the gas and dust occurs rapidly enough so that they are in mechanical and thermal equilibrium (*instantaneous relaxation* hypothesis). For example, this is often approximately satisfied in vast regions of the physical domain where volcanic processes occur. Note that we will not assume in general equilibrium between the two phases in our numerical method, but the expression of the mixture sound speed derived in this special case can be useful to estimate a posteriori the flow regime (subsonic/supersonic). Moreover we employ this expression in [38] to show that solving the coupled system handling the source terms by a fractional step approach is successful even in this equilibrium limit.

We consider the propagation of small-amplitude pressure waves (acoustic waves) against a steady background with constant velocities $u_{g0} = u_{d0} = 0$, constant densities $\rho_0 = \vartheta_{g0}\rho_{g0}$, $\beta_0 = \vartheta_{d0}\rho_d = (1 - \vartheta_{g0})\rho_d$ (recall that $\rho_d = \text{const.}$), and constant gas pressure. We also assume that $c_{vd}\beta_0 \gg c_{vg}\rho_0$ so that the dust contains much more thermal energy than the gas. Since we are assuming the temperature equilibrates rapidly between the gas and dust and we are considering small perturbations about a constant state, it is then reasonable to assume that the dust acts as a heat reservoir that keeps the gas at a constant temperature as the acoustic wave propagates. We can then drop the corresponding source term and simply consider isothermal behavior. The effective equation of state is then

$$p_g = a_g^2 \rho_g, \quad a_g^2 = R\bar{T}_g, \tag{18}$$

where a_g represents the isothermal sound speed of the pure gaseous phase and $\bar{T}_g$ denotes the constant temperature of the two-phase mixture. Under the above assumption of isothermal flow, and using (18), equations (1) give the reduced system

$$\frac{\partial}{\partial t}(\vartheta_g \rho_g) + \frac{\partial}{\partial x}(\vartheta_g \rho_g u_g) = 0, \tag{19a}$$

$$\frac{\partial}{\partial t}(\vartheta_g \rho_g u_g) + \frac{\partial}{\partial x}\left(\vartheta_g \rho_g u_g^2\right) + a_g^2 \vartheta_g \frac{\partial \rho_g}{\partial x} = -D(u_g - u_d), \tag{19b}$$

$$\frac{\partial}{\partial t}((1 - \vartheta_g)\rho_d) + \frac{\partial}{\partial x}((1 - \vartheta_g)\rho_d u_d) = 0, \tag{19c}$$

$$\frac{\partial}{\partial t}((1 - \vartheta_g)\rho_d u_d) + \frac{\partial}{\partial x}\left((1 - \vartheta_g)\rho_d u_d^2\right) + a_g^2(1 - \vartheta_g)\frac{\partial \rho_g}{\partial x} + D(u_g - u_d), \tag{19d}$$

where we have also used the algebraic constraint $\vartheta_g + \vartheta_d = 1$. Considering small perturbations with respect to background variables, we write

$$\rho_g = \rho_{g0} + \tilde{\rho_g} \qquad \text{and} \qquad \vartheta_g = \vartheta_{g0} + \tilde{\vartheta_g}. \tag{20}$$

Linearizing (19), and dividing the resulting continuity equations of the two phases by the corresponding background microscopic densities, we obtain

$$\frac{\vartheta_{g0}}{\rho_{g0}}\frac{\partial\tilde{\rho_g}}{\partial t}+\frac{\partial\tilde{\vartheta_g}}{\partial t}+\vartheta_{g0}\frac{\partial u_g}{\partial x}=0, \tag{21a}$$

$$\vartheta_{g0}\rho_{g0}\frac{\partial u_g}{\partial t}+a_g^2\vartheta_{g0}\frac{\partial\tilde{\rho_g}}{\partial x}=-D(u_g-u_d), \tag{21b}$$

$$-\frac{\partial\tilde{\vartheta_g}}{\partial t}+(1-\vartheta_{g0})\frac{\partial u_d}{\partial x}=0, \tag{21c}$$

$$(1-\vartheta_{g0})\rho_d\frac{\partial u_d}{\partial t}+a_g^2(1-\vartheta_{g0})\frac{\partial\tilde{\rho_g}}{\partial x}=D(u_g-u_d). \tag{21d}$$

Adding now (21b) and (21d) eliminates the drag source term, and if we assume that this acts sufficiently fast that velocity perturbations satisfy $u_g = u_d \equiv u$, then the resulting equation along with the sum of (21a) and (21c) gives a reduced system of two equations for $\tilde{\rho}_g$ and u:

$$\frac{\vartheta_{g0}}{\rho_{g0}}\frac{\partial\tilde{\rho_g}}{\partial t}+\frac{\partial u}{\partial x}=0, \tag{22a}$$

$$(\vartheta_{g0}\rho_{g0}+(1-\vartheta_{g0})\rho_d)\frac{\partial u}{\partial t}+a_g^2\frac{\partial\tilde{\rho_g}}{\partial x}=0. \tag{22b}$$

This system has wave speeds $\pm c_{\mathrm{m}}$, where

$$c_{\mathrm{m}}=a_g\sqrt{\frac{\rho_{g0}}{\vartheta_{g0}(\vartheta_{g0}\rho_{g0}+\vartheta_{d0}\rho_d)}} \tag{23}$$

is the mixture sound speed in the considered case of equilibrium flow. This expression can be derived more generally for the case in which the suspended phase is also compressible and governed by a nonzero pressure law, see, for example, [45, 11].

Under the assumption of a dilute dust phase, we can approximate $\vartheta_{g0} \approx 1$, and hence $\rho_{g0} \approx \rho_0$. Therefore, $c_{\mathrm{m}} \approx a_g\sqrt{\frac{\rho_0}{\rho_0+\beta_0}}$. This expression can also be obtained directly with a procedure analogous to that described above by neglecting in (1) the terms modeling momentum exchange due to the gas pressure gradient, which is again valid if $\vartheta_{g0} \approx 1$. This in particular suggests that neglecting pressure gradient terms in the hypothesis $\vartheta_d \ll 1$ still allows us to correctly model the propagation of acoustic waves.

Note that $c_{\mathrm{m}} < a_g < c_g = (\gamma R\bar{T}_g)^{1/2}$, where c_g is the isentropic pure gas sound speed (at same T_g) and, moreover, c_{m} can be significantly lower than a_g and c_g if β_0 is large compared to ρ_0.

It is interesting that this mixture sound speed arises only from the source terms coupling the gas and dust equations and can be significantly lower than the wave speeds given by the hyperbolic part of the system of equations. Numerical results presented in [38] show that using a fractional step approach in which the source terms are decoupled from the hyperbolic system, as described below, does give the correct behavior of the coupled system and the correct sound speed.

3. The numerical algorithm for the hyperbolic portion. To solve the homogeneous hyperbolic portion of the system (1) we use the wave-propagation algorithm described in [27, 29]. This is a Godunov-type finite volume method in which we first solve the Riemann problem at each interface between grid cells, i.e., the conservation law with piecewise constant initial data given by the current cell averages Q^n_{i-1} and Q^n_i at time t_n. The waves arising from this Riemann solution are used to update the cell average on either side. These waves are also used to define second-order correction terms based on Taylor series expansion. Near discontinuities such as shocks, however, these corrections do not improve the accuracy but instead lead to nonphysical oscillations in the numerical approximation. Hence limiters must be applied to the waves before using them in the correction terms in order to avoid oscillations and obtain high-resolution results. We use the "f-wave" formulation of these methods discussed in [2, 29]. With this approach the Riemann solution is represented by a decomposition of the flux difference $f(Q_i) - f(Q_{i-1})$ as a linear combination of eigenvectors of an approximate Jacobian matrix. Both the first-order Godunov updates and the second-order correction terms can be defined in terms of these f-waves, and the limiters are now applied to these f-waves before computing the correction terms. This formulation of the algorithm for the gas phase will be advantageous when we add gravity in section 6. For the dust phase it helps to simplify the treatment of the nonstrictly hyperbolic Riemann solution.

Recall that the hyperbolic equations decouple into separate systems (15) for the gas phase and (16) for the dust phase. To approximately solve the Riemann problem in the gas phase between states Q^{L}_g and Q^{R}_g in adjacent grid cells, we use the Roe linearization [29, 39] to define eigenvectors r^1, r^2, r^3 based on Roe averages of the solution quantities. We decompose

$$f(Q^{\text{R}}_g) - f(Q^{\text{L}}_g) = \sum_{p=1}^{3} \mathcal{Z}^p_g, \tag{24}$$

where each f-wave $\mathcal{Z}^p_g$ is a scalar multiple of r^p.

For the dust phase we use the algorithm presented in [30]. For the pressureless dust equations the solution to the Riemann problem consists of either a single delta shock if $u^{\text{L}}_d > u^{\text{R}}_d$ or a pair of waves with a vacuum between them if $u^{\text{L}}_d < u^{\text{R}}_d$. With the f-wave formulation we generally use only a single wave in either case with magnitude $\mathcal{Z}_d = f(Q^{\text{R}}_d) - f(Q^{\text{L}}_d)$ and speed

$$\hat{u}_d = \frac{\sqrt{\beta^{\text{L}}}\, u^{\text{L}}_d + \sqrt{\beta^{\text{R}}}\, u^{\text{R}}_d}{\sqrt{\beta^{\text{L}}} + \sqrt{\beta^{\text{R}}}}. \tag{25}$$

This is the usual Roe average for the dust velocity, based on the dust density, and it can also be shown to be the correct delta-shock propagation speed in the pressureless equations [30]. The only time we use two waves is if $u^{\text{L}}_d < 0 < u^{\text{R}}_d$, in which case they are spreading out with a vacuum state at the interface. Then we take

$$\begin{aligned} \mathcal{Z}^1_d &= -f_d(Q^{\text{L}}_d), \qquad s^1 = u^{\text{L}}_d, \\ \mathcal{Z}^2_d &= f_d(Q^{\text{R}}_d), \qquad s^2 = u^{\text{R}}_d, \end{aligned} \tag{26}$$

as the waves and speeds. For more details, see [30].

TABLE 1
Nomenclature.

ϑ_g, ϑ_d = volume fractions, $\vartheta_g + \vartheta_d = 1$, $\vartheta_d \ll 1$;
ρ_g, ρ_d = material microscopic mass densities (ρ_d = constant);
$\rho = \vartheta_g \rho_g$ = gas macroscopic density;
$\beta = \vartheta_d \rho_d$ = dust macroscopic density (concentration);
p_g = gas pressure, $p = \vartheta_g p_g$;
$\mathcal{V}_g = (u_g, v_g, w_g)^{\mathrm{T}}$, $\mathcal{V}_d = (u_d, v_d, w_d)^{\mathrm{T}}$ = vectorial velocities;
ε_g, ε_d = specific internal energies;
$e_g = \varepsilon_g + \frac{1}{2}\lvert\mathcal{V}_g\rvert^2$, $e_d = \varepsilon_d + \frac{1}{2}\lvert\mathcal{V}_d\rvert^2$ = specific total energies;
$E = \vartheta_g \rho_g e_g$, $\Omega = \vartheta_d \rho_d e_d$ = total energies per unit volume;
T_g, T_d = temperatures;
R = gas constant;
$\gamma = c_{pg}/c_{vg}$ = ratio of specific heats for gas;
c_{vd} = dust specific heat;
μ = gas dynamic viscosity;
κ_g = gas thermal conductivity;
$\mathfrak{g} = (0, 0, -g)^{\mathrm{T}}$ = gravity acceleration (z-direction);
D = drag function;
d = dust particle diameter;
C_{d} = drag coefficient;
$\mathcal{Q}$ = heat transfer function;
Re = Reynolds number;
Nu = Nusselt number;
Pr = Prandtl number.

4. The multidimensional equations and algorithm. The multidmensional equations we use have the form

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathcal{V}_g) = 0, \tag{27a}$$

$$\frac{\partial}{\partial t}(\rho \mathcal{V}_g) + \nabla \cdot (\rho \mathcal{V}_g \otimes \mathcal{V}_g + pI) = -D(\mathcal{V}_g - \mathcal{V}_d), \tag{27b}$$

$$\frac{\partial E}{\partial t} + \nabla \cdot ((E + p)\mathcal{V}_g) = -D(\mathcal{V}_g - \mathcal{V}_d) \cdot \mathcal{V}_d - \mathcal{Q}(T_g - T_d), \tag{27c}$$

$$\frac{\partial \beta}{\partial t} + \nabla \cdot (\beta \mathcal{V}_d) = 0, \tag{27d}$$

$$\frac{\partial}{\partial t}(\beta \mathcal{V}_d) + \nabla \cdot (\beta \mathcal{V}_d \otimes \mathcal{V}_d) = D(\mathcal{V}_g - \mathcal{V}_d), \tag{27e}$$

$$\frac{\partial \Omega}{\partial t} + \nabla \cdot (\Omega \mathcal{V}_d) = D(\mathcal{V}_g - \mathcal{V}_d) \cdot \mathcal{V}_d + \mathcal{Q}(T_g - T_d). \tag{27f}$$

Here $\mathcal{V}_g = (u_g, v_g, w_g)^{\mathrm{T}}$ and $\mathcal{V}_d = (u_d, v_d, w_d)^{\mathrm{T}}$ are the velocity vectors (in the three-dimensional case). The closure relations are the same as those presented in section 2, and the drag and heat transfer terms are also unchanged except that now

$$D = \frac{3}{4} C_{\mathrm{d}} \frac{\beta \rho}{\rho_d d} |\mathcal{V}_g - \mathcal{V}_d| \tag{28}$$

depends on the Euclidean norm of the velocity difference, as does the Reynolds number $Re = (\rho\, d|\mathcal{V}_g - \mathcal{V}_d|)/\mu$. See Table 1 for a summary of the nomenclature used.

The finite volume method is extended to multidimensions using the wave-propagation approach described in [24, 27, 29] and implemented in CLAWPACK. We assume

the grid is logically rectangular, but a grid mapping can be incorporated so that a general quadrilateral (in two dimensions) or hexahedral (in three dimensions) grid is used. The two-dimensional case is discussed in detail in [29].

In each time step, one-dimensional Riemann problems are solved normal to each cell interface and the resulting waves are used to update the cell averages on either side. For the dusty gas equations, the one-dimensional Riemann problem in an arbitrary direction takes the same form as the Riemann problem in the x-direction discussed in section 3, using the normal components of the gas and dust velocities in each grid cell to define the data. Jumps in transverse velocities advect with the flow, at the velocity of the contact discontinuity in the gas equations or of the delta shock in the dust phase. The waves resulting from these normal Riemann solutions are also used in second-order correction terms, just as in one dimension after suitable limiters have been applied to avoid nonphysical oscillations.

A "transverse Riemann solver" is also incorporated that takes the information propagating into each grid cell and splits this into eigenvectors of the flux Jacobian matrix (evaluated in the cell) in the direction orthogonal to each of the adjacent cell faces (for a nonorthogonal mapped grid this must be done separately for each adjacent face). Use of these transverse Riemann solvers increases the accuracy of the multidimensional method by modeling cross-derivative terms needed in the Taylor series expansion, and also improves stability so that a Courant number close to 1 can typically be used (see [24] for a detailed discussion of accuracy and stability issues in three dimensions).

5. Source terms for drag and heat transfer. We also use a fractional step approach in applying the source terms for drag and heat transfer, first solving the ODEs arising from the drag terms over a time step Δt and then solving the ODEs for heat transfer. For the drag terms we have the system of ODEs

$$\frac{\partial \rho}{\partial t} = 0, \tag{29a}$$

$$\frac{\partial}{\partial t}(\rho \mathcal{V}_g) = -A|\mathcal{V}_g - \mathcal{V}_d|(\mathcal{V}_g - \mathcal{V}_d), \tag{29b}$$

$$\frac{\partial E}{\partial t} = -A|\mathcal{V}_g - \mathcal{V}_d|(\mathcal{V}_g - \mathcal{V}_d)\cdot \mathcal{V}_d, \tag{29c}$$

$$\frac{\partial \beta}{\partial t} = 0, \tag{29d}$$

$$\frac{\partial}{\partial t}(\beta \mathcal{V}_d) = A|\mathcal{V}_g - \mathcal{V}_d|(\mathcal{V}_g - \mathcal{V}_d), \tag{29e}$$

$$\frac{\partial \Omega}{\partial t} = A|\mathcal{V}_g - \mathcal{V}_d|(\mathcal{V}_g - \mathcal{V}_d)\cdot \mathcal{V}_d, \tag{29f}$$

with initial data ρ^0, $(\rho\mathcal{V}_g)^0$, E^0, β^0, $(\beta\mathcal{V}_d)^0$, Ω^0 at the beginning of the time step that come from the hyperbolic solver.

The densities ρ and β are constant in this step. We also assume that the Reynolds number is constant in time, so that the drag coefficient is equal to its initial value, $C_{\mathrm{d}} = C_{\mathrm{d}}^0$. Then A is constant in time and equal to

$$A^0 = \frac{3}{4} C_{\mathrm{d}}^0 \frac{\beta^0 \rho^0}{\rho_d d}. \tag{30}$$

Then we can write (29b), (29e) for the momentum of the two phases as

$$\frac{\partial \rho^0 \mathcal{V}_g}{\partial t} = -A^0 |\mathcal{V}_g - \mathcal{V}_d|(\mathcal{V}_g - \mathcal{V}_d), \tag{31a}$$

$$\frac{\partial \beta^0 \mathcal{V}_d}{\partial t} = A^0 |\mathcal{V}_g - \mathcal{V}_d|(\mathcal{V}_g - \mathcal{V}_d). \tag{31b}$$

These equations can be solved exactly to obtain

$$(\rho \mathcal{V}_g)(\Delta t) = (\rho \mathcal{V}_g)^0 + \frac{\mathcal{V}_g^0 - \mathcal{V}_d^0}{\xi_D^0} \left[\frac{1}{A^0 \xi_D^0 |\mathcal{V}_g^0 - \mathcal{V}_d^0| \Delta t + 1} - 1 \right], \tag{32a}$$

$$(\beta \mathcal{V}_d)(\Delta t) = (\beta \mathcal{V}_d)^0 - \frac{\mathcal{V}_g^0 - \mathcal{V}_d^0}{\xi_D^0} \left[\frac{1}{A^0 \xi_D^0 |\mathcal{V}_g^0 - \mathcal{V}_d^0| \Delta t + 1} - 1 \right], \tag{32b}$$

where we have introduced

$$\xi_D = \frac{1}{\rho} + \frac{1}{\beta}. \tag{33}$$

Once the momentum equations have been solved, we can use the form of the energy equations to calculate the corresponding changes in energy of each phase. The right-hand side of (29f) corresponds exactly to the change in kinetic energy of the dust corresponding to the change in momentum, and hence the internal energy of the dust remains constant in this step. The dust energy Ω is simply updated by the change in kinetic energy resulting from the momentum update.

The right-hand side of the gas energy equation (29c) is just the negative of the right-hand side of (29f), as required by conservation of total energy. Hence the gas energy E is updated by the negative of the update to Ω calculated above in order to leave the total energy unchanged. This energy change models both the change in kinetic energy in the gas and also a change in internal energy due to drag dissipation and the resulting heating of the gas.

Note that if the time step is large relative to the drag relaxation time, then the gas and dust velocities determined by the momentum updates (32) both approach the common equilibrium value

$$\mathcal{V}_{\text{eq}} = \frac{\rho^0 \mathcal{V}_g^0 + \mathcal{V}_d^0 \beta^0}{\rho^0 + \beta^0}. \tag{34}$$

The source terms for heat transfer have the form

$$\frac{\partial \rho}{\partial t} = 0, \tag{35a}$$

$$\frac{\partial}{\partial t}(\rho \mathcal{V}_g) = 0, \tag{35b}$$

$$\frac{\partial E}{\partial t} = -\mathcal{Q}(T_g - T_d), \tag{35c}$$

$$\frac{\partial \beta}{\partial t} = 0, \tag{35d}$$

$$\frac{\partial}{\partial t}(\beta \mathcal{V}_d) = 0, \tag{35e}$$

$$\frac{\partial \Omega}{\partial t} = \mathcal{Q}(T_g - T_d), \tag{35f}$$

with initial data ρ^0, β^0, $(\rho\mathcal{V}_g)^0$, $(\rho\mathcal{V}_d)^0$, E^0, Ω^0 coming from the result of applying the drag terms, as described above. In this step the densities and momenta remain constant and only the energies change due to heat transfer between the dust and gas. It follows that the Reynolds and Nusselt numbers are constant; hence $\mathcal{Q}$ is constant and equal to its initial value $\mathcal{Q}^0$:

$$\mathcal{Q}^0 = \frac{Nu^0\, 6\kappa_g \beta^0}{\rho_d d^2}. \tag{36}$$

By using $\varepsilon_g = c_{vg} T_g$, $\varepsilon_d = c_{vd} T_d$, where c_{vg}, c_{vd} are the specific heats at constant volume, we obtain a system of two ODEs for the temperature of each phase,

$$\frac{\partial T_g}{\partial t} = -\frac{1}{\rho^0 c_{vg}} \mathcal{Q}^0 (T_g - T_d), \tag{37a}$$

$$\frac{\partial T_d}{\partial t} = \frac{1}{\beta^0 c_{vd}} \mathcal{Q}^0 (T_g - T_d). \tag{37b}$$

These can be solved and used to compute the energy updates:

$$E(\Delta t) = E^0 + \frac{T_g^0 - T_d^0}{\xi_{\mathcal{Q}}^0} \left[e^{-\mathcal{Q}^0 \xi_{\mathcal{Q}}^0 \Delta t} - 1 \right], \tag{38a}$$

$$\Omega(\Delta t) = \Omega^0 - \frac{T_g^0 - T_d^0}{\xi_{\mathcal{Q}}^0} \left[e^{-\mathcal{Q}^0 \xi_{\mathcal{Q}}^0 \Delta t} - 1 \right], \tag{38b}$$

where

$$\xi_{\mathcal{Q}} = \frac{1}{\rho c_{vg}} + \frac{1}{\beta c_{vd}}. \tag{39}$$

If the time step is long compared to the relaxation time for heat transfer, the two gases reach the common equilibrium temperature

$$T_{\text{eq}} = \frac{\rho^0 c_{vg} T_g^0 + \beta^0 c_{vd} T_d^0}{\rho^0 c_{vg} + \beta^0 c_{vd}}. \tag{40}$$

6. Source terms for gravity. With gravity, the equations (27) take the form

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathcal{V}_g) = 0, \tag{41a}$$

$$\frac{\partial}{\partial t} (\rho \mathcal{V}_g) + \nabla \cdot (\rho \mathcal{V}_g \otimes \mathcal{V}_g + pI) = \rho \mathfrak{g} - D(\mathcal{V}_g - \mathcal{V}_d), \tag{41b}$$

$$\frac{\partial E}{\partial t} + \nabla \cdot ((E + p)\mathcal{V}_g) = \rho \mathcal{V}_g \cdot \mathfrak{g} - D(\mathcal{V}_g - \mathcal{V}_d) \cdot \mathcal{V}_d - \mathcal{Q}(T_g - T_d), \tag{41c}$$

$$\frac{\partial \beta}{\partial t} + \nabla \cdot (\beta \mathcal{V}_d) = 0, \tag{41d}$$

$$\frac{\partial}{\partial t} (\beta \mathcal{V}_d) + \nabla \cdot (\beta \mathcal{V}_d \otimes \mathcal{V}_d) = \beta \mathfrak{g} + D(\mathcal{V}_g - \mathcal{V}_d), \tag{41e}$$

$$\frac{\partial \Omega}{\partial t} + \nabla \cdot (\Omega \mathcal{V}_d) = \beta \mathcal{V}_d \cdot \mathfrak{g} + D(\mathcal{V}_g - \mathcal{V}_d) \cdot \mathcal{V}_d + \mathcal{Q}(T_g - T_d), \tag{41f}$$

where $\mathfrak{g} = (0, 0, -g)^{\mathrm{T}}$ is the gravitational acceleration in the vertical direction. The gravity term for the dust equations is handled in the fractional step procedure along with the source terms for drag and heat transfer. Since the mass is constant over a time step in solving these ODEs, the equation for momentum is easily solved by modifying the vertical velocity by $-g\beta\Delta t$ and then adjusting the energy to reflect this change in kinetic energy.

The gravity source term for the gas could be handled in the same way. However, the gas also has pressure. In hydrostatic equilibrium (as in a stationary atmosphere) there is a pressure gradient that exactly balances the gravity source term. With a fractional step approach, the jump in pressure leads to nontrivial waves in the Riemann solution computed across each cell interface, which in turn causes a change in the cell averages during the hyperbolic step. The gravitational source term then leads to a compensating change in the cell averages that ideally would exactly cancel out the changes generated by the Riemann solution. In practice this will not happen and the numerical algorithm will not maintain a steady state. In fact the errors generated can be quite significant.

Instead we use an approach that does a much better job of maintaining hydrostatic balance and that applies the second-order correction terms and limiters in the hyperbolic solver to deviations from the steady state rather than to waves that arise artificially from the pressure gradient.

We take advantage of the f-wave formulation described in section 3 and incorporate an appropriate component of the gravity source term into the jump in flux across each interface before splitting this jump into waves. In place of (24), we split

$$f_g(Q_g^{\mathrm{R}}) - f_g(Q_g^{\mathrm{L}}) - \Psi = \sum_{p=1}^{3} \mathcal{Z}^p \tag{42}$$

into f-waves $\mathcal{Z}^p$. In solving this Riemann problem we have left and right states given by the cell averages in the cells on the two sides of the interface, with the velocities u_g^{L} and u_g^{R} taken to be the gas velocity normal to the interface from each side. On the left side, for example, we have

$$Q_g^{\mathrm{L}} = \begin{bmatrix} \rho^{\mathrm{L}} \\ \rho^{\mathrm{L}} u_g^{\mathrm{L}} \\ E^{\mathrm{L}} \end{bmatrix}, \qquad f(Q_g^{\mathrm{L}}) = \begin{bmatrix} \rho^{\mathrm{L}} u_g^{\mathrm{L}} \\ \rho^{\mathrm{L}} (u_g^{\mathrm{L}})^2 + p^{\mathrm{L}} \\ u^{\mathrm{L}}(E^{\mathrm{L}} + p^{\mathrm{L}}) \end{bmatrix}. \tag{43}$$

The source term for this Riemann problem that we use in (42) is then

$$\Psi = \Delta z \begin{bmatrix} 0 \\ -g(\rho^{\mathrm{L}} + \rho^{\mathrm{R}})/2 \\ -g(\rho^{\mathrm{L}} u_g^{\mathrm{L}} + \rho^{\mathrm{R}} u_g^{\mathrm{R}})/2 \end{bmatrix}. \tag{44}$$

Here $\Delta z = z^{\mathrm{R}} - z^{\mathrm{L}}$ is the difference in the vertical coordinate of the centroids of the two cells adjacent to this interface. In the hydrostatic case $u^{\mathrm{L}} = u^{\mathrm{R}} = 0$, and we expect the jump in pressure between the cell averages to approximately equal $-\Delta z\, g(\rho^{\mathrm{L}} + \rho^{\mathrm{R}})/2$, with this cancellation occurring before the waves are generated for use in the hyperbolic solver.

This approach to incorporating the gravitational source term into the Riemann solver is related to other so-called well-balanced methods for solving conservation laws with source terms in which the source terms balance the divergence of the flux in a

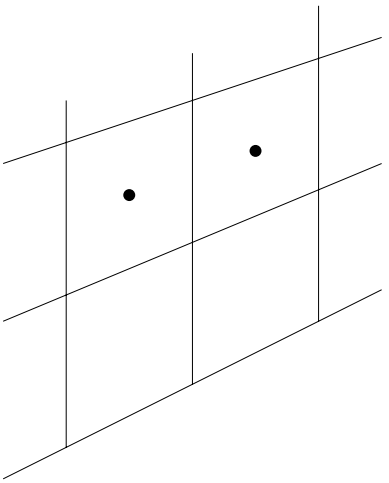


FIG. 1. *A two-dimensional mapped grid fitted to surface topography.*

nontrivial steady state. See, for example, [3, 6, 14, 15, 16, 20, 21, 28] for some other approaches. We note that the other source terms in our equations (drag, heat transfer, and the geometric source terms for axisymmetry) are nonzero only in transient regions of the flow and thus we do not need to worry about maintaining steady states and the fractional step approach we have adopted for these terms seems to work well in general. (In other applications there can exist steady states with nonzero velocities and then balancing these against the drag and heat transfer sources may be important as well.)

For applications to volcanic flows we will use a mapped grid that conforms to the ground surface, of the type shown in Figure 1 in two dimensions (blown up near the surface). A quadrilateral logically rectangular grid is used in which the horizontal grid lines are interpolated between the surface topography and a fixed upper elevation, while the vertical grid lines are still vertical. It is important to note that the gravitational source term will generally come into every Riemann solve on such a grid, even those in the x-direction that are normal to a vertical cell interface (e.g., between the two cells whose centroids are marked by dots in the figure). Since gravity does not act normal to this interface, one might be tempted to drop the source term in this normal Riemann solve. However, the difference in elevation of the cell centroids means that there will be a pressure difference in the steady state solution that must be cancelled by the source term. The expression (44) can be more formally justified by noting that the gravitational source is the gradient of the gravitational potential and hence is a covariant tensor. The directional derivative of this tensor in the coordinate direction comes into the Riemann problem across this interface, along with the normal components of the contravariant velocity.

7. Numerical experiments. We present numerical results of some test problems obtained by applying the presented dusty gas model. Second order corrections are performed in the solution of the hyperbolic portion of the system by using the monotonized centered (MC) limiter for the gas equations, and the minmod limiter for the dust equations.

In section 7.1 we consider a one-dimensional test with no gravity taken from [43].

In section 7.2 we report results of two-dimensional computations including gravity effects in the context of simulation of volcanic processes. Results on different bottom

TABLE 2
Initial data for Saito's experiment.

Variable	Driver section	Driven section
p/p_{ref}	10.0	1.0
ρ/ρ_{ref}	10.0	1.0
β/ρ_{ref}	0.0001	1.0
u_g/u_{ref}	0.0	0.0
u_d/u_{ref}	0.0	0.0
T_g/T_{ref}	1.0	1.0
T_d/T_{ref}	1.0	1.0

topography and preliminary three-dimensional computations can be found in [38].

7.1. Shock tube experiment. We simulate a shock tube experiment that has been studied numerically by Saito in [43] and was originally considered in [31]. It involves high-pressure and high-density pure air ($\gamma = 1.4$) in the driver section and air laden with solid particles at room conditions in the driven section. As in [43], an extremely small amount of dust is assumed here in the driver section for numerical reasons. Dimensionless initial data are reported in Table 2. Flow variables are normalized by using as reference density ρ_{ref}, reference pressure p_{ref}, and reference temperature T_{ref} the values for air at room conditions. Moreover, the reference velocity is defined as $u_{\text{ref}} = \sqrt{\frac{p_{\text{ref}}}{\rho_{\text{ref}}}}$.

For this experiment only we adopt the constitutive relations for drag and heat transfer that are used in [43]. These slightly differ from the relations reported in section 2.1 that we normally employ. Here, instead of (9),

$$C_{\text{d}} = 0.46 + 28\, Re^{-0.85}. \tag{45}$$

Moreover, in (10) κ_g is computed as $\kappa_g = \mu c_{pg} Pr^{-1}$, where the Prandtl number is assumed to be a constant, $Pr = 0.75$, while the gas dynamic viscosity μ varies with the temperature T (see [7]):

$$\mu = 1.71 \times 10^{-5} \left(\frac{T}{273}\right)^{0.77}. \tag{46}$$

The algorithm for the source terms described in section 5 is adapted to allow for this temperature dependent expression of μ by assuming this variable frozen at each time step in the ODE solution.

Finally, the dust particle diameter needed to compute the Reynolds number is $d = 10\,\mu$m, and the dust and gas heat coefficients at constant volume are assumed to be equal, with a dimensionless value given by $1/(\gamma - 1)$.

Following [43], we solve dimensionless equations, which are normalized by using $\ell = \frac{4\rho_d d}{3\rho_{\text{ref}}}$ as a characteristic length and $\tau = \frac{\ell}{u_{\text{ref}}}$ as a characteristic time. The computational domain is the interval $[0, 100]$, and the diaphragm is located at $x = 50$ units. We use 1000 cells (as in [43]) and CFL = 0.9.

Results of the computations at normalized time units = 5, 10, and 30 are displayed in Figures 2, 3, and 4, respectively, together with results corresponding to the case of pure gas in both the driver and driven sections for comparison.

The structure of the solution of this shock tube problem involves a left-going rarefaction wave, a contact discontinuity, and a right-going shock wave. In particular,

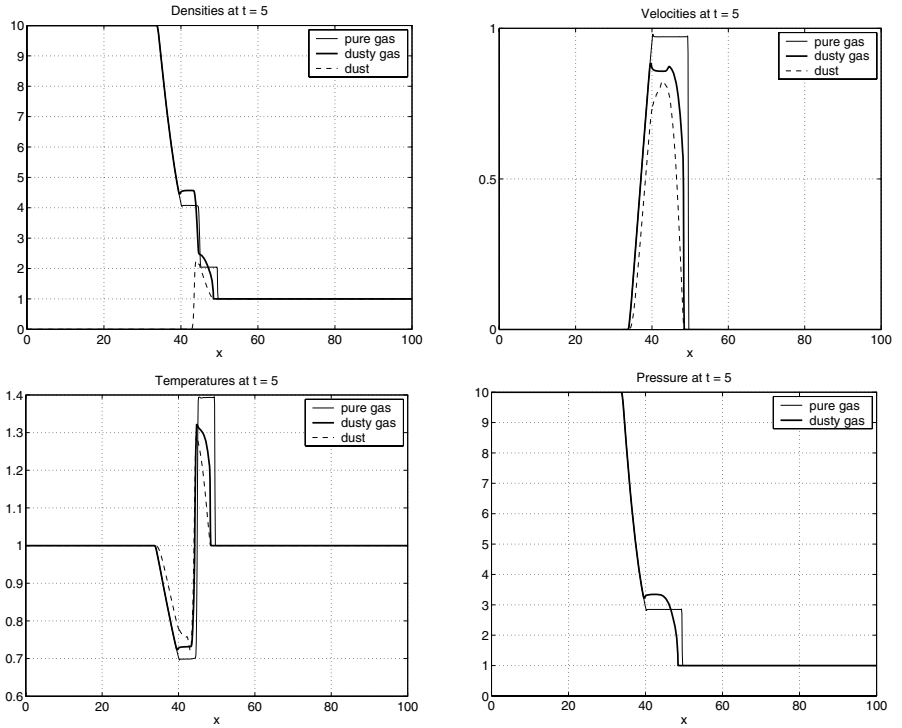


FIG. 2. *Saito's experiment. Densities, velocities, temperatures, and pressure at* $t = 5$.

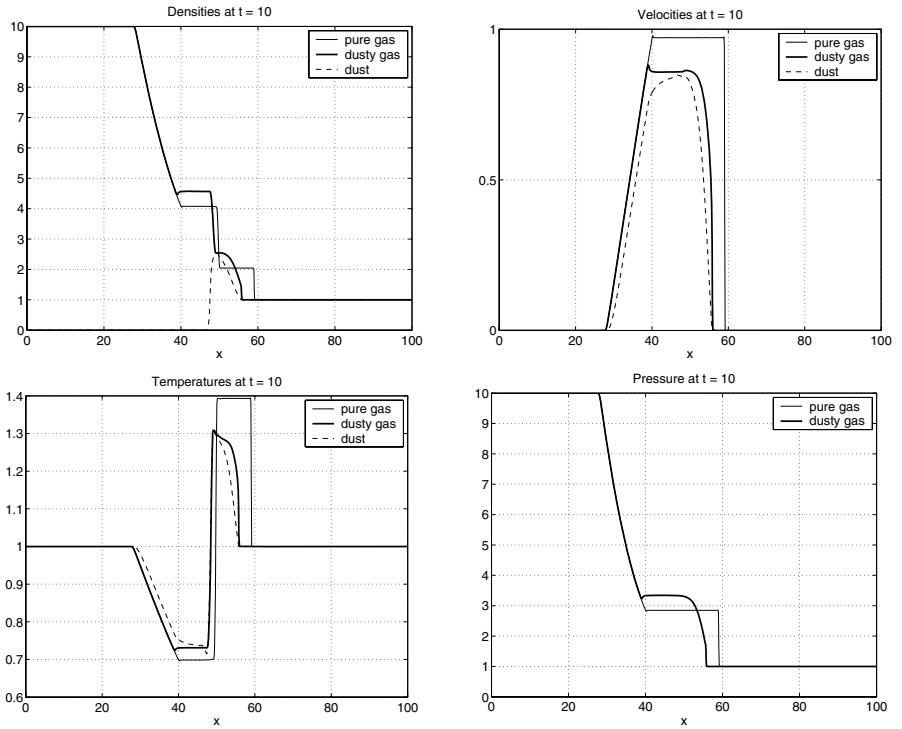


FIG. 3. *Saito's experiment. Densities, velocities, temperatures, and pressure at* $t = 10$.

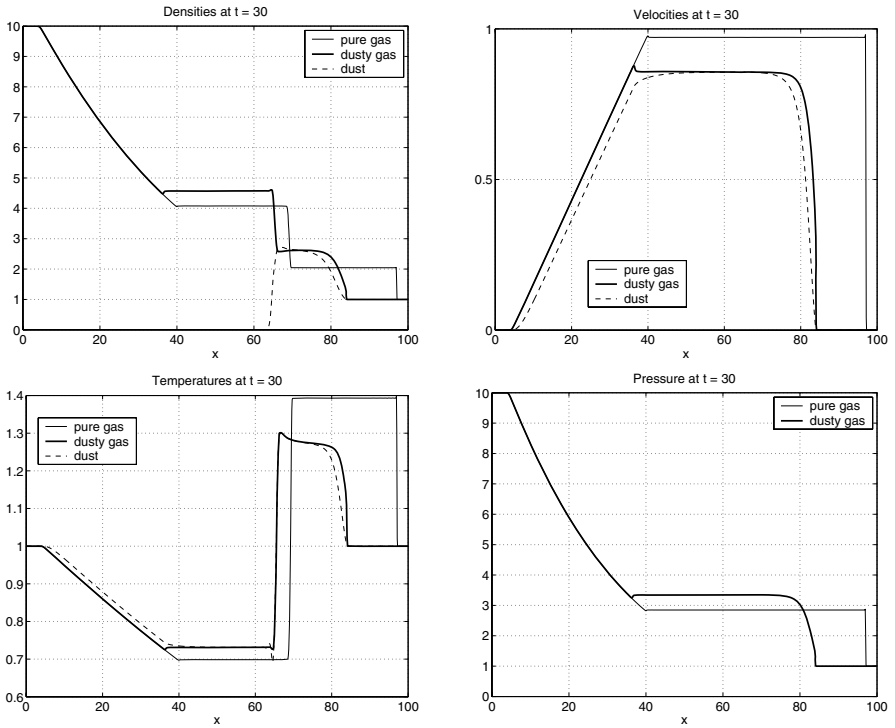


FIG. 4. *Saito's experiment. Densities, velocities, temperatures, and pressure at* $t = 30$.

the presence of the particulate phase leads to the development of a partially dispersed shock wave, characterized by a frozen shock in the gas followed by a relaxation region. This is formed since dust particles cannot follow any abrupt change in the gas due to their large inertia. With respect to the pure gas case, as described in [43], the dusty gas solution exhibits a deceleration of the shock front and a consequent higher compression behind it.

As time progresses, the structure of the solution tends to a stationary configuration characterized by an equilibrium region around the contact surface, and relaxation regions of constant width behind the rarefaction and shock wave. This stationary state can be observed in Figure 4, where in particular we can notice the equilibrium zones from the velocity and temperature profiles.

The results presented here agree well with those reported by Saito in [43], which are validated by the author by comparison with pseudostationary solutions obtained through numerical integration of the steady conservation equations. The numerical method of Saito is based on an operator splitting technique that uses the Harten [18] and Yee [48] scheme for the gas phase and a semianalytical approach for the dust phase. An inaccuracy in our results, not observed in [43], is the appearance of oscillations in the computed dust temperature in the neighborhood of the contact surface. These arise only when second-order corrections are performed, and they decrease with mesh refinement. We remark that oscillations are generated in the process of deriving the dust temperature from the internal energy per unit volume and dividing this by the computed dust density. No oscillations appear in the dust energy (or in other conserved variables), as shown in Figure 5. Note also that this

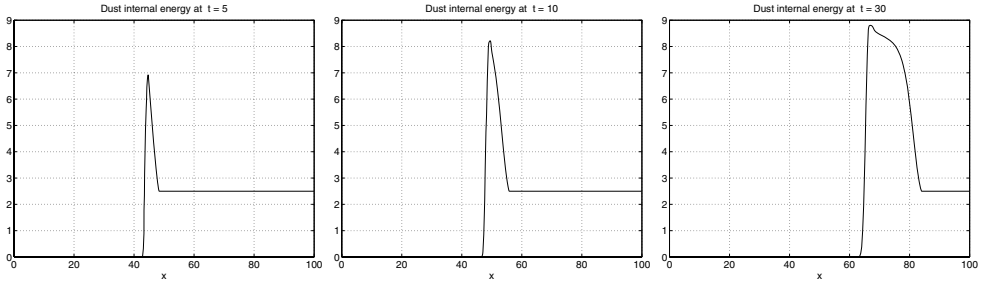


FIG. 5. *Saito's experiment. Dust internal energy per unit volume at* $t = 5$, 10, 30.

TABLE 3
Physical parameters for the two phases.

Quantity	Value	Unit
R	287.0	J/(kg K)
γ	1.4	
c_{vd}	$1.3 \cdot 10^3$	J/(kg K)
μ	$\mu = 10^{-5}$	Pa s
κ_g	0.05	W/(m K)

type of inaccuracy does not affect the computation process in our algorithm described in section 5. In fact, in the updating of the energies through (38) we actually do not need the dust temperature itself, but the product βT_d, as we can see by rewriting the term $\frac{T_g^0 - T_d^0}{\xi_{\mathcal{Q}}^0}$ as $\frac{\rho c_{vg} c_{vd}}{\rho c_{vg} + \beta c_{vd}} (\beta T_g^0 - \beta T_d^0)$. This is also consistent with the form of the heat transfer source term, which can be rewritten as $\mathcal{Q}(T_g - T_d) = \mathcal{Q}'(\beta T_g - \beta T_d)$, where $\mathcal{Q}' = \frac{Nu\, 6\kappa_g}{\rho_d d^2}$. The fact that in the results of Saito this difficulty does not arise can be explained by considering that his algorithm for the dust phase uses the dust temperature as a primary variable, whereas in our case the dust temperature is derived from the conserved variables.

7.2. Explosive volcanic eruptions. We employ the dusty gas flow model presented above to simulate some of the processes that characterize explosive volcanic events. Indeed, as mentioned in the introduction, the original motivation for the development of the two-phase flow model (27) was the numerical study of volcanic phenomena. The eruption mixture in this framework is described as a two-phase flow made of gas and solid particles of a single size. In particular, the gas phase is assumed to be dry air (no water vapor content). Some of the physical parameters adopted for the two phases are reported in Table 3.

We report results of two problems from a set of simulations that we have performed with data taken from the work of Neri and Dobran [32]. The aim was to test the performance of our model in describing the dynamics of pyroclastic dispersion in the atmosphere and pyroclastic flows on the ground, and the sensitivity to the variation of some eruption parameters that influence such processes.

The physical model assumed in [32] is more complex than ours and takes into account several effects that we neglect. Among these are viscosity of the gas, turbulence, water vapor content, exchange of momentum and energy between the two phases related to the gas pressure gradient, and the dense regime for the particulate phase. Despite the simplifications of our model, we observe that we are able to capture some of the relevant features of the various eruption styles that have been described in [32], at least in the first stages of the eruption that we study here.

TABLE 4
Vent conditions for the two numerical experiments.

Simulation	D_v [m]	v_v [m/s]	$p_{g,v}$ [MPa]	T_v [K]	$\vartheta_{d,v}$	d [μm]	ρ_d [kg/m^3]
A2	100	80	0.1	1200	0.01	10	2300
A6	100	200	0.1	1200	0.01	10	2300

Additional results on the simulation of volcanic processes, including modeling of overpressured jets on different crater morphology, can be found in [38].

Initial and boundary conditions. We consider the injection of a hot supersonic particle-laden gas from a volcanic vent into a cooler atmosphere (e.g., [47, 10, 32, 33, 35, 37, 36]). Initially, a standard atmosphere vertically stratified in pressure and temperature is set all over the domain. At the vent, the gas pressure, the velocities and temperatures of the two phases, and the dust volumetric fraction are assumed to be fixed and constant.

The ground boundary is modeled as a free-slip reflector. For two-dimensional experiments, an axisymmetric configuration of the flow is used, and system (41) is rewritten in cylindrical coordinates. We obtain a new set of equations with the same form of (41) on the left-hand side, but with an additional geometric source term on the right-hand side. This additional source term is treated numerically with an operator splitting technique and by employing a Runge–Kutta solver for the corresponding system of ODEs to be solved in time. In this two-dimensional axisymmetric configuration, half of the volcanic vent, of diameter D_v, is located in the lower left-hand corner of the computational domain and the symmetry axis is modeled as a free-slip reflector, while the upper and right-hand edges of the domain are free flow boundaries and all the variables gradients are set to zero.

Vent conditions. The gas and dust phase are assumed to be in thermal and mechanical equilibrium at the vent; that is, they have the same temperature T_v and the same exit velocity v_v. Data for the two numerical tests are summarized in Table 4 (subscript v refers to the vent), together with the physical properties assumed for the dust. The nomenclature of the two simulations is borrowed from [32]. The only difference with the set of data of this work is that we do not consider the water vapor content.

Note that the data sets of these two simulations differ only in the values of the vent velocities v_v, while the other variables describing the vent conditions are the same. The results for these data presented in [32] show that the eruption column evolves with a different style depending on the exit velocity: It is collapsing for $v_v = 80$ m/s, and transitional/Plinian for $v_v = 200$ m/s. As discussed below, the same behavior is described by our simulations. In these two experiments, omitting the water vapor content does not seem to influence the column style, at least in the first minutes of the process. However, some difference can be seen; see [38] for related discussion.

Note finally that for this set of experiments the vent pressure is balanced with the atmospheric pressure at the vent exit. Vent conditions are supersonic relative to the eruptive mixture, as can be verified by using the expression of the mixture sound speed in (23), and they are representative of jet conditions after decompression in the volcanic crater [32].

7.2.1. Simulation A2: Stationary collapsing column. Figure 6 shows the evolution of the eruptive mixture as computed for simulation A2, characterized by

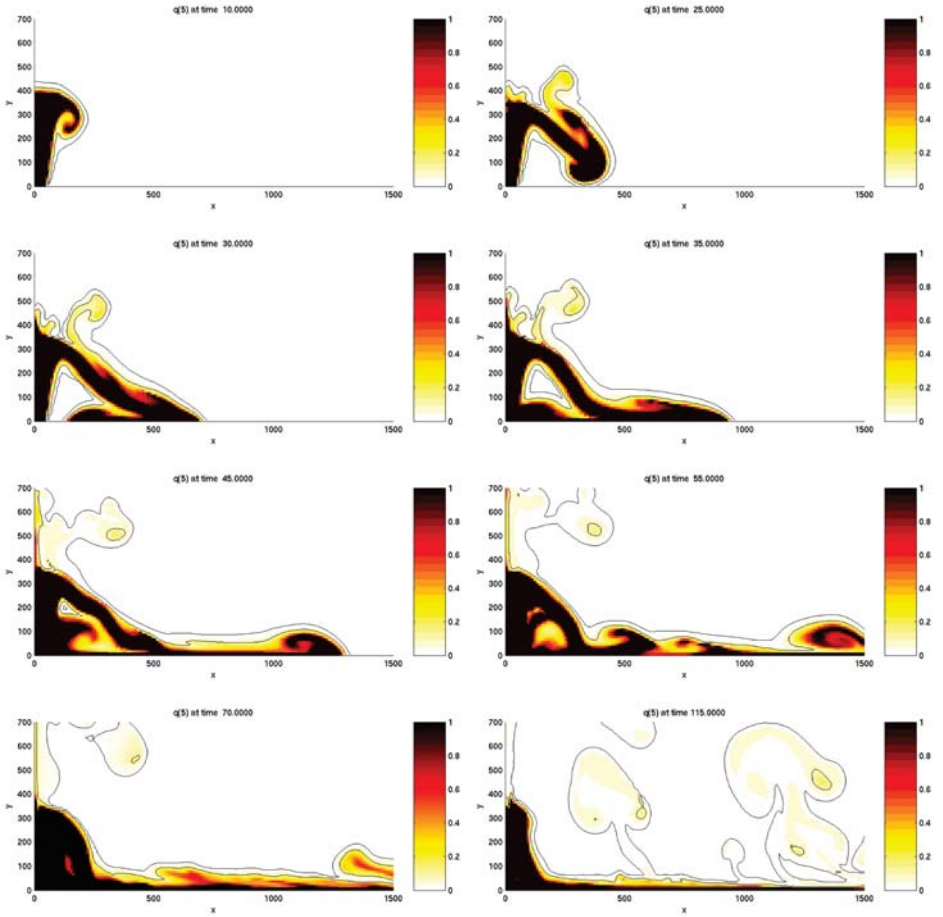


FIG. 6. *Simulation* A2. *Collapsing column. Dust density at time* $t = 10, 25, 30, 35, 45, 55, 70, 115$ *s. Computational domain:* $3000\ m \times 3000\ m$. *Cell size* $= 10\ m$ $(300 \times 300$ *cells*). *Contour values* $= 10^{[-4:1:0]}$ $[\mathrm{kg/m^3}]$.

$D_v = 100\,\mathrm{m}$ and $v_v = 80\,\mathrm{m/s}$. In agreement with [32], we can recognize the typical features of a collapsing volcanic column, that is, fountain building above the volcanic vent, radially spreading pyroclastic flow, material recycling from the collapsed column back into the fountain, and rising of ash plumes from the pyroclastic flow. At about 10 s and 400 m above the vent the two-phase flow jet loses its vertical thrust and begins to form a collapsing column. After the collapsed column has hit the ground, material starts being recycled into the fountain. As the time evolves, the impact distance from the vent decreases and the column tends to stabilize with a steady narrow shape (as observed in [32]).

In Figure 7 we display results computed on two different grid resolutions (cell size = 10 m on the left and cell size = 5 m on the right). We can see that a smaller cell size produces a longer runout of the pyroclastic flow and a smaller thickness, but there is no notable difference in the fountain height and in the dynamics of the collapse. The significant difference in the location of the flow head on the ground can be in part related to the type of free-slip boundary condition used, though the same effect of the

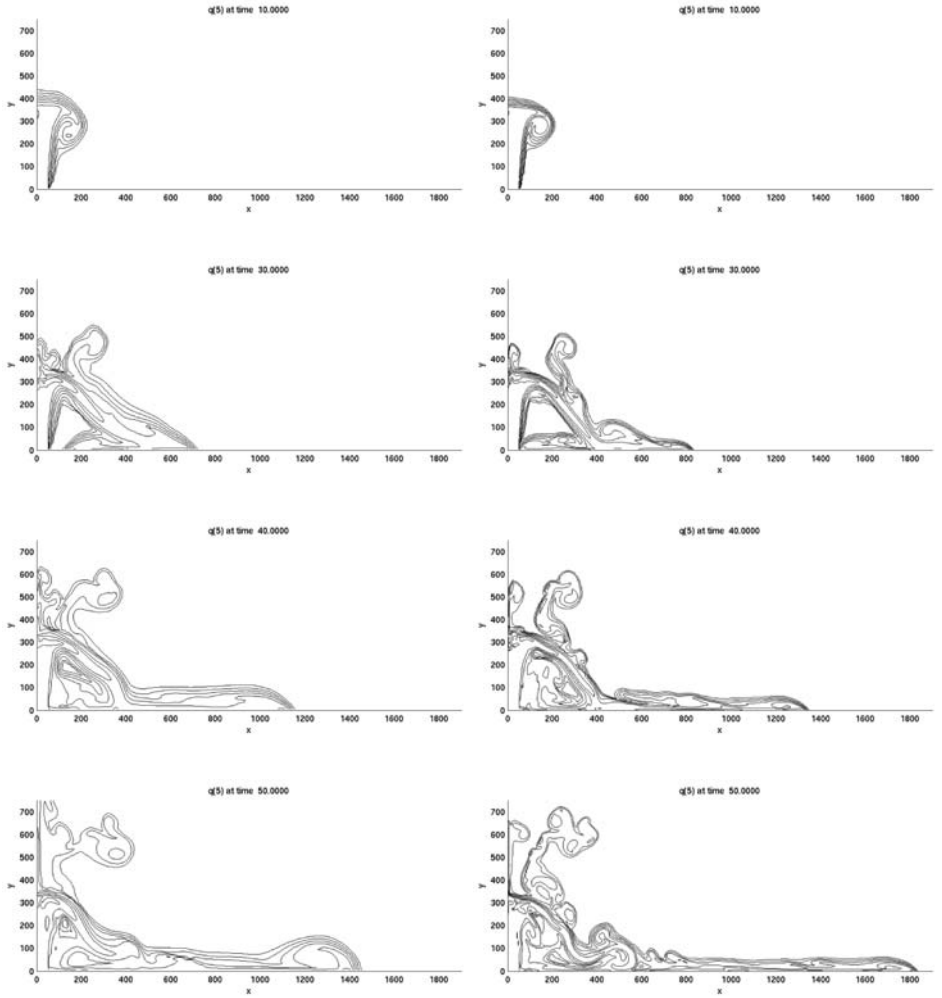


FIG. 7. *Simulation* A2. *Comparison between two different grid resolutions. Dust density at time* $t = 10, 30, 40, 50$ *s. Left:* $\Delta x = 10\,m$*; right:* $\Delta x = 5\,m$*. Contour values* $= 10^{[-5:.5:2]}$ [kg/m^3].

grid resolution has also been observed in [10], where a no-slip boundary condition is employed.

In Figure 8 we highlight the features of the recirculation region displaying a contour plot of the dust density together with the gas velocity vector field, as computed on the finest grid.

Relating our results with those in [32], we see good agreement between plots of physical quantities. In particular, the column height and the column impact distance are comparable. There is a difference in the length of the runout of the pyroclastic flow, which is larger in our computations. This is not surprising, since we use a free-slip boundary condition on the ground, whereas in [32] a no-slip condition is used (consistent with the fact that their model contains viscous terms).

7.2.2. Simulation A6: Transitional/Plinian column. Figure 9 shows the results for simulation A6, which has the same vent diameter $D_v = 100\,\text{m}$ as A2, but

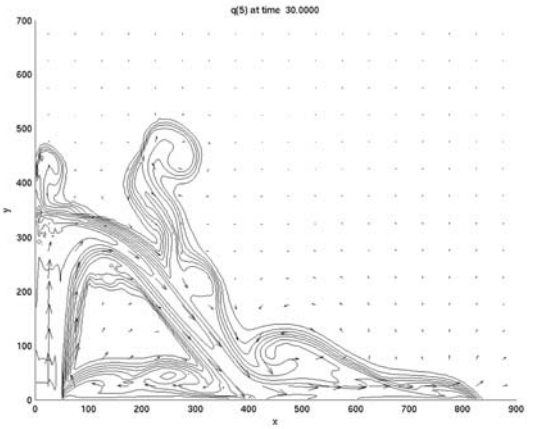


FIG. 8. *Simulation* A2. *Dust density contours and gas velocity vector field at* $t = 30$ *s. Contour values* $= 10^{[-5:.4:2]}$ [kg/m^3]. *Computational domain:* 2000 *m* $\times$ 1000 *m. Cell size* = 5 *m* (400 $\times$ 200 *cells*).

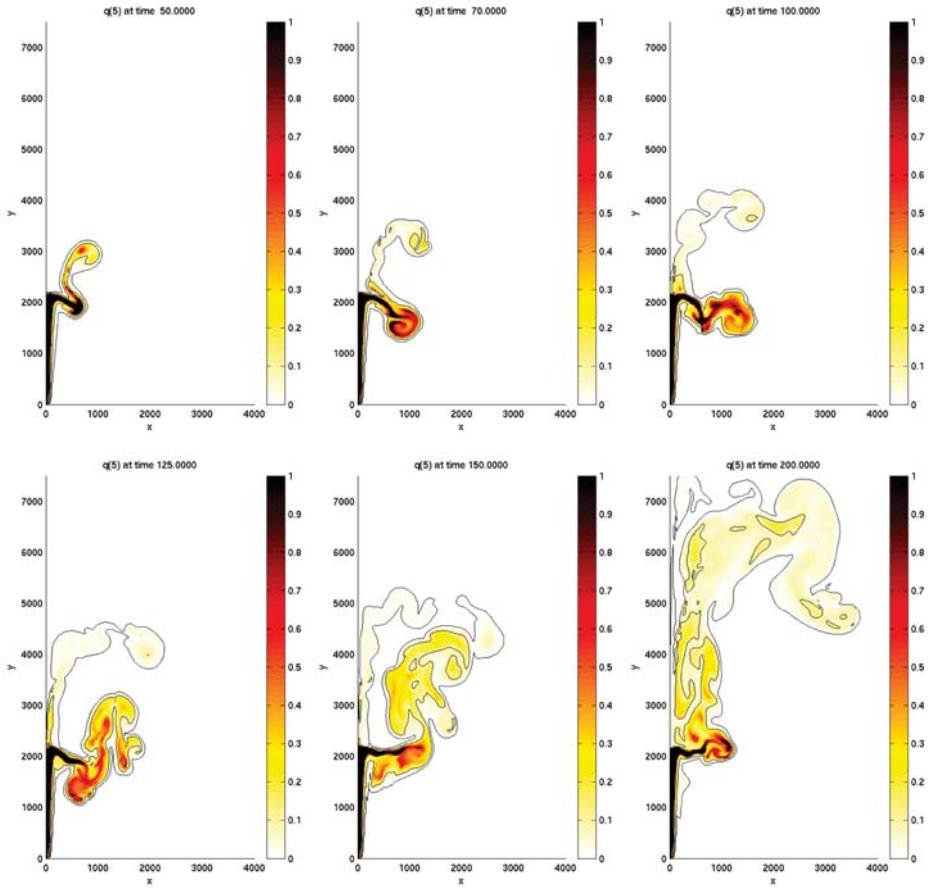


FIG. 9. *Simulation* A6. *Dust density at time* $t = 50, 70, 100, 125, 150, 170$ *s. Contour values* $= 10^{[-5:1:0]}$ [kg/m^3]. *Computational domain:* 5000 *m* $\times$ 9000 *m. Cell size* = 25 *m* (200 $\times$ 360 *cells*).

a greater exit velocity, $v_v = 200\,\mathrm{m/s}$. The increase in the jet velocity (keeping D_v fixed) generates a much more buoyant column, which in this case appears of transitional/Plinian type, consistent with the observations in [32]. This column reaches a height of about 2200 m and then begins to form a radially suspended flow with some material rising buoyantly on the top, and other material attempting instead to collapse. The eruptive mixture later develops as a buoyant column. Our plots show qualitative agreement with those displayed in [32], and in particular the column height is about the same.

8. Conclusions and extensions. We have considered a model for dusty gas flow that consists of the compressible Euler equations for the gas coupled to a similar (but pressureless) system of equations for the mass, momentum, and energy of the dust. These sets of equations are coupled via drag terms (depending on the velocity difference between dust and gas) and heat transfer (depending on the temperature difference). A high-resolution wave-propagation algorithm has been used to solve the equations numerically. This finite volume method is based on solving Riemann problems at cell interfaces. The source term modeling the effect of gravity on the gas is incorporated into the Riemann solver in order to minimize numerical artifacts that could arise from splitting this source term from the pressure gradient. Other source terms (drag, heat transfer, and the gravitational force on the dust) are handled with a fractional step approach.

The one-dimensional algorithm has been shown to give agreement with results presented by Saito [43]. The two-dimensional algorithm has been applied to model explosive volcanic eruptions in which an axisymmetric jet of hot dusty gas is injected into the atmosphere. Two test cases have been presented, one in which the jet collapses into a flow along the surface, and a second, at higher velocity, where the jet rises as a Plinian column. The expected behavior is observed in each case, based on results of other researchers.

Additional test cases are presented in [38], where it is also demonstrated that adaptive mesh refinement can be successfully applied to this problem using the AMRCLAW software in CLAWPACK [26]. Some preliminary computations in three space dimensions are also presented in [38] and compared to the two-dimensional axisymmetric results. Three-dimensional modeling over realistic topography such as Mount St. Helens is currently being investigated with an extension of this code.

Because the speed of sound in a dusty gas is much lower than in a pure gas (see section 2.2), volcanic jets can easily be supersonic and hence the jet can exhibit complex internal shock structures [22]. Some numerical study of this structure is presented in [38] and compared with some examples considered in the work of Esposti Ongaro and Neri [12]. In a joint research project with these researchers, we are continuing our study of volcanic jets using the methodology described in this paper. Further results will be presented elsewhere [13].

REFERENCES

[1] N. Andrianov and G. Warnecke, *The Riemann problem for the Baer-Nunziato two-phase flow model*, J. Comput. Phys., 195 (2004), pp. 434–464.

[2] D. S. Bale, R. J. LeVeque, S. Mitran, and J. A. Rossmanith, *A wave propagation method for conservation laws and balance laws with spatially varying flux functions*, SIAM J. Sci. Comput., 24 (2002), pp. 955–978.

[3] N. Botta, R. Klein, S. Langenberg, and S. Lützenkirchen, *Well balanced finite volume methods for nearly hydrostatic flows*, J. Comput. Phys., 196 (2004), pp. 539–565.

[4] F. Bouchut and F. James, *Duality solutions for pressureless gases, monotone scalar conservation laws, and uniqueness*, Comm. Math. Phys., 24 (1999), pp. 2173–2189.
[5] D. A. Calhoun, P. Colella, and R. J. LeVeque, chombo-claw software, available online from http://www.amath.washington.edu/~calhoun/demos/ChomboClaw.
[6] P. Cargo and A.-Y. LeRoux, *Un schéma équilibre adapté au modèle d'atmosphère avec termes de gravité*, C. R. Acad. Sci. Paris Sér. I Math., 318 (1994), pp. 73–76.
[7] S. Chapman and T. G. Cowling, *The Mathematical Theory of Nonuniform Gases*, Cambridge University Press, Cambridge, UK, 1970.
[8] G. Dal Maso, P. G. LeFloch, and F. Murat, *Definition and weak stability of nonconservative products*, J. Math. Pures Appl. (9), 74 (1995), pp. 483–548.
[9] S. Dartevelle, W. I. Rose, J. Stix, K. Kelfoun, and J. W. Vallance, *Numerical modeling of geophysical granular flows: 2. Computer simulations of Plinian clouds and pyroclastic flows and surges*, Geochem. Geophys. Geosyst., 5 (2004); available online from http://www.agu.org/pubs/crossref/2004.../2003GC000637.shtml.
[10] F. Dobran, A. Neri, and G. Macedonio, *Numerical simulation of collapsing volcanic columns*, J. Geophys. Res., 98 (1993), pp. 4231–4259.
[11] D. A. Drew and S. L. Passman, *Theory of Multicomponent Fluids*, Springer-Verlag, New York, 1999.
[12] T. Esposti Ongaro and A. Neri, *Flow Patterns of Overpressured Volcanic Jets*, European Geophysical Society, XXIV General Assembly, The Hague, The Netherlands, 1999.
[13] T. Esposti Ongaro, M. Pelanti, A. Neri, and R. J. LeVeque, *Structure and dynamics of underexpanded volcanic jets: A comparative study using two numerical multiphase flow models*, in preparation.
[14] L. Gosse, *A well-balanced flux-vector splitting scheme designed for hyperbolic systems of conservation laws with source terms*, Comput. Math. Appl., 39 (2000), pp. 135–159.
[15] L. Gosse, *A well-balanced scheme using non-conservative products designed for hyperbolic systems of conservation laws with source terms*, Math. Models Methods Appl. Sci., 11 (2001), pp. 339–365.
[16] J. M. Greenberg and A. Y. Leroux, *A well-balanced scheme for the numerical processing of source terms in hyperbolic equations*, SIAM J. Numer. Anal., 33 (1996), pp. 1–16.
[17] F. H. Harlow and A. A. Amsden, *Numerical calculation of multiphase fluid flow*, J. Comput. Phys., 17 (1975), pp. 19–52.
[18] A. Harten, *High resolution schemes for hyperbolic conservation laws*, J. Comput. Phys., 49 (1983), pp. 357–393.
[19] M. Ishii, *Thermo-fluid Dynamic Theory of Two-phase Flow*, Eyrolles, Paris, 1975.
[20] S. Jin, *A steady-state capturing method for hyperbolic systems with geometrical source terms*, M2AN Math. Model Numer. Anal., 35 (2001), pp. 631–645.
[21] B. Khouider and A. J. Majda, *A high resolution balanced scheme for an idealized tropical climate model*, Theor. Comp. Fluid Dyn., to appear.
[22] S. W. Kieffer and B. Sturtevant, *Laboratory studies of volcanic jets*, J. Geophys. Res., 89 (1984), pp. 8253–8268.
[23] J. G. Knudsen and D. L. Katz, *Fluid Mechanics and Heat Transfer*, McGraw-Hill, New York, 1958.
[24] J. O. Langseth and R. J. LeVeque, *A wave-propagation method for three-dimensional hyperbolic conservation laws*, J. Comput. Phys., 165 (2000), pp. 126–166.
[25] P. G. LeFloch and A. E. Tzavaras, *Representation of weak limits and definition of nonconservative products*, SIAM J. Math. Anal., 30 (1999), pp. 1309–1342.
[26] R. J. LeVeque, clawpack software, available online from http://www.amath.washington.edu/~claw.
[27] R. J. LeVeque, *Wave propagation algorithms for multi-dimensional hyperbolic systems*, J. Comput. Phys., 131 (1997), pp. 327–353.
[28] R. J. LeVeque, *Balancing source terms and flux gradients in high-resolution Godunov methods: The quasi-steady wave-propagation algorithm*, J. Comput. Phys., 146 (1998), pp. 346–365.
[29] R. J. LeVeque, *Finite Volume Methods for Hyperbolic Problems*, Cambridge University Press, Cambridge, UK, 2002.
[30] R. J. LeVeque, *The dynamics of pressureless dust clouds and delta waves*, J. Hyperbolic Differ. Equ., 1 (2004), pp. 315–327.
[31] H. Miura and I. I. Glass, *On a dusty-gas shock tube*, Proc. R. Soc. Lond. Ser. A, 382 (1982), pp. 373–388.
[32] A. Neri and F. Dobran, *Influence of eruption parameters on the thermofluid dynamics of collapsing volcanic columns*, J. Geophys. Res., 99 (1994), pp. 11833–11857.

[33] A. Neri and G. Macedonio, *Numerical simulation of collapsing volcanic columns with particles of two sizes*, J. Geophys. Res., 101 (1996), pp. 8153–8174.
[34] A. Neri, G. Macedonio, D. Gidaspow, and T. Esposti Ongaro, *Multiparticle Simulation of Collapsing Volcanic Columns and Pyroclastic Flows*, VSG Report 2001-2, Volcano Simulation Group, Istituto Nazionale di Geofisica e Vulcanologia, Pisa, Italy, 2001.
[35] A. Neri, A. Di Muro, and M. Rosi, *Mass partition during collapsing and transitional columns by using numerical simulations*, J. Volcanol. Geotherm. Res., 115 (2002), pp. 1–18.
[36] A. Neri, T. Esposti Ongaro, G. Macedonio, and D. Gidaspow, *Multiparticle simulation of collapsing volcanic columns and pyroclastic flows*, J. Geophys. Res., 108 (2003), pp. 1–22.
[37] A. Neri, P. Papale, and G. Macedonio, *The role of magma composition and water content in explosive eruptions: 2. Pyroclastic dispersion dynamics*, J. Volcanol. Geotherm. Res., 87 (1998), pp. 95–115.
[38] M. Pelanti, *Wave Propagation Algorithms for Multicomponent Compressible Flows with Applications to Volcanic Jets*, Ph.D. thesis, University of Washington, Seattle, 2005.
[39] P. L. Roe, *Approximate Riemann solvers, parameter vectors, and difference schemes*, J. Comput. Phys., 43 (1981), pp. 357–372.
[40] L. Sainsaulieu, *An Euler system modeling vaporizing sprays*, in Dynamics of Heterogeneous Combustion and Reacting Systems (Progress Series in Aeronautics and Astronautics), Vol. 152, A. L. Kuhl, J.-C. Leger, A. A. Borisov, and W. A. Sirignano, eds., American Institute of Aeronautics and Astronautics, Washington, D.C., 1993, pp. 280–305.
[41] L. Sainsaulieu, *Finite volume approximation of two-phase fluid flows based on an approximate Roe-type Riemann solver*, J. Comput. Phys., 121 (1995), pp. 1–28.
[42] L. Sainsaulieu, *Traveling waves solution of convection-diffusion systems whose convection terms are weakly nonconservative: Application to the modeling of two-phase fluid flows*, SIAM J. Appl. Math., 55 (1995), pp. 1552–1576.
[43] T. Saito, *Numerical analysis of dusty-gas flows*, J. Comput. Phys., 176 (2002), pp. 129–144.
[44] G. A. Valentine and K. H. Wohletz, *Numerical models of Plinian eruption columns and pyroclastic flows*, J. Geophys. Res., 94 (1989), pp. 1867–1887.
[45] G. B. Wallis, *One-Dimensional Two-Phase Flow*, McGraw-Hill, New York, 1969.
[46] K. H. Wohletz, T. R. McGetchin, M. T. Sanford II, and E. M. Jones, *Hydrodynamic forming of caldera-forming eruptions: Numerical models*, J. Geophys. Res., 89 (1984), pp. 8269–8285.
[47] K. H. Wohletz and G. A. Valentine, *Computer simulations of explosive volcanic eruption*, in Magma Transport and Storage, M. P. Ryan, ed., John Wiley, New York, 1990, pp. 113–135.
[48] H. C. Yee, *Upwind and Symmetric Shock-Capturing Schemes*, NASA Ames Technical Memorandum 89464, 1987.

SIAM J. SCI. COMPUT.
Vol. 28, No. 4, pp. 1361–1378
© 2006 Society for Industrial and Applied Mathematics

DISTRIBUTED IMMERSED BOUNDARY SIMULATION IN TITANIUM*

E. GIVELBERG† AND K. YELICK‡

Abstract. The immersed boundary method is a general numerical method for modeling elastic boundaries immersed within a viscous, incompressible fluid. It has been applied to several biological and engineering systems, including large-scale models of the heart and cochlea. These simulations have the potential to improve our basic understanding of the biological systems they model and aid in the development of surgical treatments and prosthetic devices. Despite the popularity of the immersed boundary method and the desire to scale the problems to accurately capture the details of the physical systems, parallelization for large-scale distributed memory machines has proved challenging. The primary difficulty is in achieving a load-balanced computation, while maintaining low communication costs when modeling the interactions between the fluid and the moving immersed boundary. In this paper we describe a parallelized algorithm for the immersed boundary method that is designed for scalability on distributed memory multiprocessors and clusters of SMPs. It is implemented using the Titanium language, a Java-based language designed for high performance scientific computing. Our software package, called `IB`, takes advantage of the object-oriented features of Titanium to provide a framework for simulating immersed boundaries that separates the generic immersed boundary method code from the specific application features that define the immersed boundary structure and the forces that arise from those structures. Our results demonstrate the scalability of our design and the feasibility of large-scale immersed boundary computations with the `IB` package.

Key words. fluid-structure interactions, immersed boundary method, distributed algorithm, large-scale computation

AMS subject classifications. 65C20, 68W15, 65Y05, 76D05, 35Q30, 73K70

DOI. 10.1137/040618734

1. Introduction.

The immersed boundary method is a general numerical method for computational modeling of systems involving fluid-structure interactions. Complex systems where elastic (and possibly active) tissue is immersed in a viscous, incompressible fluid arise naturally in biology and engineering. The immersed boundary method was developed by Peskin and McQueen to study the patterns of the blood flow in the heart [17, 15]. It has subsequently been applied to a variety of problems, such as platelet aggregation during blood clotting [6], the deformation of red blood cells in a shear flow [5], the flow in collapsible thin-walled vessels [22], the swimming of eels, sperm, and bacteria [7, 4], the flow past a cylinder [14], two-dimensional [3] and three-dimensional models of the cochlea [10], valveless pumping [13], and flexible filament flapping in a flowing soap film [26]. For a recent review of the research in immersed boundary computations and further applications, see [18].

Realistic immersed boundary simulations of complex systems, such as the heart and the cochlea, require very large computing resources: The heart model experiments were carried out on the Cray T90 [19, 16], and the cochlea was constructed on the HP Superdome at Caltech [11]. Large computational grids are necessary to reduce the numerical error and to incorporate the finer details of the simulated system into

*Received by the editors November 10, 2004; accepted for publication (in revised form) February 16, 2006; published electronically August 7, 2006.
http://www.siam.org/journals/sisc/28-4/61873.html
†17 Shoham, Haifa, Israel (givelber@cims.nyu.edu).
‡Computer Science Division, University of California, Berkeley, CA 94720 (yelick@eecs.berkeley.edu).

the model. For example, higher resolution in the heart model reveals the patterns of the turbulent flow around the valves. Similarly, in the cochlea the microstructure of the organ of Corti is of crucial importance to the dynamics of the whole system. Consequently, numerical experiments with both systems often required days of dedicated computing. Both the Superdome and the Cray T90 are shared memory machines; thus the parallelization of the serial immersed boundary code was achieved mainly (but not exclusively) with the help of compiler directives. Yet, the use of finer grids in simulations of complex systems eventually leads to computations that exceed the capabilities of the available shared memory systems. Large-scale simulations involving hundreds, and perhaps thousands, of processors must be carried out on machines with distributed architecture, which require the development of algorithms such as the one presented in this paper.

The main challenge in designing distributed algorithms consists of the need to balance the cost of communicating information among the available processors with the need to efficiently utilize these processors. On most distributed architecture multiprocessor machines, communication between processors consists of an initial phase, required to establish the connection, followed by the actual transfer of the data. The cost of the initial phase, called latency, is typically quite high. For example, on one of the most commonly available machines, the IBM SP RS/6000, the measured latency is 7.6 microseconds, while the bandwidth is about 240 MB/sec (see [2]), making small message transfer prohibitively expensive. It is therefore necessary to impose the additional constraint of bulk communication on the design of distributed algorithms.

The immersed boundary method uses a mixed Lagrangian–Eulerian formulation and has the important property that virtually any complex model of the immersed boundary force generation can be easily incorporated. The design of the distributed algorithm should preserve this property. The fluid and the immersed material are naturally described by separate computational grids. The fluid is modeled by a three-dimensional rectangular grid, while the immersed material is typically modeled as a collection of elastic fibers (one-dimensional grids) or as an elastic shell (a two-dimensional grid). Simulation proceeds in a series of time steps, where during each time step the elastic forces are computed on the material grids and then spread to the fluid grid. The fluid equations are solved yielding a new fluid velocity, which is then interpolated to the material grids and is finally used to update their position relative to the fluid.

Distributed memory implementations of the immersed boundary method have proved quite challenging. The interaction between the fluid and immersed boundaries is the primary source of difficulties in the design of a distributed algorithm. Typically, the immersed boundary structures are not evenly distributed throughout the fluid domain and, furthermore, their position may change significantly with time. The resulting system may therefore exhibit a significant amount of irregular communication. The first step toward distributed immersed boundary computations was taken by Sabbagh, who constructed a distributed Navier–Stokes solver in his doctoral dissertation [23]. Previous attempts to design a distributed algorithm for the whole immersed boundary method included a Split-C version on the Thinking Machine CM5 and an earlier Titanium version [24] on the Cray T3E, both machines having support for lightweight (i.e., low latency) communication. Despite the great need, so far no distributed memory implementation has been used in any immersed boundary model.

The heart and the cochlea are the two examples motivating our present work in developing the algorithm and the software package `IB` for immersed boundary com-

putations in Titanium. Titanium is an explicitly parallel dialect of Java developed at the University of California-Berkeley to support high-performance scientific computing on large-scale multiprocessors, including massively parallel supercomputers and distributed-memory clusters with one or more processors per node. Other language goals include safety, portability, and support for building complex data structures. Titanium is open-source software available from the UC Berkeley Computer Science Department [1].

The complexity of an immersed boundary computation is determined by the sizes of the fluid and the immersed boundary grids, and by the size of the time step. The heart model uses a 128^3-point fluid grid, with the heart muscle and the valves modeled by a collection of elastic fibers, totaling approximately 600,000 points. The cochlea model, on the other hand, uses a 256^3-point fluid grid, with the immersed material modeled as a set of elastic shells and bony walls, totaling approximately 750,000 points. Extensive numerical experiments with the cochlea have shown that the 256^3-point fluid grid is not sufficient for many numerical experiments (see [10]). The `IB` package has already been used to construct a 128^3-point heart model and a 512^3-point cochlea model, and a 256^3-point heart model is presently under construction. This work will be described in future publications.

The rest of the paper is organized as follows. In the next section we introduce the immersed boundary equations. These equations form the basis of the numerical method, which is described in section 3. The algorithm and the data structures utilized in our implementation are outlined in section 4. Sections 5 and 6 make the case for a wider adoption of the Titanium programming language for large-scale scientific computing: section 5 surveys the main features of the Titanium programming language, and section 6 briefly surveys some features of the `IB` software package. In the following section we demonstrate the feasibility of large-scale immersed boundary computations using our software. We conclude with a summary and a discussion of our plans for further research.

2. The immersed boundary equations. The immersed boundary method is based on a mixed Eulerian–Lagrangian formulation of the fluid-immersed material system. The fluid is described in the standard Cartesian coordinates on $\mathbb{R}^3$, while the immersed material is described in a different curvilinear coordinate system. Let ρ and μ denote the density and the viscosity of the fluid, and let $\mathbf{u}(\mathbf{x},t)$ and $p(\mathbf{x},t)$ denote its velocity and pressure, respectively. The Navier–Stokes equations of a viscous incompressible fluid are

$$\rho\left(\frac{\partial \mathbf{u}}{\partial t}+\mathbf{u}\cdot\nabla\mathbf{u}\right)=-\nabla p+\mu\nabla^2\mathbf{u}+\mathbf{F}, \tag{2.1}$$

$$\nabla\cdot\mathbf{u}=0, \tag{2.2}$$

where $\mathbf{F}$ denotes the density of the body force acting on the fluid. For example, if the immersed material is modeled as a thin shell, then $\mathbf{F}$ is a singular vector field, which is zero everywhere, except possibly on the surface representing the shell. While the immersed boundary equations are valid in any domain in $\mathbb{R}^3$, we will assume that the domain is rectangular and the boundary conditions are periodic. This enables us to use fast Fourier methods for the solution of the Navier–Stokes equations (2.1) and (2.2).

Let $\mathbf{X}(\mathbf{q},t)$ denote the position of the immersed material in $\mathbb{R}^3$. For a shell, $\mathbf{q}$ takes values in a domain $\Omega\subset\mathbb{R}^2$, and $\mathbf{X}(\mathbf{q},t)$ is a 1-parameter family of surfaces

indexed by t; i.e., $\mathbf{X}(\mathbf{q},t)$ is the middle surface of the shell at time t. Let $\mathbf{f}(\mathbf{q},t)$ denote the force density that the immersed material applies on the fluid. Then

$$\mathbf{F}(\mathbf{x},t) = \int \mathbf{f}(\mathbf{q},t)\delta(\mathbf{x}-\mathbf{X}(\mathbf{q},t))\,d\mathbf{q}, \tag{2.3}$$

where δ is the Dirac delta function on $\mathbb{R}^3$. This equation merely says that the fluid feels the force that the immersed material exerts on it, but it is important in the numerical method, where it is one of the equations determining fluid-material interaction. The other interaction equation is the no-slip condition for a viscous fluid:

$$\begin{aligned}\frac{\partial \mathbf{X}}{\partial t} &= \mathbf{u}(\mathbf{X}(\mathbf{q},t),t)\\ &= \int \mathbf{u}(\mathbf{x},t)\delta(\mathbf{x}-\mathbf{X}(\mathbf{q},t))\,d\mathbf{x}.\end{aligned} \tag{2.4}$$

The system has to be completed by specifying the force $\mathbf{f}(\mathbf{q},t)$ of the immersed material. In a complicated system such as the cochlea, the immersed material consists of many different components: membranes, bony walls, an elastic shell representing the basilar membrane, and various cells of the organ of Corti, including outer hair cells, which may actively generate forces. For each such component it is necessary to specify its own computation grid and an algorithm to compute its force $\mathbf{f}$. It is in the specification of these forces that models for various system components integrate into the macromechanical model.

3. The first-order immersed boundary numerical method. We describe here a first-order immersed boundary numerical scheme, which is the easiest to implement. The main algorithmic ideas developed in section 4 apply also to other versions of the immersed boundary method, such as the formally second-order immersed boundary method [14]. The fluid equations are discretized on a periodic rectangular lattice of mesh width h. The immersed material is described by a collection of one-, two- or three-dimensional computational grids, whose mesh width is typically approximately $h/2$. Some of the computational grids used in the cochlea model are shown in Figure 3.1.

The computation proceeds in time steps of duration Δt. It will be convenient to denote the time step by the superscript. For example, $\mathbf{u}^n(\mathbf{x}) = \mathbf{u}(\mathbf{x}, n\Delta t)$. At the beginning of the nth time step, $\mathbf{X}^n$ and $\mathbf{u}^n$ are known. Each time step consists of four parts, as shown in Procedure 1 (see [18] for more details).

PROCEDURE 1. The time step of the first-order immersed boundary method.

1: Compute $\mathbf{f}^n$, the force that the immersed boundary applies to the fluid.
2: Compute $\mathbf{F}^n$, the external force on the fluid.
3: Compute $\mathbf{u}^{n+1}$, the new fluid velocity.
4: Compute $\mathbf{X}^{n+1}$, the new position of the immersed boundary.

In part 1 the force $\mathbf{f}^n$ that the immersed boundary applies to the fluid is computed. For simple materials, such as fibers, this is a straightforward computation (see [20]). More complex material models may be considered. For a detailed description of an elastic shell immersed boundary force computation, see [9]. In part 2 the interpolation equation (2.3) is used to compute the external force on the fluid $\mathbf{F}^n$ from the force $\mathbf{f}^n$ applied by the immersed boundary. The force $\mathbf{F}^n$ is used in the Navier–Stokes solver

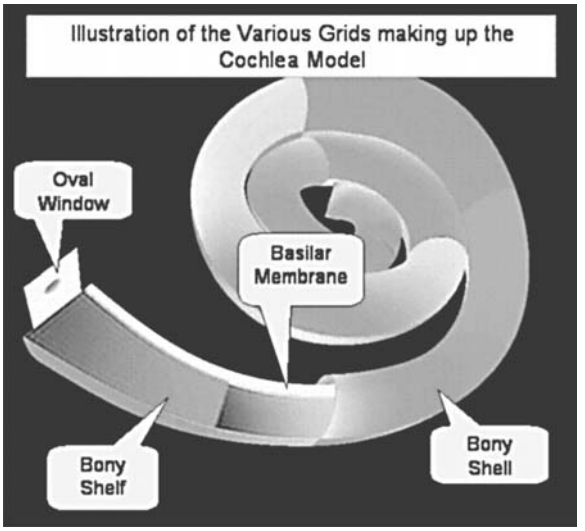


FIG. 3.1. *The immersed material grids of the cochlea model.*

in part 3 to compute the new fluid velocity $\mathbf{u}^{n+1}$. In part 4 this velocity is used in the interpolation equations (2.4) to compute the new position $\mathbf{X}^{n+1}$ of the immersed material.

We shall now describe the computations in steps 2–4 in detail, beginning with the Navier–Stokes equations. We will make use of the following difference operators which act on functions defined on the fluid lattice:

$$D_i^+\phi(\mathbf{x}) = \frac{\phi(\mathbf{x}+h\mathbf{e}_i)-\phi(\mathbf{x})}{h}, \tag{3.1}$$

$$D_i^-\phi(\mathbf{x}) = \frac{\phi(\mathbf{x})-\phi(\mathbf{x}-h\mathbf{e}_i)}{h}, \tag{3.2}$$

$$D_i^0\phi(\mathbf{x}) = \frac{\phi(\mathbf{x}+h\mathbf{e}_i)-\phi(\mathbf{x}-h\mathbf{e}_i)}{2h}, \tag{3.3}$$

$$\mathbf{D^0} = (D_1^0, D_2^0, D_3^0), \tag{3.4}$$

where $i = 1, 2, 3$ and $\mathbf{e}_1$, $\mathbf{e}_2$, $\mathbf{e}_3$ form an orthonormal basis of $\mathbb{R}^3$.

In step 3 we use the already known $\mathbf{u}^n$ and $\mathbf{F}^n$ to compute $\mathbf{u}^{n+1}$ and p^{n+1} by solving the following linear system of equations:

$$\rho\left(\frac{\mathbf{u}^{n+1}-\mathbf{u}^n}{\Delta t} + \sum_{k=1}^{3} u_k^n D_k^{\pm}\mathbf{u}^n\right) = -\mathbf{D}^0 p^{n+1} + \mu\sum_{k=1}^{3} D_k^+ D_k^-\mathbf{u}^{n+1} + \mathbf{F}^n, \tag{3.5}$$

$$\mathbf{D}^0\cdot\mathbf{u}^{n+1} = 0. \tag{3.6}$$

Here $u_k^n D_k^{\pm}$ stands for upwind differencing:

$$u_k^n D_k^{\pm} = \begin{cases} u_k^n D_k^-, & u_k^n > 0, \\ u_k^n D_k^+, & u_k^n < 0. \end{cases}$$

Equations (3.5) and (3.6) are linear constant coefficient difference equations and, therefore, can be solved efficiently with the use of the fast Fourier transform (FFT) algorithm.

We now turn to the discretization of (2.3), (2.4). Let us assume, for simplicity, that $\Omega \subset \mathbb{R}^2$ is a rectangular domain over which all of the quantities related to the shell are defined. We will assume that this domain is discretized with mesh widths Δq_1, Δq_2 and that the computational lattice for Ω is the set

$$\mathbf{Q} = \{(i_1 \Delta q_1, i_2 \Delta q_2) \mid i_1 = 1, \ldots, n_1, \quad i_2 = 1, \ldots, n_2\}.$$

In step 2 the force $\mathbf{F}^n$ is computed using the following equation:

$$\mathbf{F}^n(\mathbf{x}) = \sum_{\mathbf{q} \in \mathbf{Q}} \mathbf{f}^n(\mathbf{q}) \delta_h(\mathbf{x} - \mathbf{X}^n(\mathbf{q})) \Delta \mathbf{q}, \tag{3.7}$$

where $\Delta \mathbf{q} = \Delta q_1 \Delta q_2$ and δ_h is a smoothed approximation to the Dirac delta function on $\mathbb{R}^3$ described below.

Similarly, in step 4 the update of the position of the immersed material $\mathbf{X}^{n+1}$ is done using the equation

$$\mathbf{X}^{n+1}(\mathbf{q}) = \mathbf{X}^n(\mathbf{q}) + \Delta t \sum_{\mathbf{x}} \mathbf{u}^{n+1}(\mathbf{x}) \delta_h(\mathbf{x} - \mathbf{X}^n(\mathbf{q})) h^3, \tag{3.8}$$

where the summation is over the lattice $\mathbf{x} = (hi, hj, hk)$, where i, j, and k are integers.

The function δ_h, which is used in (3.7) and (3.8), is an approximation to the Dirac delta function. It is constructed to have compact support in order to reduce the complexity of the computation. The most commonly used δ_h spreads the force (in phase 2) of a single material point to a $4 \times 4 \times 4$ cube within the fluid grid. Similarly, in step 4 of the algorithm the fluid velocity of a $4 \times 4 \times 4$ cube within the fluid grid is needed to determine the fluid velocity of an immersed material point. This function is defined by the following equation:

$$\delta_h(\mathbf{x}) = h^{-3} \phi\left(\frac{x_1}{h}\right) \phi\left(\frac{x_2}{h}\right) \phi\left(\frac{x_3}{h}\right),$$

where

$$\phi(r) = \begin{cases} \frac{1}{8}(3 - 2|r| + \sqrt{1 + 4|r| - 4r^2}), & |r| \le 1, \\ \frac{1}{2} - \phi(2 - |r|), & 1 \le |r| \le 2, \\ 0, & 2 \le |r|. \end{cases}$$

For an explanation of the construction of δ_h, see [20].

4. The distributed algorithm. As a consequence of the mixed Eulerian–Lagrangian formulation of the immersed boundary method, the fluid and the immersed material are described by different data structures, all of which must be distributed among available processors. The main challenge in the design of the distributed algorithm lies in achieving a load-balanced computation while keeping the communication costs low. Furthermore, there are several natural design requirements that we impose on the algorithm. The algorithm must perform well for systems with sparse one- or two-dimensional immersed boundaries, as well as for systems with thick three-dimensional immersed structures. The immersed boundary may or may not move much during the simulation, and the performance of the algorithm should not depend on the mobility of the immersed boundary. In practice, the ability to easily integrate complex models of immersed material structures into the simulation model is very important. We would like to maintain this property of the immersed boundary method in distributed computations.

We begin the description of the algorithm by listing several basic design choices:

- The fluid is partitioned into a number of domains, which are assigned to all available processors.
- The immersed boundary is described by a collection of grids, each of which is wholly assigned to one of the available processors.
- The assignment of the data structures to processors is constant throughout the computation: Neither fluid points, nor material points migrate between processors.

Previous attempts to develop a distributed immersed boundary algorithm were based on different design choices, for example, splitting immersed boundary structures among processors. Our assumptions significantly simplify the development of the distributed algorithm, and while they appear to be somewhat restrictive, the resulting algorithm is nevertheless efficient, flexible, and scalable.

4.1. Interprocessor communication. As mentioned in the introduction, the interprocessor communication on most available distributed computers must be carried out in bulk, in order to avoid the penalty of high latency. Fine-grain communication is the main reason for the difficulties with scalability encountered in previous attempts to develop a distributed immersed boundary algorithm. All interprocessor communications in our algorithm use the following simple mailbox model: Each processor maintains a mailbox for incoming mail and a mailbox for outgoing mail for every processor, including itself. The act of sending mail from processor i to processor j means that the jth outgoing mailbox of processor i is copied onto the ith incoming mailbox of processor j.

4.2. Force computation. The force computation phase is the only part of the immersed boundary time step which depends on the specific structure of the immersed material. In phases 2 and 4 of the time step the material is treated simply as an unordered collection of points. In most applications force computation is by far the cheapest part of the immersed boundary time step.

The assumption that all data structures related to a given piece of the immersed boundary are completely contained in the memory of some processor implies that the computation of the force in phase 1 is entirely local, i.e., requires no communication, and is very efficient. However, we will see that in order to carry out the remaining, more expensive, phases of the immersed boundary time step efficiently, it is necessary to distribute the immersed material structures as evenly as possible among all available processors. In practice, certain portions of the immersed boundary may be quite large, leading to load imbalance in the computation. Our experience has shown that it is generally possible to subdivide such large structures among processors in an efficient way. For example, the basilar membrane in the cochlea model is a relatively large structure, modeled as an elastic shell. In order to achieve a load balanced computation we have partitioned the shell into a number of components assigned to different processors. As a result, the computation of the elastic force of the shell requires a preliminary communication phase in which boundary data for each component are exchanged between processors that own neighboring shell components. Each processor bundles the data it communicates, sending a single message to each of its neighbors. The total amount of data exchanged in this phase is typically very small, and the force computation phase remains significantly cheaper than the other phases of the immersed boundary time step. In order to simplify the discussion, in the remainder of this paper we assume that

4.3. The fluid solver. The solution of the discretized Navier–Stokes equations (3.5)–(3.6) is the central and most expensive part among the four parts that compose each time step of the first-order immersed boundary numerical method. The general framework of the immersed boundary method allows the use of fluid solvers different from the one described in section 3. Our design ensures that this remains true in the distributed case as well. It will be convenient, however, to describe the approach in the case of the specific fluid solver we have implemented.

Our fluid solver is perhaps the simplest possible distributed solver. It depends on the availability of efficient distributed three-dimensional FFT functions. We assume that the transform data is partitioned into slabs. When a transform of an $N_1 \times N_2 \times N_3$-point array $A[i_1, i_2, i_3]$ is computed using p processors (where N_1 is divisible by p), processor q stores the data slab

$$A[i_1, i_2, i_3]: \quad i_1 = \frac{N_1}{p}q, \ldots, \frac{N_1}{p}(q+1)-1, \quad i_2 = 0, \ldots, N_2-1, \quad i_3 = 0, \ldots, N_3-1.$$

Accordingly, fluid variables, such as velocity, pressure, and body force, are stored in such slabs: U_1, U_2, U_3, P and F_1, F_2, F_3. The application of the upwind differencing operator in (3.5) makes it necessary, however, to pad each such slab with two ghost planes corresponding to $i_1 = \frac{N_1}{p}q - 1$ and $i_1 = \frac{N_1}{p}(q+1)$. (Notice that because of the periodicity of the domain the calculation of the index i_1 is carried out modulo N_1.) In addition to the fluid variables mentioned above, we will also require three global fluid variables C_1, C_2, and C_3 for temporary storage, each occupying one slab per processor. The variables C_1, C_2, and C_3 store complex data on which a complex-to-complex Fourier transform is performed. The pressure variable is not required for time-stepping, but the pressure can be computed, if necessary, using additional storage, C_4, and one additional inverse Fourier transform computation. We require all three dimensions of the fluid slabs to be divisible by 4 (not counting the ghost planes in the first dimension), in order to facilitate the fluid cache implementation (see below).

We rewrite the Navier–Stokes equations (3.5)–(3.6) in the following form:

$$\rho\frac{\mathbf{u}^{n+1}}{\Delta t} + \mathbf{D}^0 p^{n+1} - \mu\sum_{k=1}^{3} D_k^+ D_k^- \mathbf{u}^{n+1} = \mathbf{C^n}, \tag{4.1}$$

$$\mathbf{D}^0 \cdot \mathbf{u}^{n+1} = 0, \tag{4.2}$$

where

$$\mathbf{C^n} = \rho\frac{\mathbf{u}^n}{\Delta t} - \rho\sum_{k=1}^{3} u_k^n D_k^{\pm}\mathbf{u}^n + \mathbf{F}^n. \tag{4.3}$$

The Navier–Stokes solver is shown in Procedure 2. At the beginning of this computation the slabs of U_1, U_2, and U_3 contain the values of the fluid velocity computed at the end of the nth time step. The first step is a preliminary communication phase where the ghost planes of the slabs corresponding to the three velocity components are exchanged. This can be done efficiently, with each processor sending a single message to each of its two neighbors. The barrier command synchronizes the processors, ensuring that all of the processors have completed their work before the next step can begin. At the end of the communication phase every processor has all the necessary data to compute the values stored in the slabs C_1, C_2, C_3 using (4.3). Notice that the right-hand side of (4.3) is complex data, while the left-hand side is real. The

PROCEDURE 2. The Navier–Stokes solver.

1: Exchange ghost planes for U_1, U_2, U_3;
2: *barrier*;
3: Compute C_1, C_2, C_3 using (4.3);
4: *barrier*;
5: Compute forward FFT of C_1, C_2, C_3;
6: *barrier*;
7: Solve transformed equations in C_1, C_2, C_3;
8: *barrier*;
9: Compute inverse FFT of C_1, C_2, C_3 and store the result in U_1, U_2, U_3.

Navier–Stokes equations (4.1)–(4.2) can now be solved using the distributed FFT. We assume that the FFT functions perform the transform "in place," so that the solution of the transformed equations is again a local computation requiring no communication. After the inverse Fourier transform has been completed, we copy the new fluid velocity from the complex C_1, C_2, C_3 into the real slabs U_1, U_2, U_3.

4.4. Fluid-structure interactions. The main difficulty in the design of the distributed immersed boundary algorithm lies in efficiently carrying out phases 2 and 4 of the immersed boundary time step. These phases require communication of information between the fluid grid and the immersed boundary grids. Computationally, these phases are expensive: On a single processor, depending on the immersed boundary system being simulated, each of these two phases may account for as much as 20–30% of the computing time.

In phase 2 immersed boundary forces must be communicated to the processors that own the appropriate portions of the fluid grid. In many applications, however, the immersed boundary interacts with only a small portion of the fluid grid, which may be owned by a fraction of all available processors. Communicating the immersed boundary forces to these processors will lead to a load imbalance in the computation. Furthermore, since each immersed boundary point interacts with a $4 \times 4 \times 4$ cube within the fluid grid, it is not easy to describe the set of fluid points that a given immersed boundary grid interacts with. To solve this problem we break up phase 2 into several stages and introduce a specialized cache-like data structure in each processor. We shall refer to this data structure simply as [illegible].

The force-spreading phase of the immersed boundary method is described in Procedure 3. First, each processor spreads the force of every immersed boundary grid point it owns into its fluid cache. At the end of this phase the processor fluid cache contains the representation of the forces applied to the portion of the fluid grid in the vicinity of the material grids possessed by that processor. It contains, however, only the forces contributed by the grids of that processor and not the forces due to grids owned by other processors. Next, every processor sends the contents of its cache to the processors that own the corresponding portions of the fluid force array. This communication is carried out in bulk: All information intended for a given recipient is bundled into the mailbox intended for that recipient and then sent as a single message. In the worst case, in this stage every processor sends a single large message to every other processor, including itself. Finally, when the communication has been completed every processor scans the messages it received and updates the portion of the fluid force array that it owns. At the end of this stage each force slab contains contributions from forces of [illegible] immersed material grids.

PROCEDURE 3. Force spreading for processor i $(1 \leq i \leq p)$.

```
1: for each immersed boundary grid do
2:   if the grid is owned by processor i then
3:     for every grid point do
4:       spread the grid point force f^n into the fluid cache;
5:     end for
6:   end if
7: end for
8: Pack the cache into mailboxes.
9: Send mail to each processor.
10: barrier;
11: for j = 1, ..., p do
12:   add the force data received from processor j to the force slab F_1, F_2, F_3;
13: end for
```

The first, local stage of force spreading is by far the most time-consuming part of the distributed procedure. Under most circumstances, the assignment of immersed boundary grids to processors at the start of the simulation can ensure that the total number of immersed boundary grid points owned by each processor is approximately the same. This will ensure a nearly perfect load balance distribution in the local force-spreading part of the computation. The communication stage depends on the total number of immersed boundary grid points and on their relative position. Significant savings can be achieved when portions of the immersed boundary that remain in proximity throughout the simulation are assigned to the same processor. In practice, this is usually known to the designer of the simulation experiment, who can carry out this assignment at the start of the simulation.

The final phase of the immersed boundary time step involves computing the new position of the immersed boundary by interpolating the new fluid velocity from the fluid grid to the immersed boundary grids. This phase is similar to the force-spreading phase, except that the fluid velocity data, instead of the fluid force data, is now communicated, and the information is propagating in the opposite direction: from the fluid grid to the immersed material grids. It is important to notice that the fluid portion that interacts with a given immersed boundary point in stage 2 is exactly the same as in stage 4. This information is stored in the fluid cache (see below) and is used in line 2 of Procedure 4.

4.5. The fluid cache. To finish the description of the computation we now provide the details of the fluid cache construction. We imagine the rectangular fluid grid as partitioned by a lattice into a collection of small $4 \times 4 \times 4$ cubes. Since we require the fluid slab dimensions to be divisible by 4, each fluid cube is completely contained in one slab. The fluid cache data structure in every processor consists of an $(N_1/4) \times (N_2/4) \times (N_3/4)$ array of pointers to cubes.

Initially this structure is empty, and a cube is allocated only when a point inside the cube is referenced. The fluid cache is easy to update, yet it does not contain an excessive amount of information. In addition to the pointer array, a linked list containing all the indices of the cubes that have already been allocated is maintained. Thus, at the end of the force-spreading phase each processor will have a list of all fluid cubes that interact with the portions of the immersed boundary that it owns. In effect, the data in the fluid cache describes within the fluid a narrow neighborhood of the

PROCEDURE 4. Move for processor i ($1 \le i \le p$).

```
1: for j = 1, ..., p do
2:    copy the appropriate portions of U_1, U_2, U_3 into the mailbox of processor j;
3: end for
4: Send mail to every processor.
5: barrier;
6: Unpack the mailboxes into the fluid cache.
7: for each immersed boundary grid do
8:    if the grid is owned by processor i then
9:       for every grid point do
10:         advance grid point coordinates using the velocity data in the fluid cache;
11:      end for
12:   end if
13: end for
```

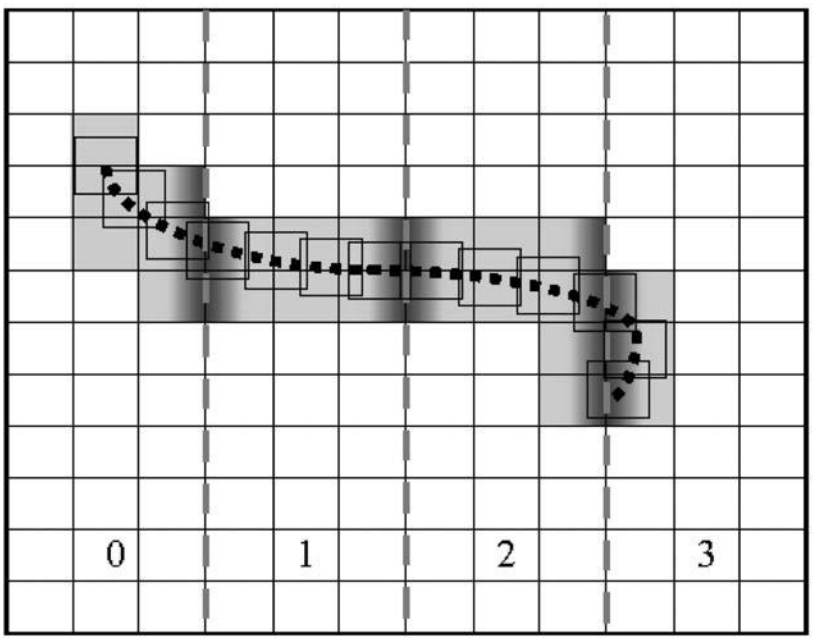


FIG. 4.1. *An illustration of the fluid cache: each point of the immersed boundary interacts with a* $4 \times 4 \times 4$ *cube of fluid around it; the shaded region describes the union of the fluid cache structures of all four processors. The cache of processor i is the intersection of slab i with the shaded region.*

immersed boundary owned by that processor as a collection of cubes (see Figure 4.1). The processor will then send cubes of fluid containing force data to processors that own these cubes. (Recall our assumption that fluid slab dimensions are divisible by 4, which ensures that each fluid cube is owned by exactly one processor.) Each recipient of a list of cubes containing fluid force data will record the cube indices and in the last stage of the algorithm it will return these cubes containing fluid velocity data to the sender.

Several issues related to cache management require further research. At present, we do not deallocate the cubes in the cache at the end of each time step. At the beginning of each time step we use the list of the cube indices to initialize all the allocated cubes to zero. This works well for portions of immersed boundary that do not move much during the simulation. For a highly mobile immersed boundary it may be necessary to implement a garbage collection mechanism in the fluid cache. While the size of the fluid cache in each processor is proportional to the total size of the fluid, our construction is practical because many of the currently available multiprocessor systems possess large amounts of memory (up to 32 GB per processor). Memory requirements for the fluid cache can be significantly reduced by implementing the cache using more complex data structures, but in such an implementation cache access may be slower. The $4 \times 4 \times 4$ size of the fluid cache cubes is not necessarily optimal. Good results have also been obtained with $4 \times 4 \times 1$ blocks. The choice of the block size in the fluid cache is not directly related to the width of support of the discrete delta function δ_h. With smaller cache blocks the total size of the cache in every processor is larger, while potentially less information needs to be communicated during the fluid-structure interaction phases.

5. The Titanium programming language. We have built a software package, called `IB`, implementing the distributed immersed boundary algorithm described above using the Titanium programming language. This section describes some of the features of the language that make it particularly suitable for large-scale scientific computing projects.

While much of scientific computing is still carried out in Fortran, the complexity of the simulated systems naturally leads to complex data structures. Modern multiprocessor computers require language-level tools to develop distributed algorithms, and programming languages have evolved to support object-oriented programming and to provide mechanisms for distributed programming. These tools have proved to be very successful in many fields, but they have rarely been used in the context of high-performance scientific computing because they often carry with them a performance penalty.

The Titanium programming language was developed with the goal of providing a modern language to meet the needs of scientific supercomputing. It is an explicitly parallel Java-based language designed to support high-performance scientific computing on large-scale multiprocessors, including massively parallel supercomputers and distributed-memory clusters with one or more processors per node [12, 25].

Titanium is a global address space language, closely related to UPC, Co-Array Fortran, and several older research languages based on C and C++. It combines the parallelism model commonly found in message passing modes with the shared address space found in shared memory models. In particular, Titanium uses a static parallelism (SPMD) model in which the number of parallel threads is determined at program startup time. It provides a global address space abstraction through which one thread may directly read or write to the memory of another thread, although the address space is logically partitioned so that each thread has a portion of the address space that is nearby. Programmers have control over data layout, synchronization, and load balancing for high performance, while the global address space simplifies programming but allows for direct expression of distributed data structures.

In spite of the global address space, the Titanium implementation runs on essentially any parallel machine, including shared memory multiprocessors, clusters of uniprocessors, and clusters of SMPs. On machines with hardware support for shared

memory, the compiler generates conventional load and store instructions to access memory that is associated with another thread. On a distributed memory platform, lightweight communication calls are inserted automatically when the accessed data structures reside on a remote processor. Titanium programs can run unmodified on uniprocessors, shared memory machines, and distributed memory machines—performance tuning may be necessary to arrange an application's data structures for distributed memory, but the functional portability allows for development on shared memory machines and uniprocessors.

Titanium preserves the safety properties of Java, which prevent accessing data that is unallocated through array bounds checking, strong typing, and automatic memory management. In addition to the parallelism model, which replaces conventional Java threads, Titanium adds support to improve programming for scientific applications. These include the following:

- user-defined immutable classes (often called "lightweight" or "value" classes);
- flexible and efficient multidimensional arrays with a rich set of operations for defining and manipulating the index set of an array;
- zone-based memory management, in addition to standard garbage collection;
- a type system for expressing and inferring locality and sharing properties of distributed data structures;
- compile-time prevention of deadlocks on barrier synchronization;
- a library of useful parallel collective operations such as barriers, broadcasts, and reductions;
- operator overloading;
- parameterized classes similar to a C++ style template.

6. The `IB` software package. The `IB` software package provides a collection of general-purpose classes necessary for the construction of immersed boundary simulations. This section briefly mentions some of the features of the package as an illustration of the importance of object-oriented programming in construction of large-scale computational models.

The fluid solver uses the FFTW software [8] to compute three-dimensional, distributed Fourier transforms. The fluid solver is contained in the `Fluid` class, which is defined by specifying fluid dimensions, mesh width, density, and viscosity:

```
Fluid fluid = new Fluid(N1, N2, N3, dx, rho, mu);
```

More complicated fluid models can be created using class inheritance. For example, the fluid in the heart model contains sources and sinks. Such a class can be defined as an extension of `IB.Fluid`. The new class will inherit many useful methods for manipulating fluid data structures, while overriding several methods, such as the fluid solver.

The specification of material grids and their type depends on the particular application. The `IB` package provides a number of classes for this purpose. Local properties of the material must be specified in a user-defined class derived from `IB.GridPoint`, and its global properties in a user-defined class derived from `IB.Grid`. For example, an elastic shell material type can be defined by deriving a class `Shell` and a class `ShellPoint`. The class `ShellPoint` naturally contains local elastic parameters of the material. On the other hand, the elastic force of the shell must be specified by defining a `ComputeForce` method within the `Shell` class.

The `IB` package also provides a `GridArray` class that enables handling hierarchical collections of material structures. In the cochlea model several surfaces are further subdivided into collections of grids; in the heart model the muscle fibers are arranged in groups, which are further arranged in layers.

Finally, the IB package provides a collection of methods to control the simulation and collect and analyze its output.

7. Software performance. Realistic immersed boundary models, such as the heart or the cochlea, require a significant amount of work to construct. We have therefore constructed a number of simple test models for the purpose of testing the distributed algorithm and tuning the performance of the IB software. Each such test model consists of a number of rectangular plates immersed in fluid. The precise nature of the elastic material is not important for the complexity of the immersed boundary computations because the elastic force calculation phase is negligible in time with respect to the other phases of the algorithm. The complexity of the computation is affected only by the total number of immersed boundary points and by their motion relative to the fluid. We have partitioned the immersed material into a number of grids, distributing an equal number of grids and an equal number of points among the processors. We expect to be able to achieve such a distribution in a realistic simulation. The heart, for example, is modeled by thousands of distinct fibers, which can be easily assigned to processors to achieve an even balancing of the load.

In immersed boundary computations, refining the fluid grid necessitates refining the immersed material grids accordingly. This is required to prevent the fluid leaking through the immersed boundary (for volume conservation in immersed boundary computations, see [21]). The cochlea model consists of a number of surfaces and, assuming the fluid grid is of size N^3, its total number of immersed boundary points is proportional to N^2. The heart muscle, on the other hand, though being modeled by a collection of one-dimensional fibers, occupies a three-dimensional volume, and therefore the total number of immersed boundary points in the heart model is proportional to N^3. Accordingly, we have tested the performance of the software on several models with varying numbers of immersed boundary points. Each of our test models contained a number of identical $N \times N$-point plates. We considered models with fluid grid of size N^3, where $N = 256$ and $N = 512$, and the number of plates $n = 1, 16$, and $N/8$. Accordingly, in the discussion below we refer to a model with n plates of size $N \times N$ as an (N, n) model. The models with $n = 1$ possess very little immersed material and therefore provide an insight into the performance of the fluid solver part of the algorithm. The $n = 16$ models resemble the cochlea in the size of their immersed material, while the $n = N/8$ models resemble the heart. The flow in each test model was generated by a periodic sinusoidal force of small amplitude applied to each of the plates. After a short initial phase the performance of the algorithm for each subsequent time step was essentially the same.

Our experiments were carried out on the IBM SP RS/6000 at the National Energy Research Scientific Computing Center (NERSC). This is a distributed memory computer possessing a large number of 16-processor nodes (currently 380 nodes), where each node has between 16 and 64 GB of memory. All of our tests were carried out on either 1, 2, 4, or 8 nodes, with the total number of processors used being 16, 32, 64, or 128. Figure 7.1 demonstrates the scaling of our software when a varying number of processors is used to compute two test problems. For the $(256, 1)$ we also present the data from experiments with 1, 2, 4, and 8 processors, all within a single node. Because our present algorithm requires the width of the fluid slabs (without the ghost planes) to be divisible by 4, we can utilize at most 64 processors on 256^3-fluid problems, and at most 128 processors on 512^3-fluid problems. Table 7.1 summarizes the wall-clock time per time step results for a number of test models, as well as the total number of floating point operations performed (in billions). The computations with

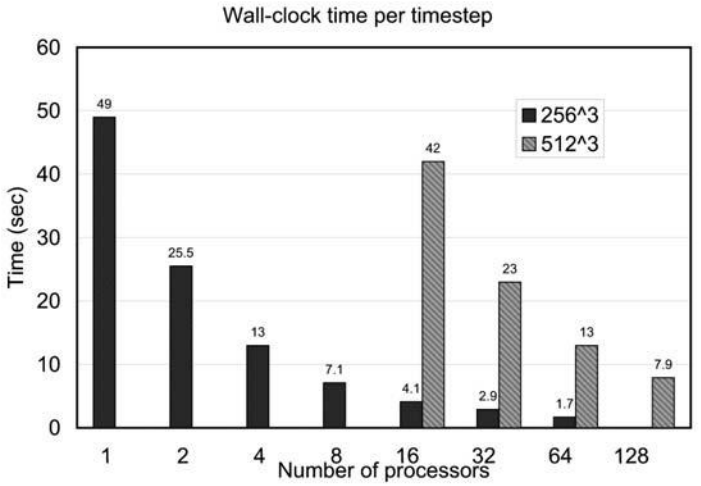


FIG. 7.1. *Execution wall-clock time (in seconds) per time step for the* $(256, 1)$ *and* $(512, 1)$ *models as a function of the number of processors.*

TABLE 7.1
Wall-clock time per time step for various test models.

Model name	Fluid grid size	Total immersed boundary size	Number of processors	Total GFLOPs	Wall-clock time
(256, 1)	256^3	$256^2 = 64$K points	64	4.34	1.7 sec
(256, 16)	256^3	$16 \times 256^2 = 1$M points	64	5.5	2.4 sec
(256, 32)	256^3	$32 \times 256^2 = 2$M points	64	6.78	3.1 sec
(512, 1)	512^3	$512^2 = 256$K points	128	38.4	7.9 sec
(512, 16)	512^3	$16 \times 512^2 = 4$M points	128	43.2	9.6 sec
(512, 64)	512^3	$64 \times 512^2 = 16$M points	128	58.1	15.2 sec

the 256^3-point fluid grid were performed on 64 processors, while the 512^3-point fluid grid computations used 128 processors.

Finally, Table 7.2 shows the timings of the main phases of each time step. The second row refers to the spreading of the forces from the immersed boundary into the local fluid cache structure. Rows 6–10 detail the performance of the fluid solver: "Upwind" refers to the initial step of the fluid solver which uses the slabs F_1, F_2, F_3 and applies the upwind differencing operator to U_1, U_2, U_3. Three Fourier transforms are applied to the components of the result of this computation. The fluid equations are then solved in the Fourier space, and the solution is obtained by means of inverse Fourier transforms. The total time devoted to the fluid solver is recorded in row 11. The last phase of the time step, named "move material," involves both the interpolation of the fluid velocity from cubes to immersed boundary grids and the updating of the immersed boundary position.

The results presented in Table 7.2 enable us to assess the scalability of the individual phases of the computation. The lines labeled with "spread force" and "move material" involve local computations with the fluid cache and show excellent scalability. It is interesting that a significant amount of time is necessary to perform local copy operations ("pack" and "unpack") to and from the fluid cache. No floating point operations are performed in these phases. The communication phases show excellent scaling: Because only fluid information is communicated, doubling the number of

TABLE 7.2
Breakdown of the wall-clock time per time step for various test models. (See explanation in the body of the article.)

	(256, 16)	(256, 32)	(512, 16)	(512, 64)
Compute material force	0.015	0.027	0.028	0.08
Spread force	0.25	0.49	0.52	1.97
Pack force cubes	0.012	0.021	0.023	0.08
Send force cubes	0.22	0.28	0.52	1.4
Update force slabs	0.08	0.1	0.43	0.84
Upwind	0.29	0.29	1.49	1.46
3× FFT	0.39	0.38	2.12	2.1
Solve	0.06	0.06	0.25	0.25
3× FFT^{-1}	0.39	0.41	2.21	2.29
Copy fluid velocity	0.06	0.06	0.24	0.25
Total (fluid solver)	1.3	1.31	6.63	6.78
Pack velocity cubes	0.04	0.07	0.14	0.47
Send velocity cubes	0.23	0.28	0.58	1.57
Unpack velocity cubes	0.01	0.02	0.034	0.078
Move material	0.22	0.47	0.47	1.86
Total (time step)	2.4	3.09	9.39	15.16

immersed boundary points may require much less than double the amount of fluid information to be communicated.

8. Summary and conclusions. We have developed an efficient algorithm for immersed boundary simulations on distributed memory multiprocessor systems. The main difficulty in the design of the algorithm arises because of the need to communicate information between the fluid and the immersed boundary data structures, which are distributed across multiple processors. The difficulty of achieving load balance and maintaining low communication cost in the fluid-structure interaction is exacerbated by the hardware-imposed restriction that the interprocessor communication be carried out in bulk. We resolve this problem with the help of a specialized cache-like data structure, the fluid cache, to communicate information between processors. The fluid cache enables an efficient implementation of the fluid-structure interaction portion of the immersed boundary time step.

We have implemented the algorithm in the software package `IB` using the Titanium programming language. In our implementation the fluid solver uses the FFTW functions for efficient distributed FFT computations.

The `IB` package utilizes the object-oriented features of Titanium and its built-in facilities for generating efficient code, and provides the user with a set of versatile classes and utilities to build large-scale immersed boundary applications. A 512^3 cochlea model based on the `IB` package has been tested, and Peskin and McQueen's 128^3 heart model has already been reconstructed using the `IB` package. A 256^3 heart model is currently being built. This work will be reported in future publications. In this paper we described the development of the basic algorithm and the software performance results that enable such large-scale immersed boundary computations.

At present, these are the largest models that are practical. Indeed, since a single typical experiment with the cochlea model may require more than 10,000 time steps, we estimate that in the case of a 512^3-point fluid grid it will take more than 3 days to complete. Simulating a single beat of the heart, on the other hand, requires on the order of 100,000 time steps. Such an experiment with a 256^3 model could be completed in fewer than 3 days.

The main practical limitation in using the FFTW is the limit in the amount of parallelism that is achievable: The FFTW requires partitioning the fluid data along the first dimension and at most N_1 processors can be deployed in a transform of size $N_1 \times N_2 \times N_3$. In order to be able to simulate larger immersed boundary systems it is necessary to develop an efficient fluid solver which uses more than N_1 processors. Such a fluid solver will require a more complex partition of the fluid into subdomains. We expect the extension of our fluid cache construction, and the distributed algorithm we have presented, to the case of the more general fluid partition to be straightforward.

Our algorithm assumes that the number of immersed boundary points is constant throughout the computation and the partition of these points to processors is carried out at the beginning of the computation and is held fixed throughout the computation. The case where immersed material is either created or destroyed during the simulation introduces an additional difficulty. In such a system the immersed boundary data structures may become concentrated in only a few of the available processors, leading to an unbalanced computation. This problem can be addressed by allowing the immersed boundary data structures to migrate between processors, but periodic reassignment of immersed boundary data structures to processors may be computationally expensive.

REFERENCES

[1] *Titanium Project Home Page*, http://www.cs.berkeley.edu/projects/titanium/ (24 January 2006).

[2] C. Bell, D. Bonachea, Y. Cote, J. Duell, P. Hargrove, P. Husbands, C. Iancu, M. Welcome, and K. Yelick, *An evaluation of current high-performance networks*, in Proceedings of the 17th International Parallel and Distributed Processing Symposium (IPDPS), 2003.

[3] R. P. Beyer, *A computational model of the cochlea using the immersed boundary method*, J. Comput. Phys., 98 (1992), pp. 145–162.

[4] R. Dillon, L. J. Fauci, and D. Gaver, *A microscale model of bacterial swimming, chemotaxis and substrate support*, J. Theoret. Biol., 177 (1995), pp. 325–340.

[5] C. D. Eggleton and A. S. Popel, *Large deformation of red blood cell ghosts in a simple shear flow*, Phys. Fluids, 10 (1998), pp. 1834–1845.

[6] L. J. Fauci and A. L. Fogelson, *Truncated Newton method and modeling of complex immersed elastic structures*, Comm. Pure Appl. Math., 46 (1993), pp. 787–818.

[7] L. J. Fauci and C. S. Peskin, *A computational model of aquatic animal locomotion*, J. Comput. Phys., 77 (1988), pp. 85–108.

[8] M. Frigo and S. G. Johnson, *FFTW: An adaptive software architecture for the FFT*, in Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, vol. 3, 1998, pp. 1381–1384.

[9] E. Givelberg, *Modeling elastic shells immersed in fluid*, Comm. Pure Appl. Math., 57 (2004), pp. 283–309.

[10] E. Givelberg and J. Bunn, *A comprehensive three-dimensional model of the cochlea*, J. Comput. Phys., 191 (2003), pp. 377–391.

[11] E. Givelberg, J. J. Bunn, and M. Rajan, *Detailed simulation of the cochlea: Recent progress using large shared memory parallel computers*, in Proceedings of the 2001 International Mechanical Engineering Congress, New York, 2001.

[12] P. Hilfinger, D. Bonachea, D. Gay, S. Graham, B. Liblit, G. Pike, and K. Yelick, *Titanium Language Reference Manual*, Technical report UCB//CSD-01-1163, Computer Science Division (EECS), University of California, Berkeley, CA, 2001.

[13] E. Jung and C. S. Peskin, *Two-dimensional simulations of valveless pumping using the immersed boundary method*, SIAM J. Sci. Comput., 23 (2001), pp. 19–45.

[14] M.-C. Lai and C. S. Peskin, *An immersed boundary method with formal second order accuracy and reduced numerical viscosity*, J. Comput. Phys., 160 (2000), pp. 705–719.

[15] D. M. McQueen and C. S. Peskin, *Computer-assisted design of pivoting-disc prosthetic mitral valves*, J. Thorac. Cardiovasc. Surg., 86 (1983), pp. 126–135.

[16] D. M. McQueen and C. S. Peskin, *Shared-memory parallel vector implementation of the immersed boundary method for the computation of blood flow in the beating mammalian heart*, J. Supercomputing, 11 (1997), pp. 213–236.

[17] C. S. Peskin, *Flow Patterns around Heart Valves: A Digital Computer Method for Solving the Equations of Motion*, Ph.D. thesis, Albert Einstein College of Medicine, Yeshiva University, New York, 1972.

[18] C. S. Peskin, *The immersed boundary method*, Acta Numer., 11 (2002), pp. 479–517.

[19] C. S. Peskin and D. M. McQueen, *A three-dimensional computational method for blood flow in the heart.* I. *Immersed elastic fibers in a viscous incompressible fluid*, J. Comput. Phys., 81 (1989), pp. 372–405.

[20] C. S. Peskin and D. M. McQueen, *A general method for the computer simulation of biological systems interacting with fluids*, in Proceedings of the SEB Symposium on Biological Fluid Dynamics, Leeds, England, 1994.

[21] C. S. Peskin and B. F. Printz, *Improved volume conservation in the computation of flows with immersed elastic boundaries*, J. Comput. Phys., 105 (1993), pp. 33–46.

[22] M. E. Rosar and C. S. Peskin, *Fluid flow in collapsible elastic tubes: A three-dimensional numerical model*, New York J. Math., 7 (2001), pp. 281–302.

[23] H. G. Sabbagh, *Solving the Navier–Stokes Equations on a Distributed Parallel Computer*, Ph.D. thesis, New York University, New York, 1996.

[24] S. M. Yau, *Experiences in Using Titanium for Simulation of Immersed Boundary Biological Systems*, Master's report, 2002.

[25] K. Yelick, L. Semenzato, G. Pike, C. Miyamoto, B. Liblit, A. Krishnamurthy, P. Hilfinger, S. Graham, D. Gay, P. Colella, and A. Aiken, *Titanium: A high-performance Java dialect*, Concurrency: Practice and Experience, 10 (1998), pp. 825–836.

[26] L. Zhu and C. S. Peskin, *Simulation of a flapping flexible filament in a flowing soap film by the immersed boundary method*, J. Comput. Phys., 179 (2002), pp. 452–468.

SIAM J. SCI. COMPUT.
Vol. 28, No. 4, pp. 1379–1397
© 2006 Society for Industrial and Applied Mathematics

3-D MESH ADAPTATION ON MULTIPLE WEAK DISCONTINUITIES AND BOUNDARY LAYERS*

LAKHDAR REMAKI† AND WAGDI G. HABASHI†

Abstract. This paper presents new developments within an anisotropic mesh adaptation methodology, made essentially by the introduction of a metric modification and a shock-filter model that allow crisp capturing of weak shocks, as well as boundary layers and wakes. Inviscid and laminar external flow tests are used to demonstrate the efficiency of the method.

Key words. mesh adaptation, metric, shock filter

AMS subject classifications. 76A02, 76D05, 74S05, 65N50, 34G20, 35A05

DOI. 10.1137/S1064827503429879

1. Introduction.

Unstructured meshes have made significant inroads in industrial applications of computational fluid dynamics (CFD) due to their inherent ability to more easily discretize complex geometrical domains and incorporate adaptive strategies. It is well established that such meshes will play a growing role in the numerical simulation of inviscid and viscous flows. The mesh adaptation strategy [1], [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17] is demonstrated to be a powerful tool to improve the solver accuracy and to handle large-scale simulations. To increase the accuracy and reduce the computational costs, the strategy consists of refining the grid in regions where physical details require more nodes to be well captured while coarsening in regions where the density of nodes is more than needed. The whole procedure is driven by an error indicator which is the core of the method. Different ways are proposed in the literature to compute this error, among them indicators based on large gradients of certain physical features; see, for instance, [3], [4], [6], [11], [14]. This approach achieves good results but unfortunately shows some shortcomings due to the fact that high errors are associated with large gradients, which is not necessary true for some situations; a linear solution could show very large gradients. In [7], [8], [9], a robust and efficient three-dimensional automatic mesh adaptation methodology, with CAD integrity, has been developed. The adaptation procedure uses an a posteriori interpolation error estimate, whose magnitude and direction are controlled by the matrix of local second derivatives of a selected flow variable. This error is projected over the mesh edges and drives the nodal movement algorithm, as well as the edge refinement and coarsening, and faces swapping strategies. This method exploits the notion of curvature of the solution; a Riemannian metric is derived from the Hessian of the solution, and then the Euclidean distance is replaced by a Riemannian distance. Consequently, by considering the surface defined by the solution as a manifold, edge lengths are close to geodesics between the nodes defining the edge, which is more natural.

Another interesting method that allows handling errors based on integral quantities rather than derivatives is the adjoint method. Solving the linear adjoint problem

*Received by the editors June 13, 2003; accepted for publication (in revised form) September 6, 2005; published electronically August 16, 2006.
http://www.siam.org/journals/sisc/28-4/42987.html
†NSERC-J. Armand Bombardier Industrial Research Chair of Multi-disciplinary Computational Fluid Dynamics Laboratory, McGill University, 688 Sherbrooke Street West, Montreal H3A 2S6, QC, Canada (lakhdar.remaki@cfdlab.mcgill.ca, wagdi.habashi@mcgill.ca).

allows connecting the indicator error directly to the local residual error of the primer solution. This method is well known and extensively used for a large range of optimization problems. Details of the formulation of the method to mesh adaptation could be found in [3], [18], [19], and important improvements and extensions are proposed in [11].

In the presence of multiple shocks, whatever the adaptation method used, one needs several solution-adaptation cycles to capture all shocks. Meshes tend to migrate towards the stronger shocks, at the detriment of the weaker ones, and it is almost impossible to capture the weaker shocks to the same level of accuracy. In this work we propose a preprocessing method that allows capturing all shocks, no matter their strength, in a limited number of adaptation cycles. This method consists of a combination of two complementary strategies. First, a correction to the metric is made to force the nodes to move towards discontinuities, even weak ones (such as diffuse shocks). Second, a shock filter is applied to the solution in order to enhance all shocks. Since boundary layers and wakes could also be considered smeared discontinuities, the method also yields an improvement in boundary layers and wake capturing. A considerable improvement is obtained, greatly enhancing the accuracy of CFD calculations, still considered to be the stumbling block for the adoption of the simulation over testing in cases where accuracy is of importance. Note that in this paper we will not do a comparison between methods since what we propose is a preprocessing that could be suitable for a great number of mesh adaptation methods. A comparison will be done with and without the proposed preprocessing for a given method to show its impact on mesh adaptation efficiency. The method developed in [7], [8], [9] is selected for such a purpose.

In section 2 an overview of the selected mesh adaptation methodology is given. The proposed metric correction and shock-filter model are detailed in section 3. The effectiveness of the proposed approach is demonstrated through numerical tests in section 4. Finally, conclusions are drawn in section 5.

2. Overview of the selected mesh adaptation methodology. As mentioned in the introduction, to demonstrate the efficiency of the proposed preprocessing, the mesh adaptation method developed in [7], [8], [9] is selected. This methodology, first developed by Peraire, Perión, and Morgan [14], Habashi et al. [7], and D'azevedo and Simpson [20], is coded in a home software, OptiGrid, in which the new developments are implemented. In this section a short survey of the methodology is given; for more details see the quoted references. The method is an h-adaptation method that includes node movement, edge refinement, coarsening, and swapping for hybrid grids consisting of any combinations of tetrahedra, prisms, hexahedra, and pyramids.

It is well known that for finite element and finite volume methods the computational error is bounded by the interpolation error, which is proportional to the second derivatives for the linear interpolation (see, for instance, [6], [7], [20]). This bound holds for the norm of the Sobolev space to which the solution of the continuous problem belongs. In the case of Navier–Stokes equations, which are our concern, the error estimation holds for the $\|.\|_{H^1}$ norm. The goal of mesh adaptation is to build a mesh in such a way that the interpolation error given by $\|u - r_h u\|_{H^1}$, with r_h the interpolation operator, is minimal. It is claimed in [6], [7] that a way to minimize this error is to equidistribute the interpolation pointwise error function $E(x) = |u(x) - r_h u(x)|$ over the whole computational domain.

Recall that the second derivative of a function g in a direction $\vec{v}$ ($\vec{v}$ is a unit

vector) is given by

$$\langle H\vec{v}, \vec{v}\rangle,$$

where H is the Hessian of g. When H is positive definite it defines a metric; therefore the error along a given vector could be interpreted as the length of the vector measured by the Hessian metric. Consequently, a metric is derived from the Hessian by considering its absolute value $|H| = R^t\,|\Lambda|\,R$, with R being the eigenvectors matrix of H and $|\Lambda| = \text{diag}\,\{|\lambda_1|, |\lambda_2|, |\lambda_3|\}$ the matrix composed by the absolute values of the eigenvalues of H. The adaptation procedure is then driven by the a posteriori error estimate based on the metric $|H|$ of a selected scalar flow variable. The eigenvectors give the direction of the stretching, while the eigenvalues give its magnitude, thus gradually creating anisotropy. As mentioned above, equidistributing the error on the original grid generates the adapted grid. The a posteriori error indicator ε along an edge $\vec{x}$ is computed as

$$\varepsilon = \int_0^1 \sqrt{\langle |H(s)|\,\vec{x}, \vec{x}\rangle} ds.$$

Note that, if e is the estimation of the interpolation error E using the second derivative's bound, the error indicator ε is related to e by $\varepsilon = h^2 e$, where h is the length of the edge $\vec{x}$.

In practice, we deal with a discrete scalar variable. Several methods are proposed in the literature to compute the Hessian of such functions. In this section we present the one used in our calculations. Let us describe first a method that computes a discrete gradient.

Let g be a given discrete scalar variable and n_j be a given node. We would like to estimate the gradient ∇g of g at node n_j. Let S_j be the set of nodes connected to node n_j and $card(S_j)$ the cardinality of S_j. Denote by g_k the value of g at node n_k and by $\vec{e}_{jk}$ the edge connecting nodes n_j and n_k. Using the first-order Taylor expansion we obtain

$$g_k - g_j = \nabla g.\vec{e}_{jk}. \tag{1}$$

Writing the equality (1) for all connected nodes to the concerned node n_j, we obtain the matricial relationship:

$$A\nabla g = B. \tag{2}$$

A is a $(card(S_j), 3)$ matrix (3 columns and $card(S_j)$ rows). In general, A is not a square matrix. The right-hand vector B is a $card(S_j)$-vector, and its components are given by $B_k = g_k - g_j$. Since A is not a square matrix, to approximate the gradient ∇g we seek a vector V that minimize the norm $\|AV - B\|_2 = \sqrt{\langle AV - V, AV - B\rangle}$, that is,

$$\nabla g \approx \underset{V \in IR^3}{\text{Inf}} \|AV - B\|_2\,. \tag{3}$$

The problem (3) is classical in optimization theory, and the solution is given by the pseudoinverse of the matrix A, namely

$$\nabla g \approx A^T \left[AA^T\right]^{-1} B. \tag{4}$$

Finally, the Hessian matrix is computed by applying the same process to the obtained gradient components.

The next sections discuss strategies for cost-efficient mesh adaptation. The intent is to minimize the number of solver-adaptation cycles, while at the same time making the approach more efficient and robust. The proposed preprocessing to achieve this goal is presented in detail.

3. Capturing multiple and weak discontinuities. It makes sense that the quality of the (initial) flow solution directly affects the ability of the error indicator to detect the dominant and secondary flow features, which will determine the outcome of the adaptation and provide even better solutions. On one hand, one aims at using as few points as possible for cost-effective CFD solutions and to minimize grid setup time; yet, on the other hand, the solution must be reasonably accurate for the salient features to at least be detected for adaptation. Obviously, the aim is to start with a mesh as coarse as possible, taking less time and effort to generate. However, when the initial grid is too coarse, discontinuities are badly defined, especially in the case of multiple shocks, whatever the solver used. As a remedy to enhance the adaptation of discontinuities, a shock-filter model is proposed. Furthermore, and in order to accelerate the adaptation process, a metric correction is added to force the nodes to move towards the discontinuities.

An adaptation process like the one described above requires a target error threshold selected by the user, in order to equidistribute the error around this threshold. However, the selection of this value is not evident, as error in itself is relative and has no scale, similar to a residual: it is just a relative measure of the magnitude and distribution of error. Choosing it in a heuristic fashion or on a trial-and-error basis can require a certain number of adaptation attempts to guess a suitable value and is thus not the most cost-effective approach. To overcome this shortcoming, we also propose a method, having its origin in signal processing, to automatically compute the starting value of the error density target on the initial mesh.

3.1. Automatic determination of error-density target. To illustrate the method, an error density distribution, obtained from a Mach solution of a laminar supersonic flow around a NACA0012 wing, is plotted in Figure 1(left). Note that there are distinct areas of low and high error. The sought-after error density is defined as the optimal threshold that separates these two areas and is obtained by maximizing the variance between them. This is a classical problem in statistics, also known in signal processing as Otsu's method [22]. Let $e(s),\ s = 1, \ldots, N_{nods}$, be the sorted error distribution over the N_{nods} nodes. The optimal threshold is given by $\beta = e(t^*)$, where $t^* = \text{ArgMax}(\delta_b^2)$ for the between-class variance:

$$\delta_b^2(t) = \frac{(\mu_T \omega(t) - \mu(t))^2}{\omega(t)(1-\omega(t))}, \tag{5}$$

where

$$\omega(t) = \sum_{k=1}^{t} p(k), \quad p(k) = \frac{n_k}{N_{nods}}, \quad n_k \text{ number of nodes with error } e(k),$$

$$\mu(t) = \sum_{k=1}^{t} e(k)p(k), \quad \text{and } \mu_T = \sum_{k=1}^{N_{nods}} e(k)p(k).$$

Figure 1(right) shows a binary image obtained by assigning a value of "one" to nodes with an error larger than the threshold β and of "zero" for nodes with an

error lower than β. This demonstrates the efficiency of Otsu's threshold to optimally separate the two areas of high and low error. This method is used in all test cases presented in this paper.

Note that there are many ways to compute this error threshold in the literature; however, most of them are weighted-average based methods. This makes their results depending on the whole domain size to the body size ratio, which is not desirable since this value is user dependent. Using the proposed Otsu threshold overcomes this shortcoming, which turns the method more robust.

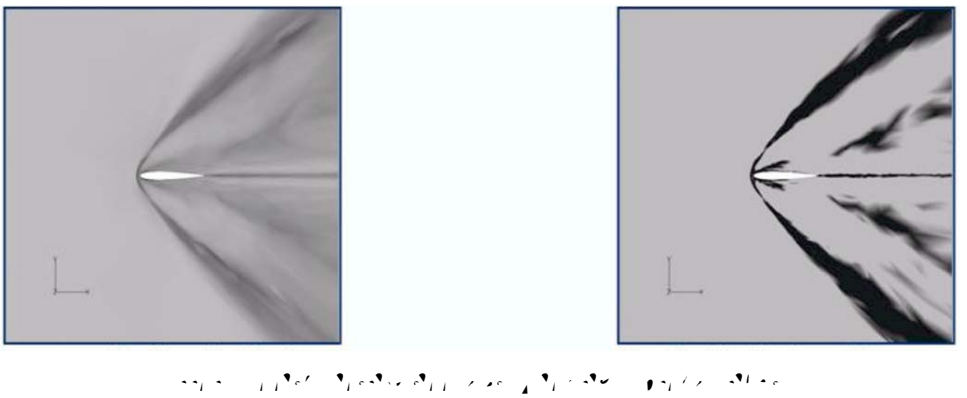


FIG. 1. *Error density distribution and Otsu's threshold binarization.*

3.2. Metric modification to accelerate adaptation of weak discontinuities. Let M be the metric obtained by taking the absolute value of the Hessian as described above, that is,

$$M = |H| = R^t\,|\Lambda|\,R. \tag{6}$$

Let e_i be a given edge; the error along the edge is given by

$$|e_i|_M^2 = \langle Me_i, e_i\rangle.$$

The idea of metric modification is to add to the edge error, the error of the edge projection on the solution gradient as shown on the Figure 2, namely

$$\|e_i\|_M^2 = \langle Me_i, e_i\rangle + \langle M(P_{\nabla U}e_i), (P_{\nabla U}e_i)\rangle,$$

where $P_{\nabla U}$ is the projection operator on the gradient of the solution∇U.

This will increase the error for edges that are oriented in the direction of the gradient, and they will be moved to be perpendicular to the gradient. This means that the corresponding connected nodes will move toward the shocks, since shocks are perpendicular to the scalar gradient.

The new error could be written as

$$\|e_i\|_{\bar{M}}^2 = \langle \bar{M}e_i, e_i\rangle, \tag{7}$$

where

$$\bar{M} = M + P_{\nabla U}MP_{\nabla U}. \tag{8}$$

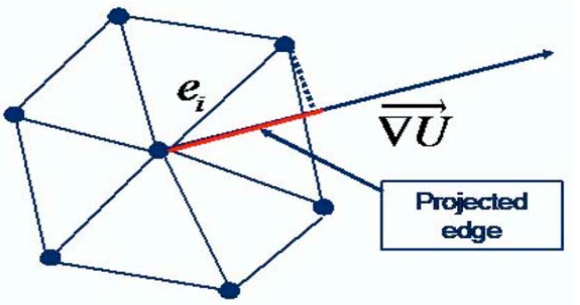


FIG. 2. *Illustration of the metric correction idea.*

In conclusion, the error correction is handled by a correction of the metric by adding the quantity $P_{\nabla U} M P_{\nabla U}$.

This metric correction could be seen as an upwinding of the original metric in the direction of the gradient. It makes sense since the maximum error around discontinuities and boundary layers is produced in the gradient direction. Therefore, and as in the case of several numerical methods, this shifting increases the stability and the robustness of the method. However, the main advantage of this correction is the convergence acceleration. Indeed, the node movement velocity, for instance, is described by the error intensities and orientations of the connected edges. Thus, by increasing the error intensities in the direction of discontinuities, what the metric correction allows us to do, nodes migrate faster towards these discontinuities.

Figure 3 shows the impact of metric correction on the adaptation for an oblique shock, using a hexahedral mesh. The test case is a supersonic flow on a ramp (more details of this test case are given below). As expected, the node movement is accelerated towards the shock in comparison to the case of adaptation without metric modification. This is demonstrated by the increase of the nodes' concentration around the shock. This obviously will reduce the necessary number of cycles of adaptation to converge to the right solution.

3.3. Shock-filter preprocessing (deconvolution) to enhance weak discontinuities. To enhance (sharpen) weak discontinuities, a shock-filter model, based on a quasi-linear hyperbolic equation, is proposed. It creates shocks at the zero-crossing of the Laplacian in the gradient direction:

$$(\mathrm{S}) \quad \begin{cases} \partial_t u^\varepsilon + F\left(\partial_\eta u^\varepsilon f'(u^\varepsilon), \partial_{\eta\eta} u_0^\varepsilon\right) \partial_\eta f(u^\varepsilon) = 0 \text{ on } IR^+ \times \Omega, \\ u^\varepsilon(0,x,y,z) = u_0^\varepsilon(x,y,z) \text{ on } \Omega, \\ \partial_n u^\varepsilon = 0 \text{ on } IR^+ \times \partial\Omega, \end{cases}$$

where Ω is an open set of IR^3 and $\partial_n$is the derivative in the outer normal direction $\vec{n}$ to $\partial\Omega$. The initial condition is smoothed in the model by a convolution product with a smooth function called a mollifier, that is, $u_0^\varepsilon = u_0 * \rho_\varepsilon$, where $*$ is a convolution product, $\rho_\varepsilon = \frac{1}{\varepsilon^3}\rho(x/\varepsilon, y/\varepsilon, z/\varepsilon)$, ρ is the mollifier, and ε is a scale parameter. This is a classical way to regularize the initial condition u_0, and the main result by doing so is that u_0^ε converges to u_0 when ε tends to zero in the L^p $1 \le p < \infty$ spaces. ∂_η and $\partial_{\eta\eta}$ are the first and second derivatives' operators in the gradient direction $\vec{\eta}$, and the functions F and f are regular (smooth) functions.

The function F, called the shock detection function, controls the position where the model must develop shocks that strengthen weak discontinuities. It satisfies the

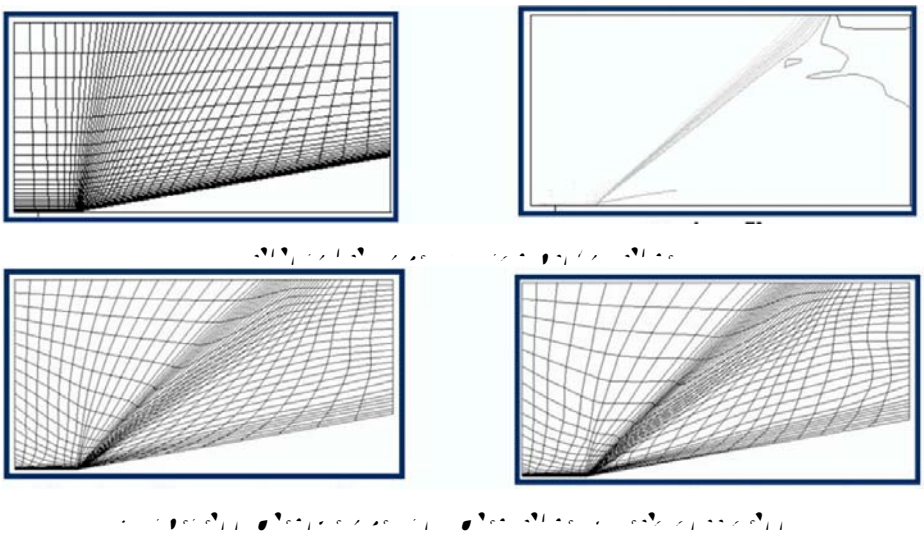


FIG. 3. *Inviscid supersonic flow on a ramp. The effect of metric correction: the concentration of nodes around the shock is more important when the metric correction is used.*

relationship $srF(s,r) > 0\ \forall (s,r) \in IR^* \times IR^*$, where $IR^* = IR \setminus \{0\}$. With the second derivatives the model distinguishes the concave and convex parts of the initial condition. The relationship satisfied by F allows the model to increase the concave parts and decrease the convex ones while leaving invariant the zero-crossing parts of the second derivatives leading to a creation of strong discontinuities (shocks). Note that the gradient direction is selected to compute derivatives because the gradient is perpendicular to shocks.

For the applications presented in this paper, we took $F(u,v) \equiv Sign(u)Sign(v)$. In theory, the function $Sign(.)$ must be regularized to obtain a C^∞ function. In practice, however, and since all functions are discrete functions, we do not need the regularization. Finally, note that any other type of functions that satisfy a shock detection condition such as sigmoid can be used. The function f controls the shock velocity according to the characteristics of the original solution (initial condition). The theoretical study (existence and uniqueness of a solution) and numerical study of such a model are developed in detail in [23], [24], [25].

To illustrate the effect of the shock filter, a one-dimensional gate signal is smoothed and a high-frequency sinusoidal noise is added. Figure 4 shows the result of applying the model to restore the signal, with the parameters $f(u) = \frac{1}{2}u^2$ and $F(u,v) \equiv Sign(u)Sign(v)$. The signal is restored quite well after only 5 time iterations.

Now, for adaptation purposes we consider the steady state of the shock filter, but before we do, we must make sure that this steady state exists. The following proposition proves such a claim.

PROPOSITION 3.1. (S) $\bar{U} \in L^\infty(\Omega)$.

. In [24], the following maximum principle for such equation is proven.

THEOREM 3.2 (maximum principle). u^ε (S) $IR^+ \times IR^3$ u_0^ε (x,y,z) $\text{Inf}\ u_0^\varepsilon(x,y,z) \leq u^\varepsilon(t,x,y,z) \leq \text{Sup}\ u_0^\varepsilon(x,y,z)$ $IR^+ \times IR^3$

Now let us give the proof of the existence of the steady state of (S). First, consider

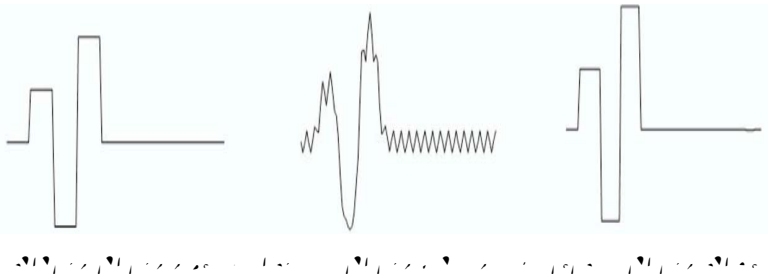


FIG. 4. *Restoration of a noisy and destroyed signal by the shock-filter process.*

the equivalent integral form of the equation:

$$u^\varepsilon(t,x,y,z) = u_0^\varepsilon(x,y,z) - \int_0^t F\left(\partial_\eta u^\varepsilon f'(u^\varepsilon), \partial_{\eta\eta} u_0^\varepsilon\right) \partial_\eta f(u^\varepsilon) dt. \tag{9}$$

Let t_n be a numerical nonnegative sequence that tends to infinity when n tend to infinity, and let (x_0, y_0, z_0) be a point such that $\partial_{\bar\eta\bar\eta} u_0^\varepsilon(x_0, y_0, z_0) > 0$.

The relationship $rsF(r,s) > 0$ implies that $\int_0^{t_n} F\left(\partial_\eta u^\varepsilon f'(u^\varepsilon), \partial_{\eta\eta} u_0^\varepsilon\right) \partial_\eta f(u^\varepsilon) dt > 0$. This, along with the maximum principle cited above, implies that $u^\varepsilon(t_n, x_0, y_0, z_0)$ is a monotone bounded sequence, which implies that it converges to a value denoted by $\bar u^\varepsilon(x_0, y_0, z_0)$.

Now let us show that this latter quantity does not depend on the sequence t_n. Let t_n and s_n be two sequences that tend to infinity, and consider the difference $|u^\varepsilon(s_n, x_0, y_0, z_0) - u^\varepsilon(t_n, x_0, y_0, z_0)|$. Let p be an integer such that $s_n < t_{n+p}$. Since $u^\varepsilon(t_n, x_0, y_0, z_0)$ is monotone, we have

$$|u^\varepsilon(s_n, x_0, y_0, z_0) - u^\varepsilon(t_n, x_0, y_0, z_0)| < |u^\varepsilon(t_{n+p}, x_0, y_0, z_0) - u^\varepsilon(t_n, x_0, y_0, z_0)|.$$

If $u^\varepsilon(t_n, x_0, y_0, z_0)$ is a [illegible] sequence, then $|u^\varepsilon(s_n, x_0, y_0, z_0) - u^\varepsilon(t_n, x_0, y_0, z_0)|$ tends to zero. This proves that there exists a steady state $\bar U$ such that $\bar U(x_0, y_0, z_0) = \bar u^\varepsilon(x_0, y_0, z_0) \forall (x_0, y_0, z_0) \in \Omega$. Moreover, and according to the maximum principle, $\bar U \in L^\infty(\Omega)$. For the case $\partial_{\bar\eta\bar\eta} u^\varepsilon(x_0, y_0, z_0) < 0$, we proceed analogously and achieve the same conclusions. □

As a corollary of this proposition, the steady state is a piecewise constant function. Consequently, the shocks are enhanced as shown in Figure 4, but certain useful curvatures may be lost and the adaptation in such areas may degrade. Thus, we propose adapting on an average scalar solution between the original solution and the steady state of the shock filter. This allows enhancing the discontinuities and preserving the curvatures since the steady state is a piecewise constant function. Figure 5 illustrates this claim.

For the rest of the paper, when we refer to the shock-filter output, we mean the steady state averaged by the initial condition. In the following subsection, a short discussion on the numerical aspect of the shock-filter model is given.

3.3.1. Numerical approximation. To apply the shock filter we need to perform two calculations: first compute the initial condition smoothing $u_0^\varepsilon = u_0 * \rho_\varepsilon$ and then compute the shock-filter solution iteratively to reach the steady state. For the smoothing calculation, the Gaussian kernel is selected as a mollifier. This choice is motivated by the fact that the convolution product with a Gaussian is equivalent to

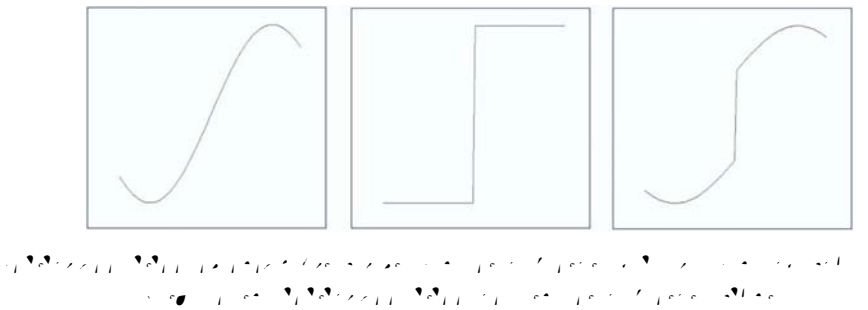


FIG. 5. *The average value of the original and steady state of the shock-filter model as scalar function of adaptation. The shock is enhanced and the curvature is preserved.*

solving the heat conduction equation, namely

$$\begin{cases} \partial_\tau u - \Delta u = 0, \\ u(0,.) = u_0, \end{cases} \quad \text{with } \varepsilon = \tau. \tag{10}$$

Then, instead of computing the convolution product, which is time-consuming, we solve (10).

For the discretization of the shock-filter equation itself, note first that for a given regular function g, the first and second derivatives in the gradient direction $\vec{\eta}$ are given by

$\partial_{\vec{\eta}} g = \sqrt{(\partial_x g)^2 + (\partial_y g)^2 + (\partial_z g)^2}$ and $\partial_{\vec{\eta}\vec{\eta}} g$

$$= \frac{(\partial_x g)^2 \partial_{xx} g + (\partial_y g)^2 \partial_{yy} g + (\partial_z g)^2 \partial_{zz} g + 2\partial_x g \partial_y g \partial_{xy} g + 2\partial_x g \partial_z g \partial_{xz} g + 2\partial_y g \partial_z g \partial_{yz} g}{(\partial_x g)^2 + (\partial_y g)^2 + (\partial_z g)^2}.$$

Any classical method such as FVM or first-order FEM could be used to discretize the equation, as one does not need high precision. Note that for both (S) and (10), the Neumann boundary condition $\partial_n u^\varepsilon = 0$ is considered.

We exploited the fact that the Laplacian operator does not change under axis rotation, which implies that if $(\vec{\zeta_1}(x,y,z), \vec{\zeta_2}(x,y.z))$ are two vectors such that $(\vec{\eta}, \vec{\zeta_1}, \vec{\zeta_2})$ forms an orthonormal reference at each point (x,y,z) of Ω, we have

$$\Delta u = \partial_{\vec{\eta}\vec{\eta}} u + \partial_{\vec{\zeta_1}\vec{\zeta_2}} u + \partial_{\vec{\zeta_2}\vec{\zeta_2}} u;$$

then we propose solving both heat conduction and shock-filter equations by the following explicit finite difference scheme.

We consider first a finite element discretization of the domain and classical first-order shape function for interpolation purposes. Now let $N_j = (x_j, y_j, z_j)$ be a given node on the mesh and h_j the minimum length of the connected edges to the node N_j. Let $K^+_{\vec{\eta}}$, $K^-_{\vec{\eta}}$, $K^+_{\vec{\zeta_1}}$, $K^-_{\vec{\zeta_1}}$ $K^+_{\vec{\zeta_2}}$, $K^-_{\vec{\zeta_2}}$ be the elements connected to node N_j, such that $K^+_{\vec{\eta}}$ contains the point $N_j + h_j\eta$ and $K^-_{\vec{\eta}}$ contains the point $N_j - h_j\eta$, and so on. Let $u^n_{j,1}$, $u^n_{j,-1}$, $u^n_{j,2}$, $u^n_{j,-2}$,$u^n_{j,3}$,$u^n_{j,-3}$ be the interpolated values (using first-order shape functions) at the positions $N_j + h_j\eta$, $N_j - h_j\eta$, $N_j + h_j\zeta_1$, $N_j - h_j\zeta_1$, $N_j + h_j\zeta_2$, $N_j - h_j\zeta_2$, respectively, and for the time step n. Then we make the following space approximations:

$$\partial_{\bar{\eta}\bar{\eta}} u^n_j \approx u^n_{j,1} - u^n_j + u^n_{j,-1}, \quad \partial_{\bar{\zeta_1}\bar{\zeta_1}} u^n_j \approx u^n_{j,2} - u^n_j + u^n_{j,-2}, \quad \partial_{\bar{\zeta_2}\bar{\zeta_2}} u^n_j \approx u^n_{j,3} - u^n_j + u^n_{j,-3},$$

$$\partial_{\bar{\eta}} u_j^n \approx \begin{cases} u_j^n - u_{j,-1}^n, & F(f'(u_j^n)(u_{j,1}^n - u_{j,1}^n), \partial_{\bar{\eta}\bar{\eta}} u_j^0) > 0, \\ u_{j,1}^n - u_j^n & \text{else.} \end{cases}$$

By replacing these quantities in the continuous equations and by using the Euler time discretization, we obtain an explicit centered finite difference scheme for the heat conduction equation and an explicit upwind finite difference scheme for the shock-filter equation. The CFL condition is obviously respected for the stability of the schemes.

Before showing the impact of the metric and shock-filter model on the adaptation on weak and multiple shocks, let us see the effect of the shock filter on a physical solution. Figure 6 shows the effect of shock enhancement filter on a Mach number solution of an inviscid flow around a NACA0012 airfoil: on the left is the original Mach profile and on the right is the shock-filter steady state. The figure shows a significant enhancement of the shocks that are much more diffusive in the original solution. Finally, note that the obtained solution of problem (S) has no physical meaning; it is used for adaptation purposes only. For readers that are not familiar with the NACA0012 test case and the Mach number notion, definitions are given in the appendix as well as very short introductory of the mechanics of the fluid.

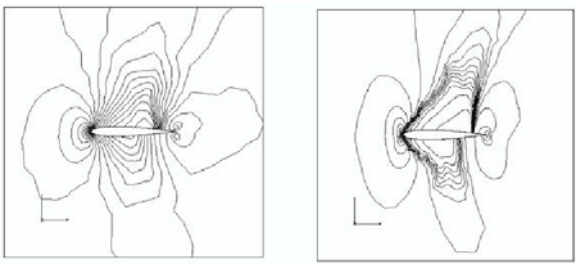


FIG. 6. *The enhancement effect of the shock filter.*

4. Numerical results.

4.1. Convergence improvement. Let us first see the impact of the proposed preprocessing (shock filter and metric correction) on the solution convergence. For this purpose an inviscid supersonic flow on a ramp is chosen. The angle of attack (AoA) is taken to be equal to 0 and the Mach number (M) equal to 2. We compare the results obtained on the original grid and those obtained after two cycles of adaptation in both situations (with and without the preprocessing). Figure 7 shows the grids and the Mach profile as well as the corresponding residual curves. The figure shows clearly the improvement of the convergence rate after mesh adaptation with a substantial improvement when the preprocessing is used. Note also the effect of the preprocessing filter on the shock-capturing quality. More convincing effects of the preprocessing on the whole process of adaptation solution are demonstrated through the following test cases.

4.2. Shock capturing. The following figures show the effect of shock-filter preprocessing and metric correction on capturing shocks. Figure 8 shows a transonic inviscid flow around a NACA0012 at an angle of attack AoA = 1 degree and Mach M= 0.85. The solution is obtained on a coarse grid. Note that one of the two shocks is quite smeared. Figure 9 shows different adapted grids and corresponding solutions obtained without shock-filter preprocessing and metric correction. As shown in the

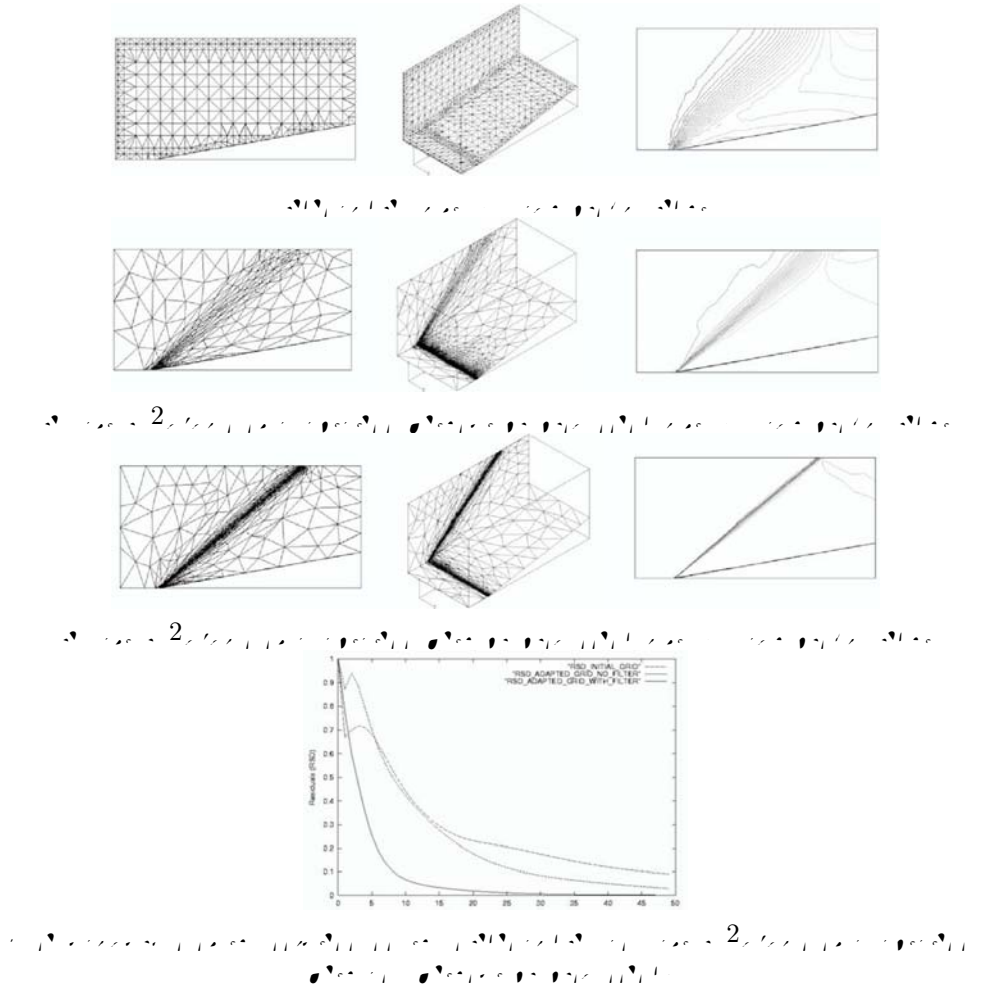


FIG. 7. *Inviscid supersonic flow around a ramp;* $A0A = 0, M = 2.0$.

figure, 7 solution-adaptation cycles are necessary to converge to the solution. Figure 10 shows the same test case using the proposed shock filter and metric correction. One can see that the shocks are well captured after only 3 solution-adaptation cycles and that, moreover, the results are improved for both mesh and solution.

The next test is that of an inviscid flow around a NACA0012 at AoA =1.25, M=0.8. Figure 11 shows the initial grid and the corresponding Cp profile: one can see that one of the shocks is very weak compared to the other and is smeared. The test demonstrates the capability of the proposed method to capture shocks in such extreme situations where the amplitudes of shocks are sensibly different. Figure 12 shows the adaptation without shock filter and metric correction. After 6 cycles, the weak shock is still not captured, and, moreover, the mesh is coarsened around the position of that shock. This means that even if the adaptation process continued, the shock would not be captured unless the number of nodes were sufficiently increased, which is contrary to the philosophy of mesh optimization. Figure 13 shows the effect of the shock-filter and metric modification preprocessing: both shocks are well captured

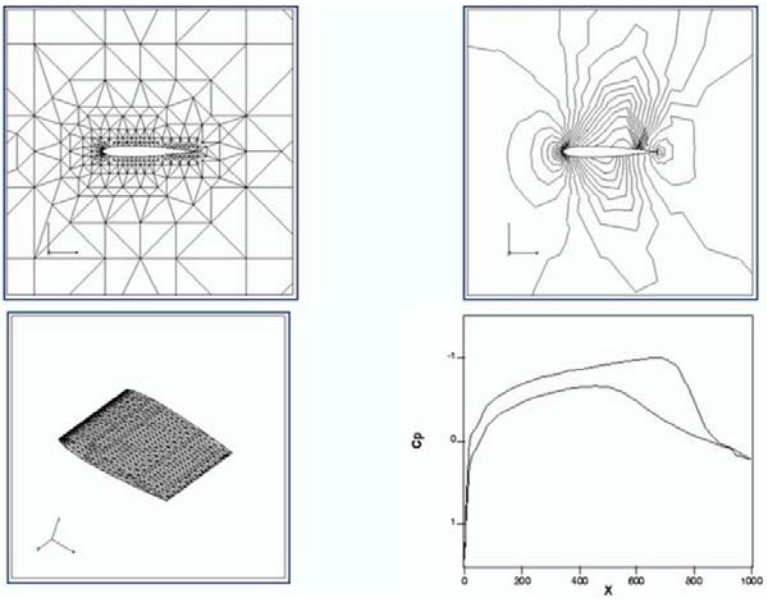


FIG. 8. *NACA*0012 *inviscid flow;* $AoA = 1, M = 0.8$. *The original grid and the Cp solution.*

with a smaller number of adaptation cycles (4) and without increasing the number of nodes, namely 50,000 nodes.

1. A question that naturally arises is what would happen if an artificial shock is created by the shock-filter, which is possible especially for internal flows. First, as mentioned in section 3.3 the adaptation is done on the steady state of the shock-filter equation averaged by the original solution; as a result any prejudicial coarsening of regions of interest in terms of physical features is avoided. Consequently, in such a situation a high concentration of nodes (more then necessary) is produced in a region where the artificial shock is detected; therefore this will not affect the result of the solver. Moreover, when all physical shocks are captured by the solver we stop applying the preprocessing since it becomes unnecessary, and then by doing only one more adaptation cycle the unneeded concentration of nodes disappears. Note that based on the explanation given above, the last operation is not necessary; it is more for aesthetic than for convergence purposes. The test case of Figure 13 illustrates very clearly this situation. The shock filter seems to consider the extension around the leading edge as a weak shock. The first three adapted grids show an artificial shock starting from the leading edge and growing to the back, while the corresponding solutions are not at all affected. Moreover, this artificial shock disappears in the last grid where the adaptation is done without the shock filter.

4.3. Effect on boundary layers and wakes. Even if the proposed shock-filter model was originally developed to enhance shocks and allow capturing weak and multiple shocks, this filter is also able to improve boundary layers and wake capturing. As with shocks, it improves the quality of the results and reduces the solution-adaptation cycles. Let us first see the effect of the shock filter on a laminar solution; considering the boundary layers and wakes to be smeared discontinuities, Figure 14 shows how the shock filter is able to sharpen the boundary layer thickness

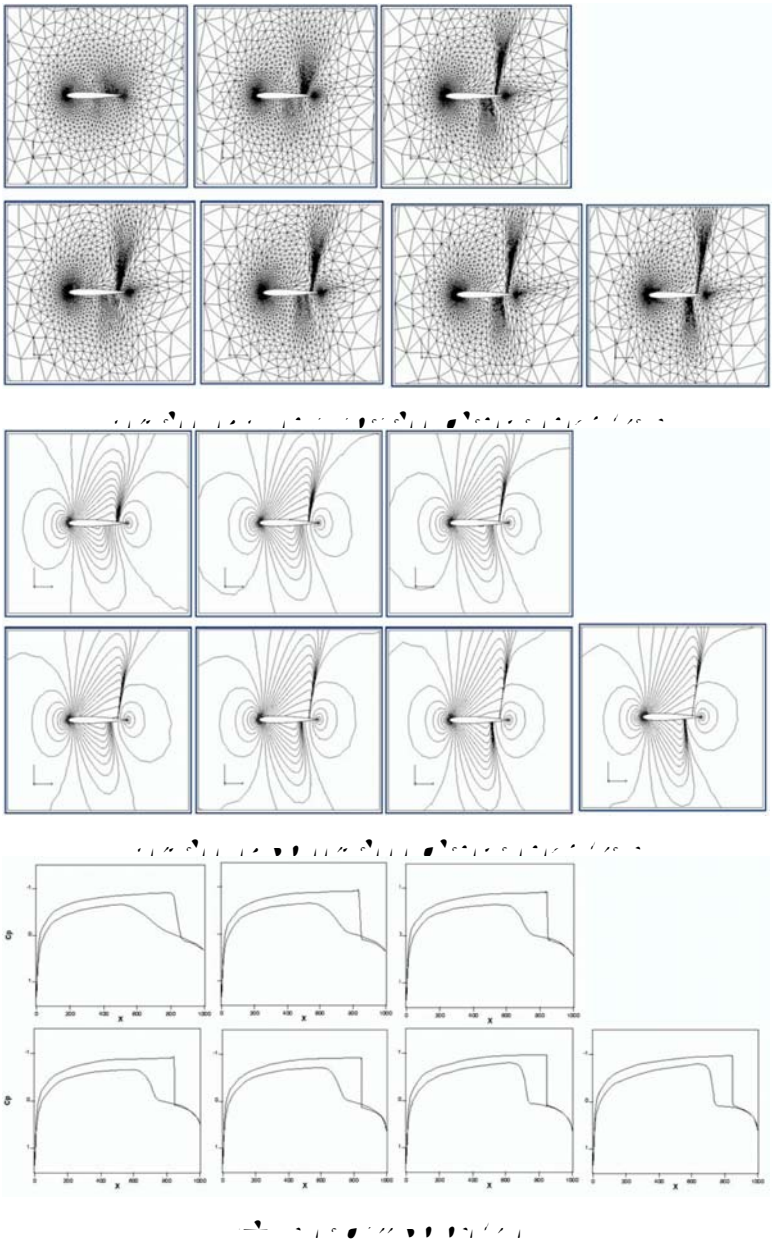


FIG. 9. *NACA*0012 *inviscid flow;* $AoA = 1, M = 0.8$. *Different cycles of adaptation without shock filter and metric correction.*

and to reinforce the wake.

Figures 15 and 16 show a laminar supersonic test case around a NACA0012 wing at AoA=0, M=2, and Re=2000. The results demonstrate that, in addition to sharpening weak discontinuities, the shock-filter and metric correction preprocessing improve the boundary layers and wake capturing. The solution and grid achieve a high quality, as well as a sensitive reduction of the solution-adaptation cycle numbers (from 7 to 3 for this test case), as in the case of inviscid shock solutions.

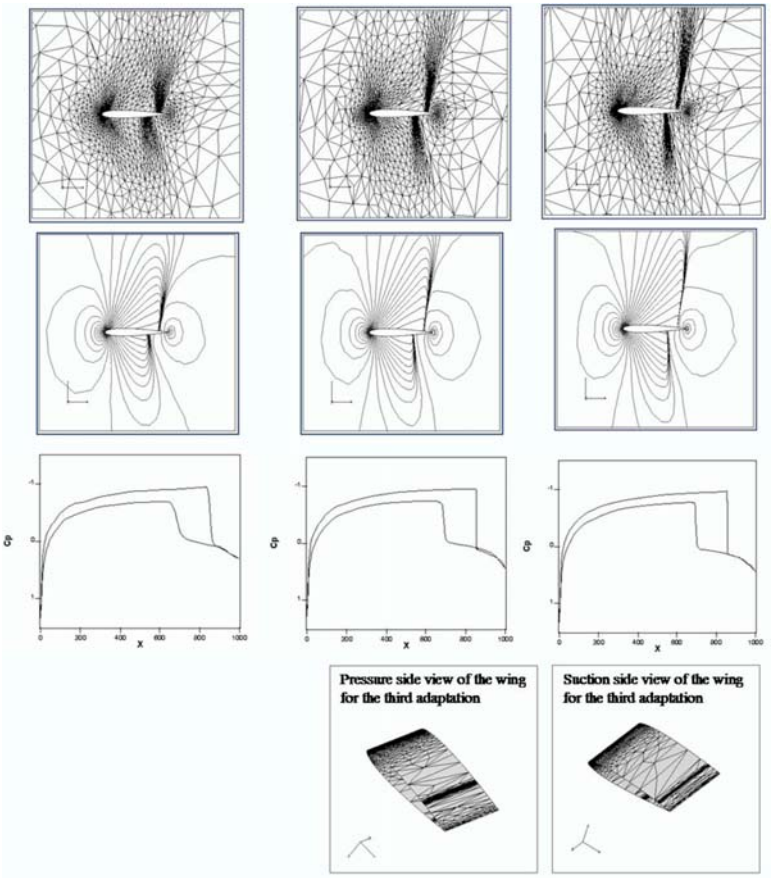


FIG. 10. *NACA*0012 *inviscid flow; AoA* $= 1, M = 0.8$. *Different cycles of adaptation with shock filter and metric correction.*

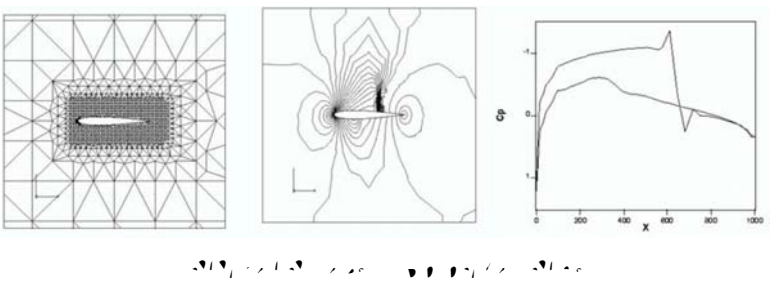


FIG. 11. *NACA*0012 *inviscid flow; AoA* $= 1.25, M = 0.85$. *The original grid and the Cp solution.*

Finally, note that the cost of the proposed preprocessing is negligible and does not increase the complexity of the global adaptation process.

5. Conclusions. This paper has proposed an original and efficient method to crisply capture weak discontinuities in multiple shocks cases, as well as boundary layers and wakes. At the same time, the method significantly reduces the number of solution-adaptation cycles. The method consists of complementary techniques. First,

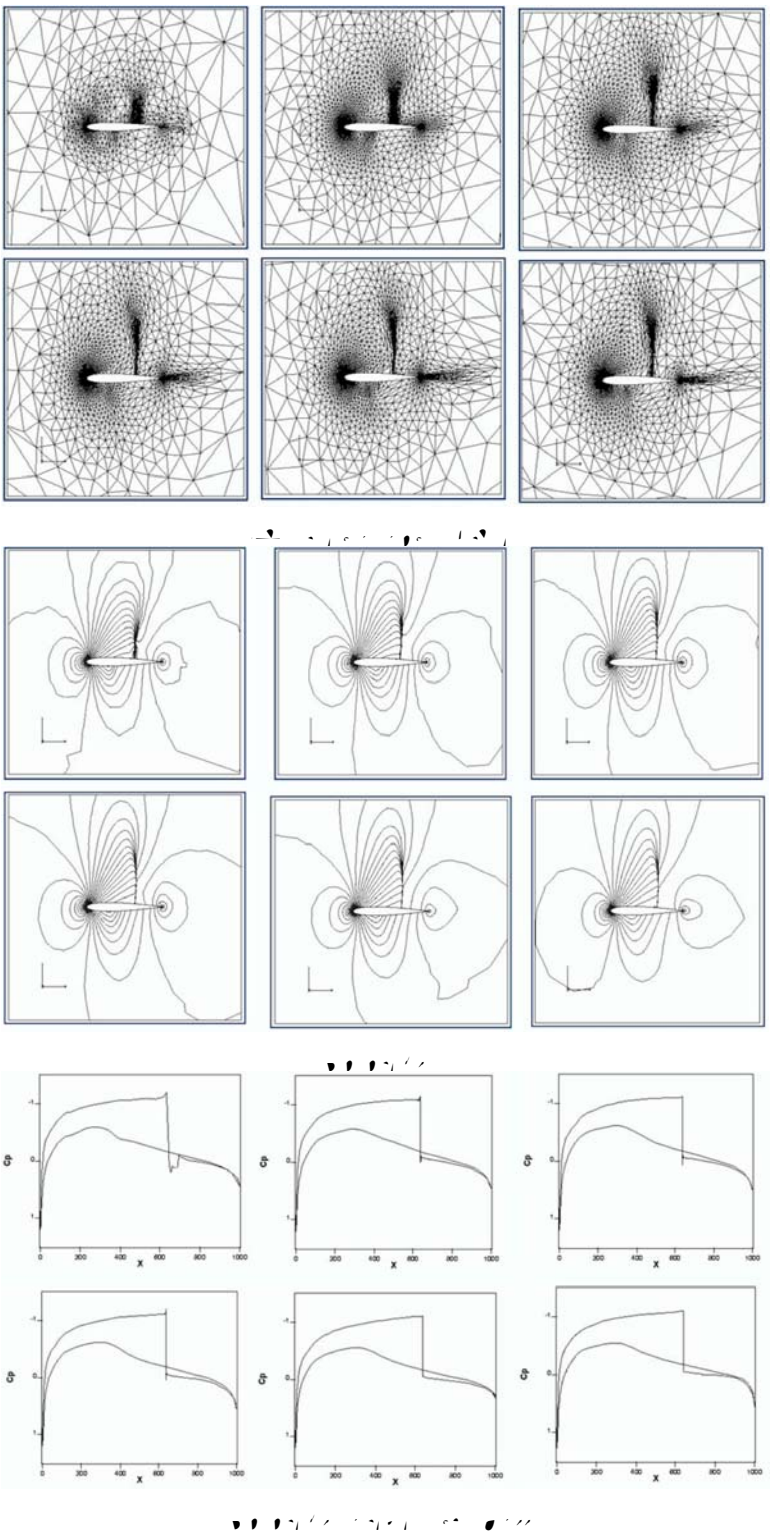


FIG. 12. *NACA*0012 *inviscid flow;* $AoA = 1.25, M = 0.8$. *Different cycles of adaptation without shock filter and metric correction.*

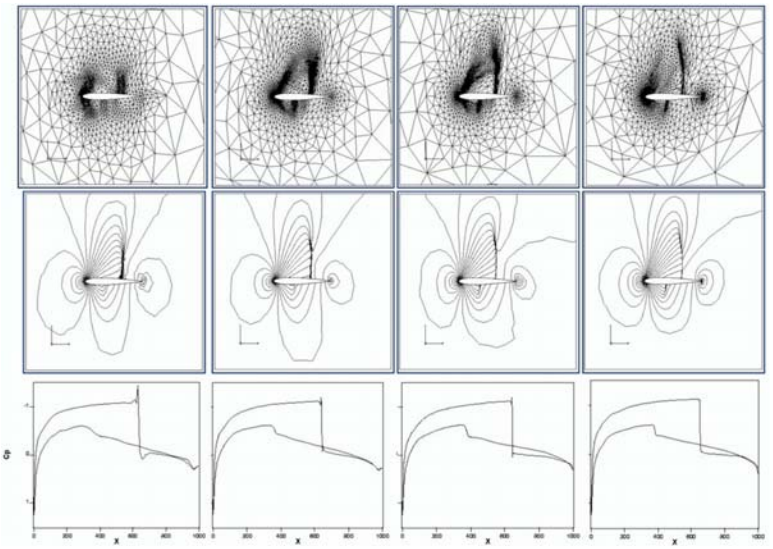


FIG. 13. *NACA*0012 *inviscid flow; AoA* $= 1.25, M = 0.8$. *Different cycles of adaptation with shock filter and metric correction.*

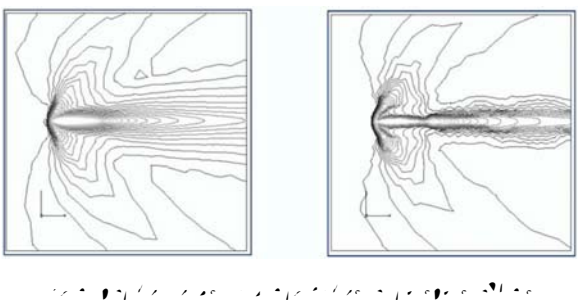


FIG. 14. *NACA*0012 *laminar supersonic flow. The effect of shock filter on boundary layer and wake.*

a shock filter is applied to the solution to enhance shocks, boundary layers, and wakes. Then a modification of the metric is proposed to speed up adaptation, by forcing the nodes to move toward the desired areas. The output of the shock filter is averaged by the selected original solution in order to conserve curvatures. The shock filter is modeled by a hyperbolic quasi-linear equation whose well-posedness was studied in previous works. However, a proof of the existence of a steady state is given within this paper. An automatic target error value is also proposed in this paper. Finally, the efficiency of the method is demonstrated by means of inviscid and laminar test cases, the general adaptation scheme having been shown to be equally and successfully applicable to turbulent flow cases, as well.

Appendix. Some definitions. The motion of a fluid (such as water and gas) is governed by the following Navier–Stokes equations:

$\partial_t \rho + \nabla \bullet (\rho \vec{v}) = 0$ (continuity equation),

$\partial_t(\rho \vec{v}) + \nabla \bullet (\rho \vec{v} \otimes \vec{v}) + \nabla p = \rho \vec{f} + \nabla(\lambda \nabla \bullet \vec{v}) + \nabla \bullet (2\mu D)$ (momentum equation),

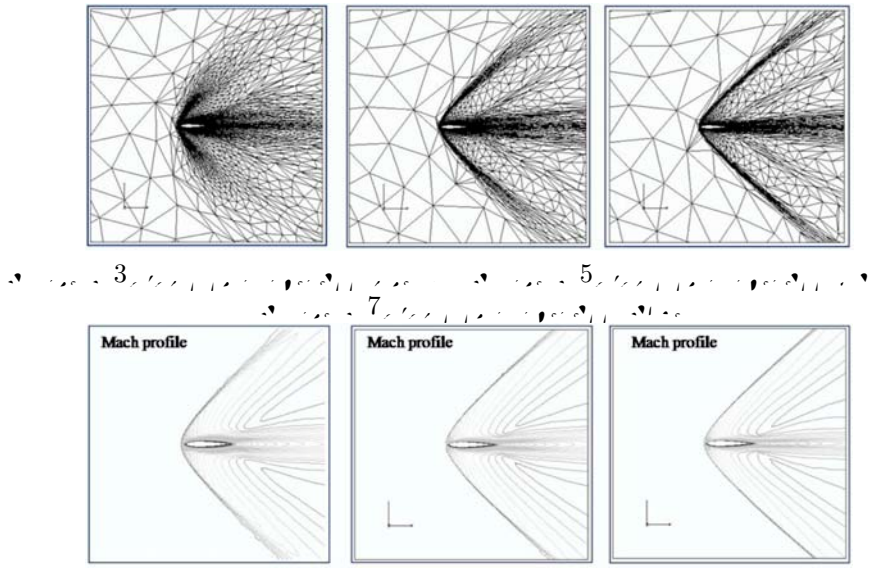


FIG. 15. *NACA*0012 *laminar supersonic flow;* $AoA = 0.0, M = 2.0$. *Different cycles of adaptation without shock filter and metric correction.*

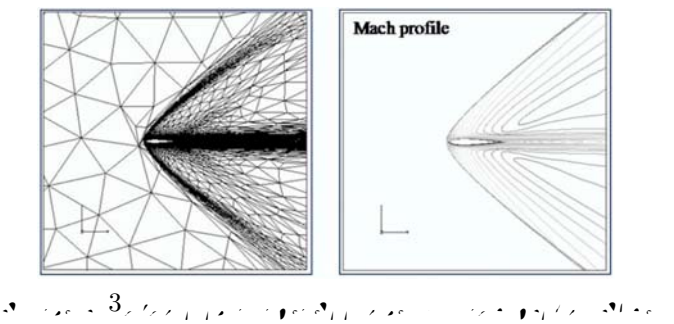


FIG. 16. *NACA*0012 *laminar supersonic flow;* $AoA = 0.0, M = 2.0$. *The third cycle of adaptation with shock filter and metric correction.*

$\partial_t(\rho E) + \nabla \bullet [(\rho E + p)\vec{v}] + \nabla p = \rho \vec{f} \bullet \vec{v} + \nabla \bullet (\kappa \nabla \theta) + \nabla \bullet (\lambda(\nabla \bullet \vec{v})\vec{v}) + \nabla \bullet (2\mu D\vec{v})$ (energy equation),

where ρ is the density, $\vec{v} = (v_1, v_2, v_3)^T$ the velocity of the fluid, $D = \frac{1}{2}(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i})_{ij}$ the deformation tensor, $\vec{f}$ the external force, $E = e + |\vec{v}|^2/2$ the specific total energy, e the specific internal energy, p the pressure, θ the temperature, λ, μ the Lamé viscosity coefficients, and κ the heat conductivity

The Navier–Stokes equations describe how the velocity, pressure, temperature, and density of the fluid are related. When the viscous effects are neglected the equations are called Euler equations and the flow is called inviscid. As we can see, these equations are composed of one equation of conservation of mass, three equations of momentum, and one equation of conservation of energy. Some important physical features are derived from the primitive physical variable mentioned above, among them the Mach number. Let us give describe this feature.

A.1. Mach number definition. As an aircraft moves through the air, the air molecules near the aircraft are disturbed and move around the aircraft. If the aircraft passes at a low speed, the density of the air remains constant. But for higher speeds, some of the energy of the aircraft goes into compressing the air and locally changing the density of the air. This compressibility effect alters the amount of resulting force on the aircraft. The ratio of the speed of the aircraft to the speed of sound in the gas determines the magnitude of many of the compressibility effects. Because of the importance of this speed ratio, aerodynamicists have designated it with a special parameter called the Mach number in honor of Ernst Mach, a late 19th century physicist who studied gas dynamics. The Mach number M allows us to define flight regimes in which compressibility effects vary. Subsonic conditions occur for Mach numbers less than one. When the speed of the object approaches the speed of sound (the flight Mach number is nearly equal to one), the flow is said to be transonic. At flight speed exactly equal to the speed of sound (Mach number equal to one) the flow is said to be sonic. Supersonic conditions occur for Mach numbers greater than one. For speeds greater than five times the speed of sound, the flow is said to be hypersonic. Shocks appear in the flow for the transonic, supersonic, and hypersonic regimes. The Mach number could be computed using the following formula: $M = \frac{|V|}{\sqrt{1.4P/\rho}}$.

Another important feature is the angle of attack, which represents the direction of the flow in the referential in which the Navier–Stokes (or Euler) equations are solved. This is specified by the boundary conditions on the inlet through the velocity components.

Let us now give a short definition of the NACA0012 test case used in this paper.

A.2. NACA0012 definition. An airfoil is the cross section of a wing. The NACA0012 is an airfoil belonging to the family NACA airfoils designs invented by the NACA organization that became NASA (National Aeronautics and Space Administration). NACA0012 foil is a 12% foil shape, meaning the maximum cord depth (rudder thickness) is 12% of chord length (distance from the most forward point of the rudder to the most aft point). Figure 17 shows the NACA0012 geometry meshed with triangles.

FIG. 17. *A meshed NACA*0012 *geometry.*

REFERENCES

[1] M. AINSWORTH AND J. T. ODEN, *A posteriori error estimation in finite element analysis*, Comput. Methods Appl. Mech. Engrg., 142 (1997), pp. 1–88.

[2] I. BABUŠKA AND A. MILLER, *The post-processing approach in the finite element method—Part* 1: *Calculation of displacements, stresses and other higher order derivatives of the displacements*, Internat. J. Numer. Methods Engrg., 20 (1984), pp. 1085–1109.

[3] T. J. BAKER, *Mesh adaptation strategies for problems in fluid dynamics*, Finite Elem. Anal. Des., 25 (1997), pp. 243–273.

[4] R. Becker and R. Rannacher, *Weighted a posteriori error control in finite element methods*, in Proceedings of ENUMATH-97, World Scientific, Singapore, 1998, pp. 621–637.

[5] J. C. Carette, *Adaptive Unstructured Mesh Algorithms and SUPG Finite Element Method for Compressible High Reynolds Number Flows*, Ph.D. thesis, von Karman Institute for Fluid Dynamics, Rhode-Saint-Genese, Belgium, 1997.

[6] V. Dolejsi and J. Felcman, *Anisotropic mesh adaptation for transonic and supersonic flow simulation*, in Proceedings of the ALGORITMY 2002 Conference on Scientific Computing, pp. 78–85.

[7] W. G. Habashi, J. Dompierre, Y. Bourgault, D. Ait-Ali-Yahia, M. Fortin, and M. G. Vallet, *Anisotropic mesh adaptation: Towards user-independent, mesh-independent and solver-independent CFD solutions: Part* I: *General principles*, Internat. J. Numer. Methods Fluids, 32 (2000), pp. 725–744.

[8] D. Ait-Ali-Yahia, G. Baruzzi, W. G. Habashi, M. Fortin, J. Dompierre, and M. G. Vallet, *Anisotropic mesh adaptation: Towards user-independent, mesh-independent and solver-independent CFD solutions: Part* II: *Structured grids*, Internat. J. Numer. Methods Fluids, 39 (2002), pp. 657–674.

[9] C. J. Dompierre, M. G. Vallet, Y. Bourgault, M. Fortin, and W. G. Habashi, *Anisotropic mesh adaptation: Towards user-independent, mesh-independent and solver-independent CFD solutions: Part* III: *Unstructured meshes*, Internat. J. Numer. Methods Fluids, 39 (2002), pp. 675–702.

[10] J. Peraire, M. Vahdati, K. Morgan, and O. C. Zienkiewicz, *Adaptive remeshing for compressible flow computations*, J. Comput. Phys., 72 (1987), pp. 449–466.

[11] D. A. Venditti and D. L. Darmofal, *Adjoint error estimation and grid adaptation for functional outputs: Application to quasi-one-dimensional flow*, J. Comput. Phys., 164 (2000), pp. 204–227.

[12] G. P. Warren, W. K. Anderson, J. T. Thomas, and S. L. Krist, *Grid Convergence for Adaptive Methods*, Technical Paper 91-1592, AIAA Press, Washington, DC, 1991.

[13] O. C. Zienkiewicz and J. Z. Zhu, *A simple error estimation and adaptive procedure for practical engineering analysis*, Internat. J. Numer. Methods Engrg., 24 (1987), pp. 337–357.

[14] J. Peraire, J. Perió, and K. Morgan, *Adaptive remeshing for three-dimensional compressible flow computation*, J. Comput. Phys., 103 (1992), pp. 269–285.

[15] B. Mohammadi and O. Pironneau, *Mesh adaption and automatic differentiation in a CAD-free framework for optimal shape design*, Internat. J. Numer. Methods Fluids, 30 (1999), pp. 127–146.

[16] M. J. Castro-Diaz, H. Borouchaki, P. L. George, F. Hecht, and B. Mohammadi, *Anisotropic adaptive mesh generation in two dimensions for CFD*, Computational Fluid Dynamics, 96 (1996), pp. 181–192.

[17] H. Borouchaki, P. L. George, and B. Mohammadi, *Delaunay mesh generation governed by metric specications. Part* 2 : *Application examples*, Finite Elem. Anal. Des., 25 (1997), pp. 85–109.

[18] M. G. Larson and T. J. Barth, *A Posteriori Error Estimation for Discontinuous Galerkin Approximations of Hyperbolic Systems*, NAS Technical Report NAS-99-010, 1999.

[19] N. A. Pierce and M. B. Giles, *Adjoint recovery of superconvergent functionals from PDE approximations*, SIAM Rev., 42 (2000), pp. 247–264.

[20] E. F. D'azevedo and R. B. Simpson, *On optimal triangular meshes for minimizing the gradient error*, Numer. Math., 59 (1991), pp. 321–348.

[21] M. Fortin, *Anisotropic mesh adaptation through hierarchical error estimators*, in Advances in Computation: Theory and Practice, Nova Science Publishers, Commack, NY, 2001, pp. 53–65.

[22] N. Otsu, *A threshold selection method from gray-level histograms*, IEEE Transactions on Systems, Man and Cybernetics, 9 (1979), pp. 62–66.

[23] L. Remaki and M. Cheriet, *Numerical scheme of shock filter models for image enhancement and restoration*, J. Math. Imaging Vision, 18 (2003), pp. 129–143.

[24] L. Remaki and M. Cheriet, *Enhanced and restored signals as a generalized solution for shock filter models: Part* I: *Existence and uniqueness result of the Cauchy problem*, J. Math. Anal. Appl., 279 (2003), pp. 189–209.

[25] L. Remaki and M. Cheriet, *Enhanced and restored signals as a generalized solution for shock filter models: Part* II: *Numerical study*, J. Math. Anal. Appl., 279 (2003), pp. 398–417.

SIAM J. SCI. COMPUT.
Vol. 28, No. 4, pp. 1398–1419
© 2006 Society for Industrial and Applied Mathematics

A STABLE HIGH-ORDER METHOD FOR TWO-DIMENSIONAL BOUNDED-OBSTACLE SCATTERING*

DAVID P. NICHOLLS† AND JIE SHEN‡

Abstract. A stable and high-order method for solving the Helmholtz equation on a two-dimensional domain exterior to a bounded obstacle is developed in this paper. The method is based on a boundary perturbation technique ("transformed field expansions") coupled with a well-conditioned high-order spectral-Galerkin solver. The method is further enhanced with numerical analytic continuation, implemented via Padé approximation. Ample numerical results are presented to show the accuracy, stability, and versatility of the proposed method.

Key words. electromagnetic scattering, acoustic scattering, high-order methods, boundary perturbations, spectral-Galerkin methods

AMS subject classifications. 78A45, 65N35, 35J05, 41A58

DOI. 10.1137/050632920

1. Introduction.

A wide variety of numerical techniques have been proposed for the approximation of electromagnetic and acoustic scattering returns from irregular obstacles (see, e.g., the book of Colton and Kress [9] and the survey papers of Warnick and Chew [35] and Reitich and Tamma [28]). Among the most compelling of these methods are those based on boundary perturbations, which can be originally traced to the work of Rayleigh [27] and Rice [29]. These methods are simple to implement, extremely fast, and quite robust within their domain of applicability. These properties have lead to not only low- and high-order implementations of these boundary perturbation algorithms, but also questions regarding their convergence, conditioning, and accuracy.

In a series of recent papers, Bruno and Reitich [3], Nicholls and Nigam [20], and Nicholls and Reitich [24, 25] have rigorously shown that not only are the scattered (acoustic/electromagnetic) fields , , , functions of the grating height (slope) parameter, ε, but also they can be analytically continued for any real value of ε. These analytical results demonstrate the convergence of boundary perturbation expansions, and would appear to justify the classical algorithms of Rayleigh [27] and Rice [29] (pursued to higher orders by Bruno and Reitich [4, 5, 6, 7, 8]; see also Milder [14, 15, 17, 16] and others [18, 19, 12]). However, it was shown in [24, 25] that, in fact, these approaches rely on significant , , , , for their convergence. Consequently these numerical implementations are highly ill-conditioned when pursued to high order, as demonstrated in [24] and, to a more limited extent, section 3.4 of the current work.

A new boundary perturbation algorithm, advocated in [25] and specified in section 2.3, which overcomes this issue of poor conditioning, follows the perturbation philosophy of Rayleigh [27] and Rice [29] , , by a change of variables, which

*Received by the editors June 1, 2005; accepted for publication (in revised form) February 10, 2006; published electronically August 16, 2006.
http://www.siam.org/journals/sisc/28-4/63292.html

†Department of Mathematics, University of Notre Dame, Notre Dame, IN 46556 (Nicholls.2@nd.edu). The work of this author was supported by the NSF through grants DMS-0139822 and DMS-0406007.

‡Department of Mathematics, Purdue University, West Lafayette, IN 47907 (shen@math.purdue.edu). The work of this author was supported by the NSF through grants DMS-0243191 and DMS-0311915.

"flattens" the shape of the scatterer to a flat configuration. This "transformed field expansions" (TFE) method proves to be not only accurate and stable, but also robust at high order, and provides reliable estimates of both near- and far-field information.

The purpose of the current research is to extend the work of Nicholls and Reitich [24, 25] in several important directions. First, we present the TFE algorithm in the setting of bounded-obstacle scattering problems, which, while related to grating scattering problems, present their own difficulties. Such difficulties include, e.g., the different nature of solutions reflected in the appearance of Hankel functions rather than complex exponential functions in the fundamental solution, and the algorithmic challenges presented by the singular nature of the Laplacian in polar coordinates.

Second, and perhaps most importantly, we construct a new, highly efficient and accurate, well-conditioned spectral-Galerkin solver for the Helmholtz equation in an annulus (with a boundary condition involving the Dirichlet–Neumann operator (DNO)), which needs to be solved repeatedly in the TFE algorithm. This solver is highly efficient and accurate since it takes advantage of the fact that the DNO, although global in physical space, is local in frequency space. Hence, we can reduce the two-dimensional problem into a sequence of one-dimensional problems with a spectrally accurate approximation to the DNO. This solver is also well-conditioned since, as is done in a series of papers [30, 31, 32, 33], we construct basis functions which lead to sparse and well-conditioned linear systems. The good conditioning properties of this new method allow us to investigate the behavior of the TFE algorithm as the frequency of scattered radiation is varied from low to moderate, which is the final extension we present. Furthermore, based upon the preliminary investigations of [34] for radially symmetric scatterers, we can speculate that this new method will be quite effective in simulating problems with high frequencies (though it is not "specially designed" for this setting; see, e.g., [2] for a high-frequency boundary integral method). However, such calculations quickly become quite large and time-consuming, and we focus, in this first phase of this project, on modest calculations that illustrate the fundamental properties of the TFE algorithm in the bounded-obstacle configuration.

In section 2.1 we review the equations which govern the scattering of electromagnetic and acoustic radiation from bounded obstacles in two dimensions. In section 2.2 we review the classical "field expansions" (FE) method for numerically computing scattering returns, and in section 2.3 we generalize the stabilized TFE algorithm to bounded-obstacle configurations. In section 2.4 we discuss new algorithmic enhancements that improve the conditioning properties of the TFE method, and in section 3 we present numerical results that display the properties of both the classical FE and stabilized TFE approaches. In particular, in section 3.4 we exhibit the instabilities present in the FE method due to cancellations which are no longer present in the new, TFE recursions. In section 4 we examine the behavior of the TFE algorithm for computing plane-wave scattering problems in a variety of scenarios. After recalling how Padé approximation can be brought to bear on such problems in section 4.1, we conclude in section 4.2 with a series of numerical experiments that illuminate the capabilities of this new TFE algorithm. We present concluding remarks and future directions in section 5.

2. Boundary perturbation methods. In this study we will focus on the problem of approximating electromagnetic and acoustic plane-wave scattering returns from bounded obstacles. For simplicity we restrict our attention to the scalar case in two dimensions, and leave the more general three-dimensional setting for future work. In this section we state the governing equations and notations of bounded-obstacle

scattering and recall a classical boundary perturbation algorithm. To conclude we derive a new, stabilized method, and then discuss numerical implementation issues.

2.1. Governing equations. We consider a two-dimensional time-harmonic acoustic or electromagnetic plane wave,

$$\tilde{v}_i(r,\theta,t) = e^{i\omega t} v_i(r,\theta) = e^{i\omega t} e^{ir(\alpha\cos(\theta)-\beta\sin(\theta))},$$

which is incident upon a bounded obstacle

$$\Sigma = \{(r,\theta) : 0 \leq r < a + g(\theta), 0 \leq \theta < 2\pi\}$$

and generates a scattered field

$$\tilde{v}_s(r,\theta,t) = e^{i\omega t} v_s(r,\theta).$$

Here $\tilde{v} = \tilde{v}_i + \tilde{v}_s$ denotes the pressure in acoustics or the component of the electric or magnetic field parallel to the invariant (z) direction in electromagnetics. The reduced scattered field, v_s, obeys the Helmholtz equation

$$\Delta v_s + k^2 v_s = 0, \quad (r,\theta) \in \Omega, \tag{2.1}$$

where

$$\Omega = \{(r,\theta) : r > a + g(\theta), 0 \leq \theta < 2\pi\},$$

and the wavenumber k satisfies

$$k^2 = \alpha^2 + \beta^2 = (2\pi/\lambda)^2,$$

where λ is the wavelength of the radiation. Of course the scattered field must be periodic in θ, but there are several physically relevant boundary conditions that can be specified at the surface of the object. For definiteness we work with a pressure release (acoustics) or perfectly conducting (TE in electromagnetics) surface where

$$v_s(a+g(\theta),\theta) = -v_i(a+g(\theta),\theta) = -e^{i(a+g(\theta))(\alpha\cos(\theta)-\beta\sin(\theta))}. \tag{2.2}$$

Finally, the scattered solution must satisfy the Sommerfeld radiation condition (SRC) to guarantee a physical solution,

$$\lim_{r\to\infty} r^{1/2}(\partial_r v_s - ikv_s) = 0. \tag{2.3}$$

Gathering equations (2.1), (2.2), and (2.3); denoting v_s by v; and giving the Dirichlet data, (2.2), the generic name ξ, we state the fundamental equations of two-dimensional time-harmonic bounded-obstacle scattering:

$$\Delta v + k^2 v = 0, \qquad (r,\theta) \in \Omega, \tag{2.4a}$$

$$v(a+g(\theta),\theta) = \xi(\theta), \tag{2.4b}$$

$$\lim_{r\to\infty} r^{1/2}(\partial_r v - ikv) = 0. \tag{2.4c}$$

An equivalent statement of this problem, which not only conveniently expresses the SRC but also poses the problem on a [illegible] domain, can be made in terms of the DNO, also called the Dirichlet-to-Neumann (DtN) map. Let $b > a + |g|_{L^\infty}$; then,

using a classical argument of separation of variables, the general solution of (2.4a) and (2.4c) for $r \geq b$ can be expressed as

$$v(r,\theta) = \sum_{p=-\infty}^{\infty} a_p H_p^{(1)}(kr) e^{ip\theta},$$

where $H_p^{(1)}$ is the pth Hankel function of the first kind. Therefore, if $v(b,\theta)$ is given,

$$v(b,\theta) = \psi(\theta) = \sum_{p=-\infty}^{\infty} \hat{\psi}_p e^{ip\theta},$$

the coefficients $\{a_p\}$ can be uniquely determined; namely, we have

$$v(r,\theta) = \sum_{p=-\infty}^{\infty} \hat{\psi}_p \frac{H_p^{(1)}(kr)}{H_p^{(1)}(kb)} e^{ip\theta}.$$

We define a DNO, T, by

$$T\ \psi \equiv \partial_r v(b,\theta) = \sum_{p=-\infty}^{\infty} k \frac{d_z H_p^{(1)}(kb)}{H_p^{(1)}(kb)} \hat{\psi}_p e^{ip\theta}, \tag{2.5}$$

where

$$d_z H_p^{(1)}(kb) = \left. \frac{dH_p^{(1)}(z)}{dz} \right|_{z=kb},$$

which takes Dirichlet data, ψ, to Neumann data, $\partial_r v|_{r=b}$. Thus, (2.4) can be equivalently restated as

$$\Delta v + k^2 v = 0, \qquad (r,\theta) \in \Omega_{a+g,b}, \tag{2.6a}$$

$$v(a + g(\theta), \theta) = \xi(\theta), \tag{2.6b}$$

$$\partial_r v(b,\theta) - T\ v(b,\theta) = 0, \tag{2.6c}$$

where

$$\Omega_{a+g,b} = \{(r,\theta) : a + g(\theta) < r < b,\ 0 \leq \theta < 2\pi\}.$$

2.2. Field expansions. Considering a sufficiently smooth boundary deformation of the form $g(\theta) = \varepsilon f(\theta)$, it is known [3, 20, 24, 25] that the field, v, depends analytically on the parameter ε; i.e., we have the strongly convergent expansion

$$v(r,\theta;\varepsilon) = \sum_{n=0}^{\infty} v_n(r,\theta)\ \varepsilon^n.$$

This observation serves as the basis for the method of "field expansions" (FE); see [7, 8] and the references therein. This algorithm consists of finding $\{v_n\}$ recursively by solving a sequence of Helmholtz problems in the unperturbed domain

$$\Omega_{a,b} = \{(r,\theta) : a < r < b,\ 0 \leq \theta < 2\pi\}.$$

More precisely, it is not hard to derive from (2.6) that the v_n must satisfy

$$\Delta v_n + k^2 v_n = 0, \qquad (r,\theta) \in \Omega_{a,b}, \tag{2.7a}$$

$$v_n(a,\theta) = R_n(\theta), \tag{2.7b}$$

$$\partial_r v_n(b,\theta) - T\ v_n(b,\theta) = 0, \tag{2.7c}$$

where

$$R_n(\theta) = \delta_{n,0}\ \xi(\theta) - \sum_{l=0}^{n-1} \frac{f^{n-l}}{(n-l)!} \partial_r^{n-l} v_l(a,\theta) \tag{2.7d}$$

and $\delta_{n,m}$ is the Kronecker delta. Equations (2.7a) and (2.7c) imply that

$$v_n(r,\theta) = \sum_{p=-\infty}^{\infty} d_{n,p} \frac{H_p^{(1)}(kr)}{H_p^{(1)}(ka)} e^{ip\theta}, \tag{2.8}$$

while (2.7b) provides a recursion for the $d_{n,p}$. At order zero, $d_{0,p} = \hat{\xi}_p$, the pth Fourier coefficient of $\xi(\theta)$, and for $n > 0$

$$d_{n,p} = -\sum_{l=0}^{n-1} \sum_{q=-\infty}^{\infty} C_{n-l,p-q} \frac{d_z^{n-l} H_q^{(1)}(ka)}{H_q^{(1)}(ka)} d_{l,q}, \tag{2.9}$$

where the $C_{l,q}$ are defined by

$$\frac{(kf)^l}{l!} = \sum_{q=-\infty}^{\infty} C_{l,q} e^{iq\theta}.$$

Provided that we truncate the number of Fourier coefficients in f and retain only a finite number of terms $d_{l,p}$ (say, $0 \le l \le N$ and $-N_\theta/2 \le p \le N_\theta/2 - 1$), we have completely specified the [illegible] FE algorithm with (2.9). The only detail remaining is that the convolution product appearing in (2.9) is performed via FFT acceleration.

2.3. Transformed field expansions. As we shall see in section 3.4, the FE method outlined above suffers from severe ill-conditioning due to significant cancellations which occur in the recursion (2.9). However, a simple change of variables coupled to the FE design philosophy leads to a [illegible], high-order boundary perturbation scheme we call "transformed field expansions" (TFE) [21, 22, 23, 20, 24, 25]. The change of variables

$$r' = \frac{(b-a)r - bg(\theta)}{(b-a) - g(\theta)} = \frac{dr - bg}{d - g}, \qquad \theta' = \theta,$$

where $d = (b-a)$, maps the perturbed geometry $\Omega_{a+g,b}$ to the annulus $\Omega_{a,b}$. We now seek to restate (2.6) in these transformed coordinates. We note that

$$(d - g(\theta))\partial_\theta = (d - g(\theta'))\partial_{\theta'} - B(r',\theta')\partial_{r'}, \tag{2.10a}$$

$$(d - g(\theta))\partial_r = d\ \partial_{r'}, \tag{2.10b}$$

$$(d - g(\theta))D_r = d\ D_{r'} + A(r',\theta')\partial_{r'}, \tag{2.10c}$$

where $D_r = r\partial_r$,

$$A(r', \theta') = g(\theta')(b - r'),$$

and

$$B(r', \theta') = \partial_{\theta'} A = (\partial_{\theta'} g(\theta'))(b - r');$$

we note for future reference that

$$d\ r = d\ r' + A(r', \theta').$$

To begin the transformation we start with the Helmholtz equation, (2.6a):

$$\begin{aligned} 0 &= \Delta v + k^2 v = r^2 \Delta v + r^2 k^2 v \\ &= D_r^2 v + \partial_\theta^2 v + r^2 k^2 v \\ &= (d-g)^2 D_r^2 v + (d-g)^2 \partial_\theta^2 v + (d-g)^2 r^2 k^2 v \\ &= [(d-g)D_r][(d-g)D_r]v + [(d-g)\partial_\theta][(d-g)\partial_\theta]v \\ &\quad + (\partial_\theta g)[(d-g)\partial_\theta]v + (d-g)^2 r^2 k^2 v. \end{aligned} \tag{2.11}$$

Writing

$$u(r', \theta') = v(r' + A/d, \theta')$$

and replacing v in (2.11) by u, we obtain

$$\begin{aligned} 0 &= [dD_{r'} + A\partial_{r'}][dD_{r'} + A\partial_{r'}]u + [(d-g)\partial_{\theta'} - B\partial_{r'}][(d-g)\partial_{\theta'} - B\partial_{r'}]u \\ &\quad + (\partial_\theta g)[(d-g)\partial_{\theta'} - B\partial_{r'}]u + (d-g)^2 (r' + A/d)^2 k^2 u \\ &= d^2 D_{r'}^2 u + dA\partial_{r'} D_{r'} u + dD_{r'}[A\partial_{r'} u] + A\partial_{r'}[A\partial_{r'} u] \\ &\quad + d^2 \partial_{\theta'}^2 u - dg\partial_{\theta'}^2 u - d\partial_{\theta'}[g\partial_{\theta'} u] + g\partial_{\theta'}[g\partial_{\theta'} u] - dB\partial_{r'}\partial_{\theta'} u + B\partial_{r'}[g\partial_{\theta'} u] \\ &\quad - d\partial_{\theta'}[B\partial_{r'} u] + g\partial_{\theta'}[B\partial_{r'} u] + B\partial_{r'}[B\partial_{r'} u] \\ &\quad + d(\partial_{\theta'} g)\partial_{\theta'} u - g(\partial_{\theta'} g)\partial_{\theta'} u - (\partial_{\theta'} g)B\partial_{r'} u \\ &\quad + d^2 (r')^2 k^2 u + \sum_{j=1}^{4} C_j(g) k^2 u, \end{aligned}$$

where

$$\begin{aligned} C_1(g) &= -2d\ g(r')^2 + 2d\ Ar', \\ C_2(g) &= g^2 (r')^2 - 4gAr' + A^2, \\ C_3(g) &= (2/d)Ag^2 r' - (2/d)gA^2, \\ C_4(g) &= (1/d^2)g^2 A^2. \end{aligned}$$

We rewrite this as

$$D_{r'}^2 u + \partial_{\theta'}^2 u + (r')^2 k^2 u = F(r', \theta'; u),$$

where

$$\begin{aligned} -d^2 F &= dA\partial_{r'} D_{r'} u + dD_{r'}[A\partial_{r'} u] + A\partial_{r'}[A\partial_{r'} u] \\ &\quad - dg\partial_{\theta'}^2 u - d\partial_{\theta'}[g\partial_{\theta'} u] + g\partial_{\theta'}[g\partial_{\theta'} u] - dB\partial_{r'}\partial_{\theta'} u + B\partial_{r'}[g\partial_{\theta'} u] \\ &\quad - d\partial_{\theta'}[B\partial_{r'} u] + g\partial_{\theta'}[B\partial_{r'} u] + B\partial_{r'}[B\partial_{r'} u] \\ &\quad + d(\partial_{\theta'} g)\partial_{\theta'} u - g(\partial_{\theta'} g)\partial_{\theta'} u - (\partial_{\theta'} g)B\partial_{r'} u + \sum_{j=1}^{4} C_j(g) k^2 u. \end{aligned}$$

The Dirichlet condition, (2.6b), simply transforms to

$$u(a,\theta') = \xi(\theta'),$$

while (2.6c) can be written as

$$\begin{aligned} 0 &= \partial_r v(b,\theta) - Tv(b,\theta) \\ &= (d-g)\partial_r v(b,\theta) - (d-g)Tv(b,\theta) \\ &= d\ \partial_{r'} u(b,\theta') - d\ T'u(b,\theta') + g(\theta')T'u(b,\theta'). \end{aligned}$$

Since $T' = T$, (2.6c) transforms to

$$\partial_{r'} u(b,\theta') - Tu(b,\theta') = J(\theta'),$$

where

$$d\ J(\theta') = -g(\theta')Tu(b,\theta').$$

Collecting these transformations, we find that the transformed field u, upon dropping primes, satisfies

$$D_r^2 u + \partial_\theta^2 u + r^2 k^2 u = F(r,\theta;u), \qquad (r,\theta) \in \Omega_{a,b}, \tag{2.12a}$$

$$u(a,\theta) = \xi(\theta), \tag{2.12b}$$

$$\partial_r u(b,\theta) - T\ u(b,\theta) = J(\theta). \tag{2.12c}$$

Again, letting $g = \varepsilon f$, if f is sufficiently smooth, the analyticity of the [illegible] field u (see [20]) implies that

$$u(r,\theta;\varepsilon) = \sum_{n=0}^{\infty} u_n(r,\theta)\ \varepsilon^n. \tag{2.13}$$

Inserting the above into (2.12), it is straightforward, albeit tedious, to derive the following recursions for $\{u_n\}$:

$$D_r^2 u_n + \partial_\theta^2 u_n + r^2 k^2 u_n = F_n(r,\theta), \qquad (r,\theta)\ \in \Omega_{a,b}, \tag{2.14a}$$

$$u_n(a,\theta) = \delta_{n,0}\ \xi(\theta), \tag{2.14b}$$

$$\partial_r u_n(b,\theta) - T\ u_n(b,\theta) = J_n(\theta), \tag{2.14c}$$

where

$$\begin{aligned} -d^2 F_n = {} & d\ A\ \partial_{r'} D_{r'} u_{n-1} + d\ D_{r'}[A\partial_{r'} u_{n-1}] + A\partial_{r'}[A\partial_{r'} u_{n-2}] \\ & - d\ f\ \partial_{\theta'}^2 u_{n-1} - d\ \partial_{\theta'}[f\partial_{\theta'} u_{n-1}] + f\partial_{\theta'}[f\partial_{\theta'} u_{n-2}] \\ & - d\ B\partial_{r'}\partial_{\theta'} u_{n-1} + B\partial_{r'}[f\partial_{\theta'} u_{n-2}] \\ & - d\ \partial_{\theta'}[B\partial_{r'} u_{n-1}] + f\partial_{\theta'}[B\partial_{r'} u_{n-2}] + B\partial_{r'}[B\partial_{r'} u_{n-2}] \\ & + d(\partial_{\theta'} f)\partial_{\theta'} u_{n-1} - f(\partial_{\theta'} f)\partial_{\theta'} u_{n-2} - (\partial_{\theta'} f)B\partial_{r'} u_{n-2} \\ & + \sum_{j=1}^{4} C_j(f) k^2 u_{n-j} \end{aligned} \tag{2.14d}$$

and

$$d\ J_n = -f\ Tu_{n-1}(b,\theta). \tag{2.14e}$$

In the derivation of (2.14d) it is important to remember that A and B depend on g and therefore εf.

Remark 2.1. We note at this point the fundamentally inhomogeneous nature of (2.14). If we now wish to include effects from distributed sources, resulting in an inhomogeneous term—say, $M(r,\theta)$—appearing on the right-hand side of (2.1), the only modification to our TFE algorithm is that M will appear as F_0. By contrast, the FE recursions must be completely rederived, and even a boundary integral/element method will require a new implementation (though at the same computational cost).

2.4. TFE: Numerical implementation. It is clear from the above that the efficiency and accuracy of the TFE method depends greatly on how rapidly and accurately we can solve the Helmholtz equation (2.14a)–(2.14c) on an annular domain with the nonstandard outer boundary condition (expressed using a DNO). We now describe an efficient and accurate spectral-Galerkin method for the following problem (which is slightly more general than (2.14a)–(2.14c)):

$$D_r^2 U + \partial_\theta^2 U + r^2 k^2 U = F, \qquad (r,\theta) \in \Omega_0, \tag{2.15a}$$

$$U(a,\theta) = \xi(\theta), \tag{2.15b}$$

$$\partial_r U(b,\theta) - T\,U(b,\theta) = \eta(\theta), \tag{2.15c}$$

where $F(r,\theta)$, $\xi(\theta)$, and $\eta(\theta)$ are given functions. Expanding $U(r,\theta)$, $F(r,\theta)$, $\xi(\theta)$, and $\eta(\theta)$ in Fourier series, e.g.,

$$(U(r,\theta), F(r,\theta)) = \sum_{p=-\infty}^{\infty} (\hat{u}_p(r), \hat{f}_p(r)) e^{ip\theta}, \qquad (\xi(\theta), \eta(\theta)) = \sum_{p=-\infty}^{\infty} (\hat{\xi}_p, \hat{\eta}_p) e^{ip\theta},$$

and recalling the definition of the DNO in (2.5), we can decompose (2.15) into the following sequence of one-dimensional problems ($p = 0, \pm 1, \pm 2, \ldots$):

$$D_r^2 \hat{u}_p + (r^2 k^2 - p^2)\hat{u}_p = \hat{f}_p, \qquad r \in (a,b), \tag{2.16a}$$

$$\hat{u}_p(a) = \hat{\xi}_p, \tag{2.16b}$$

$$\partial_r \hat{u}_p(b) - k \frac{d_z H_p^{(1)}(kb)}{H_p^{(1)}(kb)} \hat{u}_p(b) = \hat{\eta}_p. \tag{2.16c}$$

It is known (see Lemma 3.2 in [10], and [11]) that

$$\operatorname{Im}\left\{ \frac{d_z H_p^{(1)}(kb)}{H_p^{(1)}(kb)} \right\} > 0, \qquad \operatorname{Re}\left\{ \frac{d_z H_p^{(1)}(kb)}{H_p^{(1)}(kb)} \right\} < 0,$$

$$-\frac{d_z H_p^{(1)}(kb)}{H_p^{(1)}(kb)} \sim \frac{p+1}{b} \quad \text{as } p \to \pm\infty, \tag{2.17}$$

which enables the well-posedness of the problem (2.16) to be directly established.

We now describe a spectral-Galerkin method for (2.16). To this end, let us first make a change of variable $x = \frac{2(r-a)}{b-a} - 1$, which maps $r \in (a,b)$ to $x \in I \equiv (-1,1)$. If we define

$$\tilde{u}_p(x) = \hat{u}_p(r), \quad \tilde{f}_p(x) = \hat{f}_p(r), \quad \tilde{\eta}_p = \frac{b-a}{2}\hat{\eta}_p, \quad \tilde{\xi}_p = \hat{\xi}_p,$$

$$c = \frac{b+a}{b-a}, \quad \omega = \frac{k(b-a)}{2}, \quad T_p = k\frac{d_z H_p^{(1)}(kb)}{H_p^{(1)}(kb)},$$

then (2.16) becomes

$$(x+c)\partial_x((x+c)\partial_x\tilde{u}_p),+((x+c)^2\omega^2-p^2)\tilde{u}_p=\tilde{f}_p,\qquad x\in I, \tag{2.18a}$$

$$\tilde{u}_p(-1)=\tilde{\xi}_p, \tag{2.18b}$$

$$\partial_x\tilde{u}_p(1)-T_p\tilde{u}_p(1)=\tilde{\eta}_p. \tag{2.18c}$$

One verifies easily that

$$h_p(x)=\frac{\tilde{\eta}_p+T_p\tilde{\xi}_p}{1-2T_p}x+\frac{\tilde{\eta}_p+\tilde{\xi}_p-T_p\tilde{\xi}_p}{1-2T_p}$$

satisfies the two boundary conditions in (2.18). Hence, setting

$$d=\frac{\tilde{\eta}_p+T_p\tilde{\xi}_p}{1-2T_p},\quad f_p=\tilde{f}_p-((x+c)^2\omega^2-p^2)h_p-d(x+c),$$
$$\tilde{u}_p(x)=u_p(x)+h_p(x),$$

we can rewrite (2.18) into the following problem with homogeneous boundary conditions:

$$(x+c)\partial_x((x+c)\partial_x u_p)+((x+c)^2\omega^2-p^2)u_p=f_p,\qquad x\in I, \tag{2.19a}$$

$$u_p(-1)=0, \tag{2.19b}$$

$$\partial_x u_p(1)-T_p u_p(1)=0. \tag{2.19c}$$

Let us denote by P_N the space of complex polynomials of degree less than or equal to N. We set

$$X_N^{(p)}=\{u\in P_N : u(-1)=0,\ \partial_x u(1)-T_p u(1)=0\}. \tag{2.20}$$

The spectral-Galerkin method for (2.19) is to find $u_N^{(p)}\in X_N^{(p)}$ such that

$$\int_I (x+c)\partial_x((x+c)\partial_x u_N^{(p)})\bar{v}_N\,dx\ +\int_I((x+c)^2\omega^2-p^2)u_N^{(p)}\bar{v}_N\,dx$$
$$=\int_I f_p\bar{v}_N\,dx\quad\forall v_N\in X_N^{(p)}, \tag{2.21}$$

where $\bar{v}_N$ is the complex conjugate of v_N.

The well-posedness of (2.21) follows from (2.16). However, a complete error analysis is still elusive. In [34], an error analysis was carried out for the case where T_p is replaced by its zeroth-order approximation, ip. The analysis was based on a new, uniform a priori estimate for the approximate solution. However, the extension of that procedure to the current situation appears to be subtle and is the subject of ongoing research.

We now describe an efficient numerical algorithm for solving (2.21). We recall that if T_p is a real number and P_N consists of real polynomials, it is shown in [30, 31] that there exist unique real pairs $(\alpha_k^{(p)},\beta_k^{(p)})$ such that

$$X_N^{(p)}=\text{span}\{\phi_0^{(p)},\phi_1^{(p)},\dots,\phi_{N-2}^{(p)}\}$$

with $\phi_k^{(p)}(x)=L_k(x)+\alpha_k^{(p)}L_{k+1}(x)+\beta_k^{(p)}L_{k+2}(x)$; we recall that L_k is the kth Legendre polynomial. It is easy to see that this is still true if we allow $(\alpha_k^{(p)},\beta_k^{(p)})$ to be complex

for complex T_p and P_N, $X_N^{(p)}$ as defined above. In fact, one verifies easily that

$$\alpha_k^{(p)} = \frac{(2k+3)}{(k+2)^2 - 2T_p}, \qquad \beta_k^{(p)} = \alpha_k^{(p)} - 1.$$

We recall from (2.17) that $\mathrm{Re}(T_p) < 0$; hence, $(\alpha_k^{(p)}, \beta_k^{(p)})$ are always well defined. Therefore, setting

$$\begin{aligned}
u_N^{(p)} &= \sum_{j=0}^{N-2} u_j^{(p)} \phi_j, \qquad \mathbf{u} = (u_0^{(p)}, u_1^{(p)}, \ldots, u_{N-2}^{(p)})^T, \\
a_{jl}^{(p)} &= \int_I (x+c)\partial_x((x+c)\partial_x \phi_l^{(p)})\bar{\phi}_j^{(p)}\, dx, \qquad A^{(p)} = (a_{jl}^{(p)}), \\
b_{jl}^{(p)} &= \int_I ((x+c)^2\omega^2 - p^2)\phi_l^{(p)}\bar{\phi}_j^{(p)}\, dx, \qquad B^{(p)} = (b_{jl}^{(p)}), \\
f_j^{(p)} &= \int_I f_p \bar{\phi}_j^{(p)}\, dx, \qquad \mathbf{f}^{(p)} = (f_0^{(p)}, f_1^{(p)}, \ldots, f_{N-2}^{(p)})^T,
\end{aligned}$$

the system (2.21) becomes the following complex matrix equation:

$$(A^{(p)} + B^{(p)})\mathbf{u}^{(p)} = \mathbf{f}^{(p)}. \tag{2.22}$$

Thanks to the special form of $\phi_j^{(p)}$, we have that

$$b_{jl}^{(p)} = 0, \quad |j-l| > 4; \qquad a_{jl}^{(p)} = 0, \quad l < j-2.$$

Hence, for each p, the linear system (2.22) can be solved in $\mathcal{O}(N^2)$ operations by using a direct Gaussian elimination process. Renaming N as N_r, the Fourier–Legendre approximation for (2.15) is given (with $x = \frac{2(r-a)}{b-a} - 1$) by

$$U_{N_r,N_\theta}(r,\theta) = \sum_{p=-N_\theta/2-1}^{N_\theta/2} \sum_{j=0}^{N_r-2} (u_j^{(p)}\phi_j^{(p)}(x) + h_p(x))e^{ip\theta}, \tag{2.23}$$

which is the solution of the following approximate variational problem:

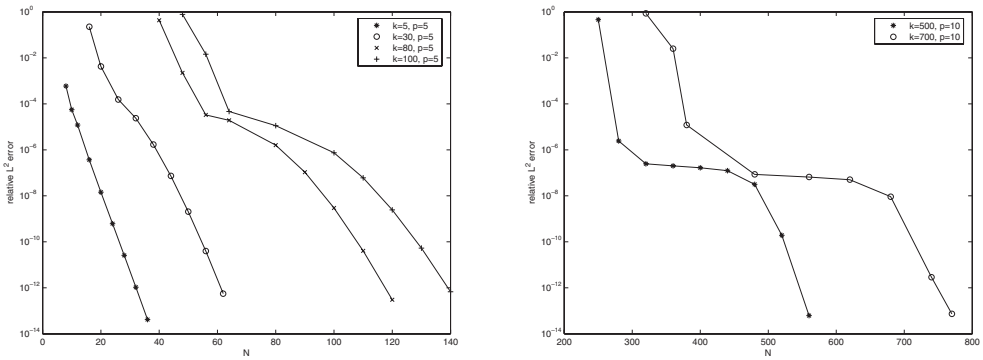
$$\begin{aligned}
&-(r\partial_r U_{N_r,N_\theta}, \partial_r(rv)) - (\partial_\theta U_{N_r,N_\theta}, \partial_\theta v) \\
&\quad + k^2(r^2 U_{N_r,N_\theta}, v) = (F, v) \quad \forall v \in V_{N_r,N_\theta}, \qquad \text{(2.24a)} \\
&U_{N_r,N_\theta}(a,\theta) = \xi_{N_\theta}(\theta), \qquad \text{(2.24b)} \\
&(\partial_r - T_{N_\theta})U_{N_r,N_\theta}(b,\theta) = \eta_{N_\theta}(\theta), \qquad \text{(2.24c)}
\end{aligned}$$

where

$$V_{N_r,N_\theta} = \mathrm{span}\{\phi_j^{(p)}(x(r))e^{ip\theta} : -N_\theta/2 \le p \le N_\theta/2 - 1;\ 0 \le j \le N_r - 2\}, \tag{2.25a}$$

$$(\xi_{N_\theta}(\theta), \eta_{N_\theta}(\theta)) = \sum_{p=-N_\theta/2}^{N_\theta/2-1} (\hat{\xi}_p, \hat{\eta}_p)e^{ip\theta}, \tag{2.25b}$$

$$T_{N_\theta}\psi(\theta) = \sum_{p=-N_\theta/2}^{N_\theta/2-1} k\frac{d_z H_p^{(1)}(kb)}{H_p^{(1)}(kb)}\hat{\psi}_p e^{ip\theta} \quad \text{for } \psi(\theta) = \sum_{p=-\infty}^{\infty} \hat{\psi}_p e^{ip\theta}. \tag{2.25c}$$



(a) Moderate Frequencies

(b) High Frequencies

FIG. 2.1. *Relative L^2 error versus N_r with $N_\theta = 20$, as compared to an exact solution.*

The total cost for obtaining U_{N_r,N_θ} is $\mathcal{O}(N_r^2 N_\theta \log(N_\theta))$.

We now present some numerical results demonstrating the accuracy of the above scheme. As a test example, we consider the canonical problem (2.15) with $a = 1$, $b = 2$ and take

$$F(r,\theta) = 0, \quad \xi(\theta) = H_p^{(1)}(ka)e^{ip\theta}, \quad \eta(\theta) = 0.$$

In this case the exact solution is $U(r,\theta) = H_p^{(1)}(kr)e^{ip\theta}$. It is clear that for given p, $e^{ip\theta}$ can be exactly determined with $N_\theta \geq 2p$. Thus we will concentrate on the approximation behavior of our scheme with respect to the frequency k. In Figure 2.1, we present the relative L^2 error versus N_r for a wide range of frequencies. We note that for k relatively small, the errors quickly enter the asymptotic range and converge at an exponential rate. For moderate to large k, the errors exhibit an initial sharp drop at $N_r \sim k/2$, then decay slowly until $N_r \sim k$, and finally enter the asymptotic range, converging at an exponential rate. These results indicate that (i) the approximate solution U_{N_r,N_θ} will converge to the exact solution $U(r,\theta)$ exponentially fast as $N_r, N_\theta \to +\infty$, provided that all $F(r,\theta)$, $\xi(\theta)$, and $\eta(\theta)$ are analytic in Ω; and (ii) our numerical scheme is stable for large N_r and capable of providing accurate results for large frequencies.

3. Numerical results: Comparison with an exact solution. In this section we present the results of a numerical experiment conducted with the two algorithms outlined above: the FE method of section 2.2 and the TFE method of section 2.3. We begin by outlining the diagnostic test (based upon an exact solution), which we have found useful in measuring the performance of these methods. We then describe how the FE algorithm suffers from severe numerical ill-conditioning, while the TFE approach is largely immune from such issues.

3.1. A diagnostic test. For the testing of numerical algorithms it is extremely useful to compare approximations with exact solutions whenever possible. In the case of scattering from a bounded obstacle, we consider again solutions of the form

$$v_p(r,\theta) = a_p H_p^{(1)}(kr)e^{ip\theta}, \tag{3.1}$$

which satisfy (2.6a) and (2.6c). Given a profile $g(\theta) = \varepsilon f(\theta)$, we have a solution to (2.6), provided that we set

$$\xi(\theta) = \xi_p(\theta) = v_p(a + \varepsilon f(\theta), \theta).$$

So, provided with (ε, f, ξ_p), both the FE and TFE algorithms deliver approximations to the field which can be compared with the exact field, v_p. Of course one need not compare the entire field and may choose, instead, to simply compare the field in the "near field" (at $r = a + \varepsilon f$) or the "far field" (at $r = b$). Additionally, derivatives of the field have physical interpretations and may be measured as well. In particular, the normal derivative of the field on a surface has the interpretation in electromagnetics as the current, and it is useful to measure this in both the near and far fields.

In this paper we focus on the most challenging computation mentioned above, the current at the surface of the obstacle; here the effects of the geometric perturbation are felt most strongly. We define this surface current as

$$\nu(\theta) = \nabla v|_{r=a+g} \cdot N_g,$$

where $N_g = ((a + g), -\partial_\theta g)^T$ is chosen unnormalized to match the definition of the DNO studied in Nicholls and Nigam [20] and Nicholls and Reitich [24, 25]. With this choice of normal it is easy to see that

$$\nu(\theta) = (a + g)\, \partial_r v|_{r=a+g} - \frac{(\partial_\theta g)}{(a + g)}\, \partial_\theta v|_{r=a+g} \tag{3.2}$$

so that, given (ε, f, ξ_p), an [illegible] current ν_p can be specified using (3.2) and compared with FE and TFE approximations.

3.2. Computation of the surface current: Field expansions. The FE approximation to the surface current can be generated by noting that (3.2), upon multiplication by $(a + g)$, can be rewritten as

$$\begin{aligned}\nu = a\, \partial_r v|_{r=a+g} + 2g\, \partial_r v|_{r=a+g} + \frac{1}{a} g^2\, \partial_r v|_{r=a+g} \\ - \frac{1}{a}(\partial_\theta g)\, \partial_\theta v|_{r=a+g} - \frac{1}{a} g\, \nu.\end{aligned}$$

Setting $g = \varepsilon f$, recalling our FE expansion, (2.8),

$$v(r, \theta, \varepsilon) = \sum_{n=0}^{\infty} \varepsilon^n \sum_{p=-\infty}^{\infty} d_{n,p} \frac{H_p^{(1)}(kr)}{H_p^{(1)}(ka)} e^{ip\theta},$$

and using the fact that $\nu(\theta, \varepsilon)$ is analytic in ε (see [20, 24, 25]),

$$\nu(\theta, \varepsilon) = \sum_{n=0}^{\infty} \nu_n(\theta)\, \varepsilon^n,$$

it is not difficult to see that

$$\begin{aligned}\nu_n = (ak)\sum_{l=0}^{n}\frac{(kf)^l}{l!}\sum_{p=-\infty}^{\infty}\frac{d_z^{l+1}H_p^{(1)}(ka)}{H_p^{(1)}(ka)}d_{n-l,p}e^{ip\theta}\\ +2f\sum_{l=0}^{n-1}\frac{(kf)^l}{l!}\sum_{p=-\infty}^{\infty}\frac{d_z^{l+1}H_p^{(1)}(ka)}{H_p^{(1)}(ka)}d_{n-1-l,p}e^{ip\theta}\\ +\frac{1}{a}f^2\sum_{l=0}^{n-2}\frac{(kf)^l}{l!}\sum_{p=-\infty}^{\infty}\frac{d_z^{l+1}H_p^{(1)}(ka)}{H_p^{(1)}(ka)}d_{n-2-l,p}e^{ip\theta}\\ -\frac{1}{a}(\partial_\theta f)\sum_{l=0}^{n-1}\frac{(kf)^l}{l!}\sum_{p=-\infty}^{\infty}\frac{d_z^{l}H_p^{(1)}(ka)}{H_p^{(1)}(ka)}(ip)d_{n-1-l,p}e^{ip\theta}\\ -\frac{1}{a}f\ \nu_{n-1}.\end{aligned}\tag{3.3}$$

3.3. Computation of the surface current: Transformed field expansions. To use the TFE algorithm in a simulation of the current ν, (3.2), we perform the following manipulations:

$$\begin{aligned}(a+g)(d-g)\nu &= (a+g)^2(d-g)\ \partial_r v|_{r=a+g} - (\partial_\theta g)(d-g)\ \partial_\theta v|_{r=a+g}\\ &= (a+g)^2 d\ \partial_{r'}u|_{r'=a} - (\partial_{\theta'}g)(d-g)\ \partial_{\theta'}v|_{r'=a}\\ &\quad +(\partial_{\theta'}g)^2 d\ \partial_{r'}v|_{r'=a}\,,\end{aligned}$$

where we have used (2.10) to change coordinates. Solving for $\nu(\theta')$, we find

$$\begin{aligned}\nu = a\ \partial_{r'}u + 2g\ \partial_{r'}u + \frac{1}{a}g^2\partial_{r'}u - \frac{1}{a}(\partial_{\theta'}g)\partial_{\theta'}u + \frac{1}{ad}g(\partial_{\theta'}g)\partial_{\theta'}u\\ +\frac{1}{a}(\partial_{\theta'}g)^2\partial_{r'}u + \frac{a-d}{ad}g\nu + \frac{1}{ad}g^2\nu,\end{aligned}\tag{3.4}$$

where all u quantities are evaluated at $r'=a$. Setting $g=\varepsilon f$, this normal current in the new coordinates can also be shown to be analytic in ε [20, 24, 25] so that

$$\nu(\theta',\varepsilon) = \sum_{n=0}^{\infty}\nu_n(\theta')\ \varepsilon^n,$$

and (3.4) can be used to show that

$$\begin{aligned}\nu_n = a\ \partial_{r'}u_n + 2f\partial_{r'}u_{n-1} + \frac{1}{a}f^2\partial_{r'}u_{n-2} - \frac{1}{a}(\partial_{\theta'}f)\partial_{\theta'}u_{n-1}\\ +\frac{1}{ad}f(\partial_{\theta'}f)\partial_{\theta'}u_{n-2} + \frac{1}{a}(\partial_{\theta'}f)^2\partial_{r'}u_{n-2} + \frac{a-d}{ad}f\nu_{n-1} + \frac{1}{ad}f^2\nu_{n-2},\end{aligned}\tag{3.5}$$

where terms with negative index are set to zero.

3.4. Cancellations. It is a well-established phenomenon [21, 22, 23, 24, 25] that boundary perturbation algorithms such as the FE approach of section 2.2 are numerically ill-conditioned, due to subtle but significant cancellations present in the recursions whereby finite quantities are computed as differences of nearly equal, but arbitrarily large, numbers. For a brief demonstration of this property in the present context we consider the diagnostic calculation outlined in section 3.1 for a specified profile to high perturbation order.

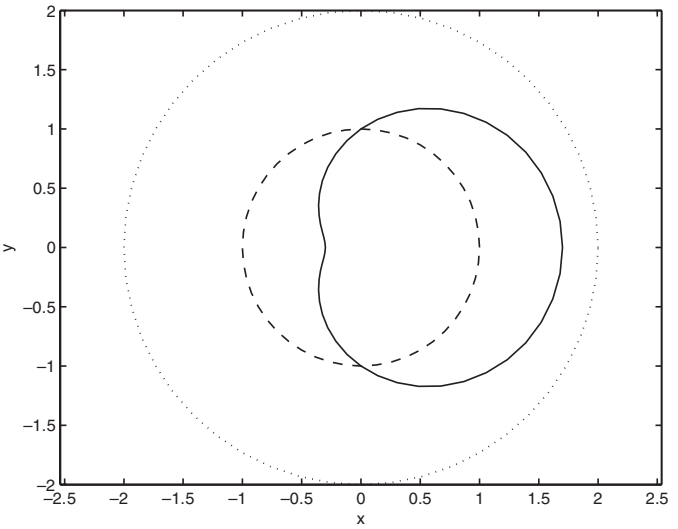


FIG. 3.1. *Geometry of a scattering configuration with smooth perturbation, $f(\theta) = \cos(\theta)$, $\varepsilon = 0.7$. The scattering profile is given by the solid line, and the artificial boundary at $r = 2$ is depicted with the dotted line. The dashed line portrays the unperturbed scatterer at $r = 1$.*

For an FE approximation we compute the $d_{n,p}$ of (2.8) using the recursion (2.9) for $-N_\theta/2 \le p \le N_\theta/2 - 1$ and $0 \le n \le N$. With these coefficients we approximate ν by

$$\nu^N(\theta, \varepsilon) = \sum_{n=0}^{N} \nu_n(\theta)\varepsilon^n, \tag{3.6}$$

where we use (3.3) to compute ν_n. For a TFE simulation we generate the $u_j^{(p)}$ (cf. (2.23)) from the algorithm outlined in section 2.4 for $-N_\theta/2 \le p \le N_\theta/2 - 1$, $0 \le j \le N_r - 2$, and $0 \le n \le N$. Again, with these coefficients we approximate ν by (3.6); however, we now use (3.5) to compute the ν_n.

In Figure 3.1 we display the geometry of a numerical experiment where we consider a large ($\varepsilon = 0.7$, $a = 1$) but smooth deformation, $f(\theta) = \cos(\theta)$, of a circle with artificial boundary at $r = b = 2$. The results of this experiment are summarized in Figure 3.2 with measurements of L^∞ error versus perturbation order N. The numerical parameters are $N_\theta = 48$, $N_r = 48$, and $N = 23$, and the frequency is $k = 1$, while the parameter p in (3.1) is set to one. We point out the abrupt and explosive divergence of the FE approximation from the exact solution beyond $N = 7$. In contrast, the TFE simulation gives progressively more accurate answers throughout all orders N. In particular, we note that the best answer that the FE method can ever deliver has L^∞ accuracy of 10^{-1} (when $N = 7$), while the TFE method gives answers reliable to 10^{-4} (at $N = 23$). As demonstrated in [22, 24], while the [illegible] features of Figure 3.2 are quite generic as a function of profile size (ε) and shape (f), the [illegible] details of the plot (e.g., divergence order of FE, slope of convergence for TFE, etc.) can be manipulated by an astute choice of ε and f.

Of course there are many questions one can ask regarding the cancellations present in the FE recursions. For instance, one can ask whether an "extended precision" implementation of the FE recursions could prevent the divergence witnessed in Figure 3.2, resulting in a reliable high-order method. In fact, this precise implementation was

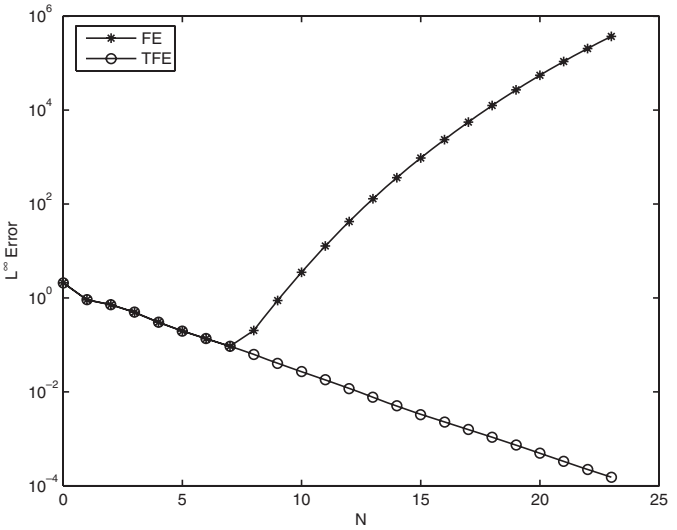


FIG. 3.2. *L^∞-norm of difference between exact current at the surface of a scatterer and the numerical approximation, versus perturbation order N. Both the FE (stars) and TFE (circles) approximations are plotted for the smooth perturbation $f(\theta) = \cos(\theta)$, $\varepsilon = 0.7$, with parameters $N_\theta = 48$, $N_r = 48$, $N = 23$, $a = 1$, $b = 2$, $k = 1$, $p = 1$. Notice the abrupt divergence of the FE approximation beyond $N = 7$ and the predictable, progressive convergence of the TFE approach throughout all orders.*

made for the FE recursions in the related context of traveling water waves in [26]. For this problem [illegible] precision calculations were necessary to display the strong cancellations present in this problem. From these experiments we learned that extended precision can [illegible] the onset of divergent behavior but will never eliminate it. Furthermore, with the computational cost of current software implementations of extended precision it was [illegible] computationally advantageous to use the FE recursions. We direct the interested reader to [21, 22, 23, 24, 25, 26] (particularly [24]) for a complete discussion of these issues and their resolution.

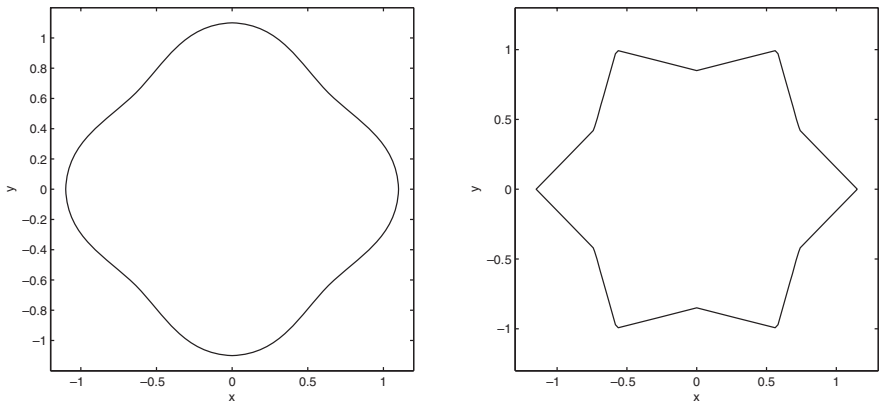
4. Numerical results: Plane-wave scattering. In this section we present the results of numerical experiments conducted with the TFE method of section 2.3 in a variety of physical configurations. At this point we abandon the convenient "diagnostic test" calculations described in section 3.1 and return to our original problem of plane-wave scattering. Of course in this scenario there are no exact solutions for nontrivial geometries, so we appeal to a high-order integral equation (IE) method (Maue's method; see [13, 9]) to provide an "exact solution." We first investigate the performance of the TFE method as the frequency of radiation is increased from low to moderate values, and then study its behavior as the size and smoothness of the obstacle perturbation are changed from small to large and from smooth to rough.

4.1. Padé approximation. Before we describe our plane-wave numerical experiments we comment on one final enhancement to our algorithm: analytic continuation via Padé approximation. Recall that the TFE method computes a spectral approximation to the current, ν, of the form (cf. (3.6))

$$\nu_{N_\theta}^N(\theta,\varepsilon) = \sum_{n=0}^{N} \sum_{p=-N_\theta/2}^{N_\theta/2-1} a_{n,p} e^{ip\theta} \varepsilon^n = \sum_{p=-N_\theta/2}^{N_\theta/2-1} S_p^N(\varepsilon) e^{ip\theta}$$

TABLE 4.1
Summary of numerical experiments.

Experiment number	Profile type	k	ε	q	N_{IE}
1	Sinusoidal	1	0.1	4	128
2	Sinusoidal	10	0.1	4	256
3	Sinusoidal	50	0.1	4	1024
4	Sinusoidal	1	0.3	4	256
5	Sinusoidal	1	0.1	16	1024
6	Star-like	1	N.A.	N.A.	N.A.



(a) Smooth (b) Star-like

FIG. 4.1. *Shape of smooth* (*cf.* (4.2) *with* $\varepsilon = 0.1$ *and* $q = 4$) *and star-like scatterers.*

for ε small, where $S_p^N(\varepsilon)$ is the truncated Taylor series of the pth Fourier coefficient $S_p(\varepsilon)$. The analytic continuation results of [23, 20] indicate that the functions $S_p(\varepsilon)$ can be extended beyond the disk of convergence of their power series about $\varepsilon = 0$; a numerical technique for this analytic continuation is the Padé approximation method [1]. We recall that the [P,Q] Padé approximant to a series

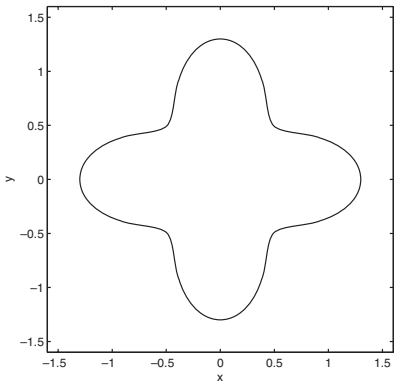
$$S(\varepsilon) = \sum_{n=0}^{\infty} a_n \varepsilon^n \tag{4.1}$$

is the unique rational function of order P over Q, which coincides with $S(\varepsilon)$ to order $P+Q+1$. In the experiments below we use diagonal or paradiagonal Padé sequences. It is well known that Padé approximants have some remarkable properties of approximation of (a large subclass of) analytic functions from their Taylor series (4.1) for points far outside their radii of convergence; see, e.g., [1]. They can be calculated by first solving a set of linear equations for the denominator coefficients, and then using simple formulas to compute the numerator coefficients.

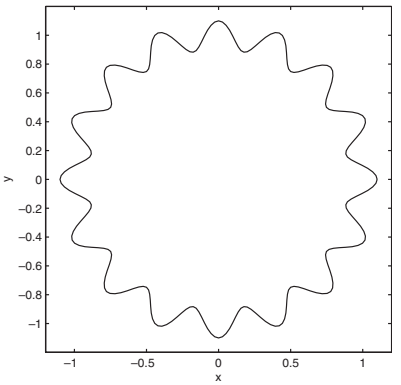
4.2. Numerical experiments. We now report upon six numerical experiments of plane-wave scattering, which are summarized in Table 4.1. In the first five the obstacle shape is quite smooth, in fact a sinusoidal perturbation of a circle, described by the equation

$$r = a + \varepsilon f_q(\theta), \qquad f_q(\theta) = \cos(q\theta); \tag{4.2}$$

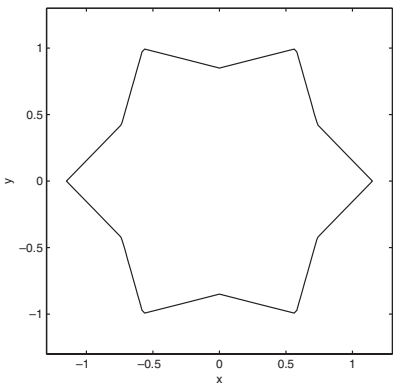
see Figure 4.1(a) ($q = 4$, $\varepsilon = 0.1$, and $a = 1$). The final experiment features a very rough, Lipschitz, scatterer of "star-like" shape; see Figure 4.1(b).



(a) Large



(b) Oscillatory



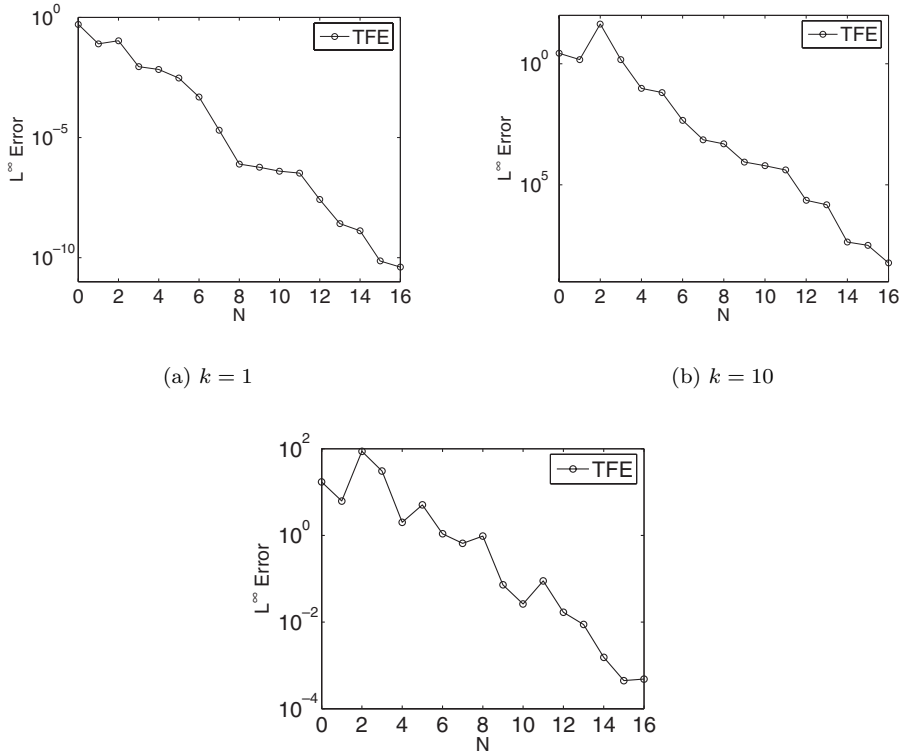
(c) Star-like

FIG. 4.2. *Shape of large* (*cf.* (4.2) *with* $\varepsilon = 0.3$ *and* $q = 4$)*, oscillatory* ((4.2) *with* $\varepsilon = 0.1$ *and* $q = 16$)*, and star-like scatterers.*

In the first three experiments we study the performance of the TFE method as the plane-wave frequency is increased from low to moderate. For this we use a small/smooth obstacle ($\varepsilon = 0.1$ and $q = 4$ in (4.2); see Figure 4.1(a) with $k = 1$, 10, and 50. In the final three experiments we gauge the capabilities of the TFE algorithm as the regularity of the obstacle is changed. We consider large/smooth ($\varepsilon = 0.3$ and $q = 4$), small/oscillatory ($\varepsilon = 0.1$ and $q = 16$), and Lipschitz (star-like) profiles for these experiments; see Figure 4.2.

In the final column of Table 4.1 we report the minimum number of discretization points necessary to guarantee a "well-resolved" IE approximation for each experiment. To measure the accuracy of the IE solution we compute the relative error in a "factor of two" refinement of an IE computation:

$$E_{\text{rel}}^N = \frac{\left|\nu_N^{IE} - \nu_{N/2}^{IE}\right|}{\left|\nu_N^{IE}\right|},$$



(a) $k = 1$

(b) $k = 10$

(c) $k = 50$

FIG. 4.3. *L^∞-norm of difference between highly resolved IE approximation of current at the surface of a scatterer and TFE approximation, versus perturbation order N. The shape of the scatterer is specified by* (4.2) *($\varepsilon = 0.1$ and $q = 4$), and the wavenumbers of the incident radiation are $(\alpha, \beta) = (1, 0), (10, 0)$, and $(50, 0)$, in subfigures* (a), (b), *and* (c), *respectively.*

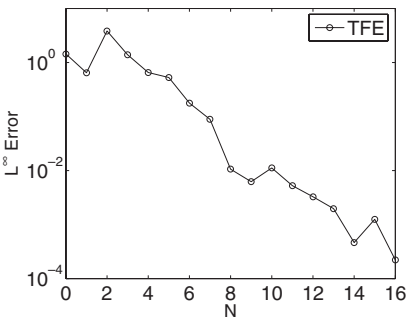
where ν_N^{IE} represents an IE approximation to the surface current with N discretization points. The N_{IE} listed in the final column of Table 4.1 is the smallest N such that

$$E_{\text{rel}}^N < 10^{-10}.$$

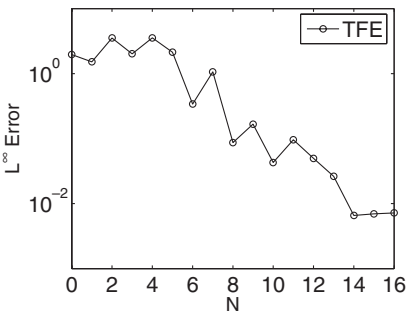
4.2.1. Variation of frequency. The results of our first set of experiments (as the frequency is varied from low to moderate) are summarized in Figure 4.3. The physical and numerical parameters for each experiment are collected in Table 4.2; note that, for uniformity, in the first experiment we actually use more discretization points than necessary in the IE solver. In each of these experiments we note that the TFE algorithm makes steady improvement (as compared to the IE exact solution) as the perturbation order is refined. In the cases $k = 1$, 10, and 50, accuracies of 10^{-10}, 10^{-9}, and 10^{-4}, respectively, are realized by $N = 16$; furthermore it is [illegible] to utilize these higher perturbation orders in order to realize this accuracy. In addition, a quite modest number of degrees of freedom are all that is required to properly resolve each configuration. Based upon the findings of [34] and the numerical results of section 2.4, we anticipate no problems simulating situations with k very large, as our new algorithm is numerically well-conditioned as [illegible] discretization parameters

TABLE 4.2
Summary of physical and numerical parameters: variation of frequency experiments ($q = 4$ *and* $\varepsilon = 0.1$).

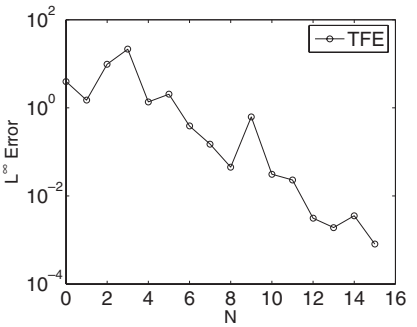
Experiment number	k	a	b	N_θ	N_r	N	$N(IE)$
1	1	1	2	96	24	16	256
2	10	1	2	128	32	16	256
3	50	1	1.2	320	50	16	1024



(a) Large



(b) Oscillatory



(c) Star-like

FIG. 4.4. (a) *and* (b) L^∞*-norm of difference between highly resolved IE approximation of current at the surface of a scatterer and TFE approximation, versus perturbation order* N. *The wavenumber of the incident radiation is* $(\alpha, \beta) = (1, 0)$. *The shapes of the scatterers are specified by* (4.2) ((a): $\varepsilon = 0.3$ *and* $q = 4$*;* (b): $\varepsilon = 0.1$ *and* $q = 16$). (c) *Cauchy convergence of TFE approximation measured in the* L^∞*-norm as a function of perturbation order* N. *The wavenumber of the incident radiation is* $(\alpha, \beta) = (1, 0)$. *The shape of the scatterer is depicted in Figure* 4.1(b).

(N_θ, N_r, and N) are refined. We note, however, that our method is not "specially designed" for the high-frequency regime in the manner of, e.g., [2].

4.2.2. Variation of shape. In the second set of experiments we recall that the frequency is fixed to be quite small ($k = 1$) while the profile shape is varied. For depictions of the scatterers for Experiments 4 (large/smooth profile), 5 (small/oscillatory profile), and 6 (Lipschitz profile) please see Figures 4.2(a), 4.2(b), and 4.2(c), respectively.

The results of these experiments are summarized in Figure 4.4. We note that for

TABLE 4.3
Summary of physical and numerical parameters: variation of shape experiments ($k = 1$).

Experiment number	ε	q	a	b	N_θ	N_r	N	$N(IE)$
4	0.3	4	1	2	256	48	16	256
5	0.1	16	1	2	256	48	16	1024
6	N.A.	N.A.	1	2	256	48	16	N.A.

the final simulation a different, "Cauchy," measure of convergence was utilized due to the Lipschitz character of the star-like profile. In this case the IE solver cannot be used without nontrivial modification, so in order to avoid this issue (essentially that of quickly and accurately resolving quadratures involving singular kernels), we focus upon the TFE algorithm and measure, again in L^∞, the difference between the TFE approximation at orders $0 \leq n < N$ and the "best resolution" TFE approximation at order N. The physical and numerical parameters for these experiments are collected in Table 4.3.

Once again we point out the consistent improvement which the TFE algorithm makes for all profiles at every perturbation order. For each profile we realize errors on the order of 10^{-2} by $N = 16$. This convergence is quite independent of the obstacle shape, although, of course, the rate of convergence does deteriorate with increasing roughness of the profile. Notice that this remains true even in the case of the Lipschitz profile, which we simply approximate by a truncated Fourier series. Again, each of these simulations could be further refined in a numerically stable fashion by increasing any of the algorithm parameters, N_θ, N_r, or N.

5. Conclusion and future directions. We have developed and implemented a stable, high-order method for numerical simulation of two-dimensional bounded-obstacle scattering. The method consists of five steps: (a) The problem (naturally posed on an unbounded domain) is reduced to one stated on a bounded domain via the Dirichlet–Neumann operator; (b) as suggested by previous theoretical work, this bounded domain is then mapped to an annulus; (c) a boundary perturbation expansion is then performed on the transformed equations, leading to a sequence of nonhomogeneous, constant-coefficient Helmholtz equations on the annulus; (d) an efficient and stable spectral-Galerkin method is developed for solving this sequence of Helmholtz problems; (e) Padé approximation is used to enhance and extend the summation of the power series representations we have formed. The resulting algorithm is shown to be very efficient and capable of handling small to moderate frequencies and scatterers with quite complex shapes.

The preliminary success of this TFE algorithm, of course, raises several natural questions which we consider as future directions for our research. It is clear that the basic ingredients of our numerical method can be extended to not only three-dimensional, bounded-obstacle configurations but also high-frequency problems in two and three dimensions. However, what is unclear is how this algorithm will compare with the popular finite element and integral equation methodologies already present in the literature. The highly accurate and efficient TFE algorithm together with its essentially exact treatment of far-field boundary conditions should make it a competitive choice compared with finite element approaches, particularly at moderate to high frequencies. On the other hand, its ability to seamlessly address problems with distributed sources will make this algorithm a compelling alternative to integral equation schemes. It is our intention to conduct an exhaustive study along these lines in future work.

Acknowledgments. The authors thank Fernando Reitich for providing a MATLAB script implementing the integral equations solver (Maue's method) for plane-wave scattering. They would also like to thank three anonymous referees for their thoughtful comments.

REFERENCES

[1] G. A. Baker, Jr., and P. Graves-Morris, *Padé approximants*, 2nd ed., Cambridge University Press, Cambridge, UK, 1996.

[2] O. P. Bruno, C. A. Geuzaine, J. A. Monro, Jr., and F. Reitich, *Prescribed error tolerances within fixed computational times for scattering problems of arbitrarily high frequency: The convex case*, Philos. Trans. R. Soc. Lond. Ser. A Math. Phys. Eng. Sci., 362 (2004), pp. 629–645.

[3] O. P. Bruno and F. Reitich, *Solution of a boundary value problem for the Helmholtz equation via variation of the boundary into the complex domain*, Proc. Roy. Soc. Edinburgh Sect. A, 122 (1992), pp. 317–340.

[4] O. P. Bruno and F. Reitich, *Numerical solution of diffraction problems: A method of variation of boundaries*, J. Opt. Soc. Amer. A, 10 (1993), pp. 1168–1175.

[5] O. P. Bruno and F. Reitich, *Numerical solution of diffraction problems: A method of variation of boundaries.* II. *Finitely conducting gratings, Padé approximants, and singularities*, J. Opt. Soc. Amer. A, 10 (1993), pp. 2307–2316.

[6] O. P. Bruno and F. Reitich, *Numerical solution of diffraction problems: A method of variation of boundaries.* III. *Doubly periodic gratings*, J. Opt. Soc. Amer. A, 10 (1993), pp. 2551–2562.

[7] O. P. Bruno and F. Reitich, *Calculation of electromagnetic scattering via boundary variations and analytic continuation*, Appl. Comput. Electromagn. Soc. J., 11 (1996), pp. 17–31.

[8] O. P. Bruno and F. Reitich, *Boundary-variation solutions for bounded-obstacle scattering problems in three dimensions*, J. Acoust. Soc. Amer., 104 (1998), pp. 2579–2583.

[9] D. Colton and R. Kress, *Inverse Acoustic and Electromagnetic Scattering Theory*, 2nd ed., Springer-Verlag, Berlin, 1998.

[10] L. Demkowicz and F. Ihlenburg, *Analysis of a coupled finite-infinite element method for exterior Helmholtz problems*, Numer. Math., 88 (2001), pp. 43–73.

[11] I. Harari and T. J. R. Hughes, *Analysis of continuous formulations underlying the computation of time-harmonic acoustics in exterior domains*, Comput. Methods Appl. Mech. Engrg., 97 (1992), pp. 103–124.

[12] P. J. Kaczkowski and E. I. Thorsos, *Application of the operator expansion method to scattering from one-dimensional moderately rough Dirichlet random surfaces*, J. Acoust. Soc. Amer., 96 (1994), pp. 957–972.

[13] A.-W. Maue, *Zur Formulierung eines allgemeinen Beugungsproblems durch eine Integralgleichung*, Z. Phys., 126 (1949), pp. 601–618.

[14] D. M. Milder, *An improved formalism for rough-surface scattering of acoustic and electromagnetic waves*, in Proceedings of SPIE - The International Society for Optical Engineering (San Diego, 1991), vol. 1558, SPIE, Bellingham, WA, 1991, pp. 213–221.

[15] D. M. Milder, *An improved formalism for wave scattering from rough surfaces*, J. Acoust. Soc. Amer., 89 (1991), pp. 529–541.

[16] D. M. Milder, *An improved formalism for electromagnetic scattering from a perfectly conducting rough surface*, Radio Sci., 31 (1996), pp. 1369–1376.

[17] D. M. Milder, *Role of the admittance operator in rough-surface scattering*, J. Acoust. Soc. Amer., 100 (1996), pp. 759–768.

[18] D. M. Milder and H. T. Sharp, *Efficient computation of rough surface scattering*, in Mathematical and Numerical Aspects of Wave Propagation Phenomena (Strasbourg, 1991), SIAM, Philadelphia, PA, 1991, pp. 314–322.

[19] D. M. Milder and H. T. Sharp, *An improved formalism for rough surface scattering.* II: *Numerical trials in three dimensions*, J. Acoust. Soc. Amer., 91 (1992), pp. 2620–2626.

[20] D. P. Nicholls and N. Nigam, *Exact non-reflecting boundary conditions on general domains*, J. Comput. Phys., 194 (2004), pp. 278–303.

[21] D. P. Nicholls and F. Reitich, *A new approach to analyticity of Dirichlet-Neumann operators*, Proc. Roy. Soc. Edinburgh Sect. A, 131 (2001), pp. 1411–1433.

[22] D. P. Nicholls and F. Reitich, *Stability of high-order perturbative methods for the computation of Dirichlet-Neumann operators*, J. Comput. Phys., 170 (2001), pp. 276–298.

[23] D. P. Nicholls and F. Reitich, *Analytic continuation of Dirichlet-Neumann operators*, Numer. Math., 94 (2003), pp. 107–146.
[24] D. P. Nicholls and F. Reitich, *Shape deformations in rough surface scattering: Cancellations, conditioning, and convergence*, J. Opt. Soc. Amer. A, 21 (2004), pp. 590–605.
[25] D. P. Nicholls and F. Reitich, *Shape deformations in rough surface scattering: Improved algorithms*, J. Opt. Soc. Amer. A, 21 (2004), pp. 606–621.
[26] D. P. Nicholls and F. Reitich, *On analyticity of traveling water waves*, Proc. Roy. Soc. London Ser. A, 461 (2005), pp. 1283–1309.
[27] Lord Rayleigh, *On the dynamical theory of gratings*, Proc. Roy. Soc. London Ser. A, 79 (1907), pp. 399–416.
[28] F. Reitich and K. Tamma, *State-of-the-art, trends, and directions in computational electromagnetics*, CMES Comput. Model. Eng. Sci., 5 (2004), pp. 287–294.
[29] S. O. Rice, *Reflection of electromagnetic waves from slightly rough surfaces*, Comm. Pure Appl. Math., 4 (1951), pp. 351–378.
[30] J. Shen, *Efficient spectral-Galerkin method.* I. *Direct solvers of second- and fourth-order equations using Legendre polynomials*, SIAM J. Sci. Comput., 15 (1994), pp. 1489–1505.
[31] J. Shen, *Efficient spectral-Galerkin method.* II. *Direct solvers of second- and fourth-order equations using Chebyshev polynomials*, SIAM J. Sci. Comput., 16 (1995), pp. 74–87.
[32] J. Shen, *Efficient spectral-Galerkin methods.* III. *Polar and cylindrical geometries*, SIAM J. Sci. Comput., 18 (1997), pp. 1583–1604.
[33] J. Shen, *Efficient spectral-Galerkin methods.* IV. *Spherical geometries*, SIAM J. Sci. Comput., 20 (1999), pp. 1438–1455.
[34] J. Shen and L.-L. Wang, *Spectral approximation of the Helmholtz equation with high wave numbers*, SIAM J. Numer. Anal., 43 (2005), pp. 623–644.
[35] K. F. Warnick and W. C. Chew, *Numerical simulation methods for rough surface scattering*, Waves Random Media, 11 (2001), pp. R1–R30.

SIAM J. SCI. COMPUT.
Vol. 28, No. 4, pp. 1420–1436
© 2006 Society for Industrial and Applied Mathematics

HIGH ORDER TERMS FOR CONDITION ESTIMATION OF UNIVARIATE POLYNOMIALS*

JOAB R. WINKLER†

Abstract. The established expressions for the condition numbers of a root of a polynomial assume that the perturbations in its coefficients are small, such that only the lowest order terms in a Taylor series need be considered. This restriction to the lowest order terms is considered in this paper, and it is shown that there exists a well-defined class of polynomial for which it is not valid. In particular, if a root x_0 of a polynomial $p(x)$ is close to but distinct from its neighboring roots $x_i, i = 1, \ldots, r$, it is shown that it is necessary to consider higher order terms in the Taylor series. These additional terms imply that a closed form expression for the condition number of x_0 cannot be obtained, and numerical methods must therefore be employed. In this paper, a high order Taylor series is used to compute the region in the complex plane in which the perturbed root $x_0 + \delta x_0$ lies, and this enables the variation of the condition number of x_0 with the distances to its neighboring roots to be established. It is shown that this extra information, which cannot be obtained if only the lowest order terms are used, is significant.

Key words. polynomials, condition numbers

AMS subject classification. 65F35

DOI. 10.1137/050629239

1. Introduction.

There has been considerable research into the numerical condition of the roots of a polynomial, and significant progress has been made [1, 2, 3, 6, 7, 8]. Closed form expressions for worst case and average case measures of the condition numbers of a root of a polynomial in an arbitrary basis have been developed [3, 10, 12, 13], and excellent agreement between the computed and theoretical results has been obtained. The derivation of these expressions requires that the perturbations in the coefficients of the polynomial are small, such that only the lowest order terms in the Taylor series of the perturbed polynomial need be considered. In this paper, the restrictions that are imposed by this assumption are considered, and it is shown that there exists a well-defined class of polynomial for which it is invalid. In particular, higher order terms in the Taylor series of the perturbed polynomial must be considered if the root x_0, whose condition number is sought, is near its neighboring roots. Numerical methods must, however, be used because closed form expressions for the numerical condition of x_0 cannot be obtained in this circumstance.

The derivation of an expression for the componentwise condition number of a root of a polynomial is reviewed in section 2, and examples of low degree polynomials are included in order to show the restrictions that must be imposed on this expression when the roots of a polynomial are close. These restrictions are removed by the inclusion of higher order terms in the Taylor series of the perturbed polynomial, and this is considered in section 3. It is shown that it is instructive to consider the region in the complex plane in which a perturbed root lies because this geometric information explains the numerical answers.

Section 4 contains the results of computational experiments that show that the

*Received by the editors April 15, 2005; accepted for publication (in revised form) March 6, 2006; published electronically August 16, 2006.
http://www.siam.org/journals/sisc/28-4/62923.html
†Department of Computer Science, The University of Sheffield, Regent Court, 211 Portobello Street, Sheffield S1 4DP, UK (j.winkler@dcs.shef.ac.uk).

roots of a perturbed polynomial lie along well-defined trajectories, or principal axes, and that calculations that are based on these results may differ significantly from results that are based on the condition number. Section 5 contains a summary of the work that is described in this paper and future work that follows from it.

The work that is described in this paper is related to the pseudozeros of polynomials [4, 5]. In particular, the pseudozero set of a polynomial $p(x)$ is the set of zeros of the polynomials that are obtained when the coefficients of $p(x)$ are perturbed, and it quantifies the numerical condition of the roots of $p(x)$. It enables the numerical condition of $p(x)$ to be considered geometrically because the pseudozero set is plotted as a set of contours in the complex plane, where each contour represents a perturbation of different magnitude in the coefficients. The condition numbers in [1, 2, 3] are used in these papers on pseudozeros, and thus the work that is described in this paper extends the use of pseudozeros to a polynomial for which high order terms must be included in the Taylor series of its perturbed form.

2. The condition number of a root of a polynomial. This section contains a proof of the expression for the componentwise condition number of a root of a polynomial. Although this proof has been established [3], its inclusion in this paper is required because reference will be made to particular lines in the proof when the assumptions required for its establishment are considered. Examples are given in order to show the situations in which this formula is or is not valid.

2.1. An expression for the condition number. Consider a root x_0 of a polynomial $p(x)$, which is of degree n and expressed in an arbitrary basis $\{\phi_i\}_{i=0}^{n}$ with coefficients $a_i,\ i = 0, \ldots, n$,

$$p(x) = \sum_{i=0}^{n} a_i \phi_i(x). \tag{2.1}$$

Each coefficient a_i is perturbed to $a_i + \delta a_i = a_i(1 + r_i \varepsilon_c)$, where r_i is a random variable that is uniformly distributed in the interval $[-1, +1]$ and ε_c is a constant, and thus

$$|\delta a_i| \leq \varepsilon_c |a_i|, \qquad i = 0, \ldots, n. \tag{2.2}$$

THEOREM 2.1. [illegible] a_i, $p(x)$ [illegible] $a_i + \delta a_i$ [illegible] [illegible] x_0, $p(x)$ [illegible] m [illegible] m [illegible] $x_0 + \delta x_0$ [illegible] [illegible] x_0

$$\kappa_c(x_0) = \max_{|\delta a_i| \leq \varepsilon_c |a_i|} \frac{|\delta x_0|}{|x_0|} \frac{1}{\varepsilon_c} = \frac{1}{\varepsilon_c^{1-\frac{1}{m}}} \frac{1}{|x_0|} \left(\frac{m!}{\left|p^{(m)}(x_0)\right|} \sum_{i=0}^{n} |a_i \phi_i(x_0)| \right)^{\frac{1}{m}}. \tag{2.3}$$

[illegible]. Consider the perturbed polynomial $p(x + \delta x)$,

$$\begin{aligned} p(x + \delta x) &= \sum_{i=0}^{n} a_i \phi_i(x + \delta x) \\ &= \sum_{i=0}^{n} (a_i + \delta a_i) \phi_i(x + \delta x) - \sum_{i=0}^{n} \delta a_i \phi_i(x + \delta x). \end{aligned}$$

By definition, $p(x_0) = 0$, and it is assumed that the change δa in the coefficients a causes a change δx_0 in the root x_0; that is,

$$\sum_{i=0}^{n} (a_i + \delta a_i)\, \phi_i (x_0 + \delta x_0) = 0,$$

from which it follows that

$$p(x_0 + \delta x_0) = -\sum_{i=0}^{n} \delta a_i \phi_i (x_0 + \delta x_0).$$

By Taylor's theorem,

$$p(x_0 + \delta x_0) = \sum_{k=0}^{n} \frac{\delta x_0^k}{k!} p^{(k)}(x_0)$$

and

$$\phi_i(x_0 + \delta x_0) = \sum_{k=0}^{n} \frac{\delta x_0^k}{k!} \phi_i^{(k)}(x_0),$$

and hence

$$\sum_{k=0}^{n} \frac{\delta x_0^k}{k!} p^{(k)}(x_0) = -\sum_{i=0}^{n} \delta a_i \sum_{k=0}^{n} \frac{\delta x_0^k}{k!} \phi_i^{(k)}(x_0). \tag{2.4}$$

Since x_0 is an m-tuple root, $p^{(k)}(x_0) = 0$ for $0 \le k \le m-1$, and the perturbation δx_0 is assumed to be small, it follows that

$$\delta x_0^m = -\frac{m!}{p^{(m)}(x_0)} \sum_{i=0}^{n} \delta a_i \sum_{k=0}^{n} \frac{\delta x_0^k}{k!} \phi_i^{(k)}(x_0), \tag{2.5}$$

and hence

$$\begin{aligned} |\delta x_0| &= \left| \frac{m!}{p^{(m)}(x_0)} \right|^{\frac{1}{m}} \left| \sum_{i=0}^{n} \delta a_i \sum_{k=0}^{n} \frac{\delta x_0^k}{k!} \phi_i^{(k)}(x_0) \right|^{\frac{1}{m}} \\ &\le \left(\frac{m!}{\left|p^{(m)}(x_0)\right|} \sum_{i=0}^{n} |\delta a_i| \sum_{k=0}^{n} \left| \frac{\delta x_0^k}{k!} \phi_i^{(k)}(x_0) \right| \right)^{\frac{1}{m}} \\ &\le \varepsilon_c^{\frac{1}{m}} \left(\frac{m!}{\left|p^{(m)}(x_0)\right|} \sum_{i=0}^{n} |a_i \phi_i(x_0)| \right)^{\frac{1}{m}}, \end{aligned} \tag{2.6}$$

because only the term corresponding to $k = 0$ is retained. Thus

$$\frac{|\delta x_0|}{|x_0|} \frac{1}{\varepsilon_c} \le \frac{1}{\varepsilon_c^{1-\frac{1}{m}}} \frac{1}{|x_0|} \left(\frac{m!}{\left|p^{(m)}(x_0)\right|} \sum_{i=0}^{n} |a_i \phi_i(x_0)| \right)^{\frac{1}{m}},$$

and the result (2.3) is established. □

Examination of this proof shows that only the term of order m is considered on the left-hand side, and only the term corresponding to $k = 0$ is considered on the right-hand side, of (2.4). The restrictions that are implied by these lowest order approximations must be considered in order to determine the class of polynomial for which (2.3) is valid. In particular, it follows from (2.4) that the change δx_0 in x_0 must satisfy

$$\sum_{j=m+1}^{n} \frac{\left|\delta x_0^j p^{(j)}(x_0)\right|}{j!} \ll \frac{\left|\delta x_0^m p^{(m)}(x_0)\right|}{m!} \tag{2.7}$$

and

$$\sum_{k=1}^{n} \left|\frac{\delta x_0^k}{k!}\phi_i^{(k)}(x_0)\right| \ll |\phi_i(x_0)|, \qquad i = 0, \ldots, n, \tag{2.8}$$

in order that (2.3) be valid. The next section contains an example that shows the restrictions that the conditions (2.7) and (2.8) impose on the condition number (2.3).

2.2. The restrictions of lowest order terms on condition estimation. Consider the polynomial

$$p(x) = (x-1)^2(x-1-\alpha) = x^3 - (3+\alpha)x^2 + (3+2\alpha)x - (1+\alpha), \tag{2.9}$$

for which, from (2.1),

$$\begin{array}{llll} a_0 = 1, & a_1 = -(3+\alpha), & a_2 = 3+2\alpha, & a_3 = -(1+\alpha), \\ \phi_0(x) = x^3, & \phi_1(x) = x^2, & \phi_2(x) = x, & \phi_3(x) = 1, \end{array} \tag{2.10}$$

and $\alpha \geq 0$. It is readily verified that the componentwise condition numbers of the double root $x_0 = 1$, the simple root $x_1 = (1+\alpha)$ (assuming the specific condition $\alpha > 0$), and the treble root $x_2 = 1$, corresponding to the condition $\alpha = 0$, are

$$\kappa_c(x_0) = \frac{2}{\sqrt{\varepsilon_c}}\left(\frac{2+\alpha}{\alpha}\right)^{\frac{1}{2}}, \qquad \kappa_c(x_1) = 2\left(\frac{2+\alpha}{\alpha}\right)^2, \tag{2.11}$$

and

$$\kappa_c(x_2) = \frac{2}{\varepsilon_c^{\frac{2}{3}}}, \tag{2.12}$$

respectively.

It follows from (2.7) that the perturbation δx_0 in the double root x_0 must satisfy $|\delta x_0| \ll \alpha$, and if it is assumed, for simplicity, that only the term defined by $k = 1$ on the left-hand side of (2.8) need be considered because $|\delta x_0| \ll 1$, from which it follows that the terms corresponding to the indices $k = 2$ and $k = 3$ are small, then this equation reduces to

$$\left|\delta x_0 \phi_i^{(1)}(x_0)\right| \ll |\phi_i(x_0)|, \qquad i = 0, \ldots, 3,$$

which yields

$$|\delta x_0| \ll \frac{1}{3}.$$

TABLE 2.1
The maximum value of ε_c for four values of the root separation α for the double root $x_0 = 1$ and the simple root $x_1 = (1 + \alpha)$.

α	$x_0 = 1$ max ε_c	$x_1 = (1+\alpha)$ max ε_c
1	9.3×10^{-5}	1.4×10^{-3}
0.1	1.2×10^{-6}	5.2×10^{-6}
0.01	1.2×10^{-9}	6.1×10^{-9}
0.001	1.2×10^{-12}	6.2×10^{-12}

If it is assumed that a factor of 10 is sufficient to replace the $\ll$ by $\leq$, then these strict inequalities can be written as

$$|\delta x_0| \leq \frac{\alpha}{10} \qquad \text{and} \qquad |\delta x_0| \leq \frac{1}{30},$$

and their satisfaction is required in order that the condition number (2.3) be valid. Also, it follows from (2.6) that the maximum value of the perturbation δx_0 is equal to

$$|\delta x_0|_{\max} = 2\varepsilon_c^{\frac{1}{2}} \left(\frac{2+\alpha}{\alpha} \right)^{\frac{1}{2}},$$

and thus the root separation α must satisfy

$$|\delta x_0|_{\max} = 2\varepsilon_c^{\frac{1}{2}} \left(\frac{2+\alpha}{\alpha} \right)^{\frac{1}{2}} \leq \min \left\{ \frac{\alpha}{10}, \ \frac{1}{30} \right\}, \tag{2.13}$$

in order that the condition number $\kappa_c(x_0)$ in (2.11) be valid.

This analysis can be repeated for the simple root $x_1 = (1 + \alpha)$, in which case (2.13) becomes

$$|\delta x_1|_{\max} = 2\varepsilon_c(1+\alpha) \left(\frac{2+\alpha}{\alpha} \right)^{2} \leq \min \left\{ \frac{1+\alpha}{30}, \ \frac{\alpha}{20} \right\}. \tag{2.14}$$

The maximum allowable value of the componentwise relative error ε_c for four values of the root separation α, computed from (2.13) and (2.14), is shown in Table 2.1, and it is seen that as α decreases, the maximum value of ε_c decreases. It is noted that if α is sufficiently small, the restriction on the value of ε_c may be too severe for practical applications, such that the condition number (2.3) is not a realistic measure of the numerical stability of the double root $x_0 = 1$ and the simple root $x_1 = (1 + \alpha)$.

It follows from (2.13) and (2.14), respectively, that

$$|\delta x_0|_{\max} = \begin{cases} \frac{\alpha}{10} & \text{if } 0 < \alpha \leq \frac{1}{3}, \\ \\ \frac{1}{30} & \text{if } \frac{1}{3} < \alpha, \end{cases}$$

and

$$|\delta x_1|_{\max} = \begin{cases} \frac{\alpha}{20} & \text{if } 0 < \alpha \leq 2, \\ \\ \frac{1+\alpha}{30} & \text{if } 2 < \alpha, \end{cases}$$

and hence

$$|\delta x_0|_{\max} + |\delta x_1|_{\max} = \begin{cases} \frac{3\alpha}{20} & \text{if } 0 < \alpha \leq \frac{1}{3}, \\ \frac{\alpha}{20} + \frac{1}{30} & \text{if } \frac{1}{3} < \alpha \leq 2, \\ \frac{1+\alpha}{30} + \frac{1}{30} & \text{if } 2 < \alpha. \end{cases}$$

It follows from this equation that

$$|\delta x_0|_{\max} + |\delta x_1|_{\max} < \alpha,$$

which implies that if only the lowest order terms are used in the Taylor series of the perturbed polynomial, the circular regions in which the perturbed roots $x_0 + \delta x_0$ and $x_1 + \delta x_1$ lie cannot intersect because α is the root separation. This is problematic because one would expect that these regions would merge for sufficiently small values of α and, moreover, form one region when $\alpha = 0$, corresponding to the perturbation region of the treble root $x_2 = 1$. The failure of the perturbed roots to satisfy this requirement is a direct consequence of the limitations that are imposed when only lowest order terms are used. It will be shown, however, in section 3.1, and particularly in Figures 3.1 and 3.2, that this problem is not present when higher order terms are included in the Taylor series of the perturbed polynomial.

More examples that show the restrictions on the condition number (2.3), and its relation to the backward error of a root of a polynomial, are in section 4 in [9]. These examples, and the example above, show that higher order terms must be included in a Taylor series when it is required to compute the condition number of a root that is near another root.

3. High order terms for condition estimation. The inclusion of high order terms in the Taylor series of a perturbed polynomial implies that a closed form expression for the condition number of a root cannot be obtained, and numerical methods must be employed. The theoretical derivation for this situation proceeds up to (2.4), but all the terms are retained on both the left- and right-hand sides of this equation. In order to establish the method that will be used, consider the polynomial $f(x)$ of degree n,

$$f(x) = \prod_{i=0}^{r} (x - x_i)^{m_i}, \qquad \sum_{i=0}^{r} m_i = n,$$

whose distinct roots are $x_i,\ i = 0, \ldots, r$, and the multiplicity of the root x_k is m_k. The computation of the condition number of the root x_0 proceeds as follows:

1. Expand both sides of (2.4) such that all terms are retained, and consider the modulus of both sides of this equation.
2. Apply the componentwise error model (2.2) to the right-hand side of (2.4), thereby obtaining an inequality for the perturbation δx_0, which may be complex, in terms of the coefficients a_i and the componentwise relative error ε_c. The values of δx_0 that satisfy this inequality define the region R in the complex plane in which the perturbed root $x_0 + \delta x_0$ lies.
3. The point on the boundary of R that is furthest away from the point in the complex plane that represents the root x_0 defines the condition number of x_0.

It is important to note that the restrictions that are imposed when only the lowest order terms are used, as shown in section 2.2 and section 4 in [9], do not arise when this method is used for condition estimation because all terms in the series on both sides of (2.4) are now used.

The next section shows the implementation of this method for the third order polynomial (2.9).

3.1. An example of condition estimation using high order Taylor series. Consider the polynomial (2.9) whose roots are $x_0 = 1$ (twice) and $x_1 = (1 + \alpha) = (x_0 + \alpha)$ (once), and it is recalled that $\alpha \geq 0$. Since

$$p^{(2)}(x_0) = -2\alpha \qquad \text{and} \qquad p^{(3)}(x_0) = 6,$$

it follows that

$$p(x_0 + \delta x_0) = -\alpha \delta x_0^2 + \delta x_0^3.$$

This term is used on the left-hand side of (2.4), and the completeness of the Taylor expansion is noted. Equation (2.4) therefore becomes

$$\begin{aligned} \left|\delta x_0^3 - \alpha \delta x_0^2\right| &= \left| \sum_{i=0}^{3} \delta a_i \sum_{k=0}^{3} \frac{\delta x_0^k}{k!} \phi_i^{(k)}(x_0) \right| \\ &\leq \sum_{i=0}^{3} |\delta a_i| \left| \sum_{k=0}^{3} \frac{\delta x_0^k}{k!} \phi_i^{(k)}(x_0) \right| \\ &\leq \varepsilon_c \sum_{i=0}^{3} |a_i| \left| \sum_{k=0}^{3} \frac{\delta x_0^k}{k!} \phi_i^{(k)}(x_0) \right|, \end{aligned}$$

and (2.10) enables the right-hand side of this equation to be simplified,

$$\begin{aligned} \left|\delta x_0^3 - \alpha \delta x_0^2\right| \leq \varepsilon_c \Big\{ &\left|(1 + \delta x_0)^3\right| + (3 + \alpha) \left|(1 + \delta x_0)^2\right| \\ &+ (3 + 2\alpha) |1 + \delta x_0| + (1 + \alpha) \Big\}. \end{aligned}$$

The values of δx_0 that satisfy this inequality define the region R in the complex plane in which the perturbed root $x_0 + \delta x_0$ lies.

The condition number of x_0 is defined by the point on the boundary of R that is furthest from the point in the complex plane with coordinates $(1, 0)$, which corresponds to the root x_0, and thus the condition estimation problem can be posed as one of constrained maximization:

$$\underset{\delta x_0 \in \mathbb{C}}{\text{maximize}} \quad \frac{1}{\varepsilon_c} \frac{|\delta x_0|}{|x_0|}$$

subject to

$$\begin{aligned} \left|\delta x_0^3 - \alpha \delta x_0^2\right| = \varepsilon_c \Big\{ &\left|(1 + \delta x_0)^3\right| + (3 + \alpha) \left|(1 + \delta x_0)^2\right| \\ &+ (3 + 2\alpha) |1 + \delta x_0| + (1 + \alpha) \Big\}. \end{aligned} \tag{3.1}$$

It is instructive to examine how the region R changes as the root separation α changes, and it is therefore desirable to plot this region for different values of α. It will be

shown that this geometric information will explain an important, but unexpected, phenomenon that is observed as α increases. The computation of the condition number then follows easily from these diagrams.

The plot of the boundary B of R requires all the complex solutions of (3.1), but their computation is not trivial. A simple approximation, however, enables points on B to be computed easily, and a check on the error that is introduced by this approximation is obtained by substituting these approximate solutions into (3.1) and verifying that the normalized residual is small.

The substitution of the approximation

$$|1 + \delta x_0| \approx 1 + |\delta x_0| \tag{3.2}$$

into (3.1) enables this equation to be approximated by

$$\begin{aligned} \left|\delta x_0^3 - \alpha \delta x_0^2\right| = \varepsilon_c \Big\{ &\left(1 + |\delta x_0|\right)^3 + (3+\alpha)\left(1 + |\delta x_0|\right)^2 \\ &+ (3+2\alpha)\left(1 + |\delta x_0|\right) + (1+\alpha) \Big\}. \end{aligned} \tag{3.3}$$

It is noted that the error introduced by the approximation (3.2) is small because $|\delta x_0| \ll 1$, and it is therefore expected that the solutions of (3.3) will be very good approximations to the solutions of (3.1).

The transformation to polar coordinates

$$\delta x_0 = r \exp i\theta, \qquad r \geq 0, \qquad 0 \leq \theta < 2\pi, \tag{3.4}$$

allows the left-hand side of (3.3) to be written as

$$r^2 \left(\left(r\cos\theta - \alpha\right)^2 + r^2 \sin^2\theta \right)^{\frac{1}{2}},$$

and it enables the right-hand side to be defined by the polynomial $g(r)$,

$$g(r) := \varepsilon_c \left\{ \left(1 + r\right)^3 + (3+\alpha)\left(1 + r\right)^2 + (3+2\alpha)(1+r) + (1+\alpha) \right\}.$$

Equation (3.3) can therefore be written as

$$r^2 \left(\left(r\cos\theta - \alpha\right)^2 + r^2 \sin^2\theta \right)^{\frac{1}{2}} = g(r),$$

from which it follows that

$$\cos\theta = \frac{1}{2\alpha r}\left(r^2 + \alpha^2 - \frac{g^2(r)}{r^4} \right). \tag{3.5}$$

Since $-1 \leq \cos\theta \leq 1$, the radius r satisfies the polynomial inequalities

$$\begin{aligned} r^6 - 2\alpha r^5 + \alpha^2 r^4 - g^2(r) \leq 0, \\ r^6 + 2\alpha r^5 + \alpha^2 r^4 - g^2(r) \geq 0. \end{aligned} \tag{3.6}$$

These inequalities are, in general, satisfied by several ranges $(r_1, r_2), (r_3, r_4), \ldots$ of r, but it is necessary to select, from these mathematical solutions, the unique solution that represents the perturbation region of x_0. The following procedure is therefore used.

The ranges of values of r that satisfy these inequalities are easily determined, and the corresponding ranges of θ can then be calculated from (3.5). The values of δx_0 are then computed from (3.4), and thus the regions in the complex plane that are mathematical solutions of (3.6) are easily determined. The following two points must be noted:

1. Since $r \geq 0$, interest is restricted to positive values of r that satisfy (3.6).
2. If there exists more than one range of values of r that satisfies (3.6), then the region in the complex plane is computed from (3.4) and (3.5) for each range $I_{jk} = (r_j, r_k)$, as specified above. It is clear that (Real δx_0, Imag δx_0) defines a local coordinate system that is centered about the point $(1, 0)$ in the complex plane. The region in which the perturbed root lies must necessarily include the point (Real δx_0, Imag δx_0) = (0,0) in this local coordinate system, and this allows the range I_{pq} that contains this point to be selected.

The substitution of these computed values of δx_0 into the exact equation (3.1) is then used to verify that the error in these approximate solutions is small.

It is easily shown from (2.4) that (3.1) is replaced by

$$\left|(\alpha + \delta x_1)^2 \delta x_1\right| = \varepsilon_c \Big\{ \left|(1 + \alpha + \delta x_1)^3\right| + (3 + \alpha)\left|(1 + \alpha + \delta x_1)^2\right| + (3 + 2\alpha)\,|1 + \alpha + \delta x_1| + (1 + \alpha) \Big\} \tag{3.7}$$

for the simple root, and it is seen that this equation follows from (3.1) by the substitution

$$\delta x_0 = \delta x_1 + \alpha. \tag{3.8}$$

This result can be obtained directly because $x_1 = x_0 + \alpha$, and thus

$$p(x_1 + \delta x_1) = p(x_0 + \alpha + \delta x_1) = p(x_0 + \delta x_0),$$

which yields (3.8). It therefore follows that if δx_0^* satisfies (3.1), then $\delta x_1^* = \delta x_0^* - \alpha$ satisfies (3.7), and thus the perturbations and condition numbers of the simple and double roots can be computed from (3.1) only, and (3.7) is not required.

Equations (2.13) and (2.14) show that if $\varepsilon_c = 10^{-7}$, the condition numbers $\kappa_c(x_0)$ and $\kappa_c(x_1)$, which are defined in (2.11), are valid if

$$\alpha \geq 4.34 \times 10^{-2} \qquad \text{and} \qquad \alpha \geq 2.56 \times 10^{-2},$$

respectively. In particular, if $\alpha < 4.34 \times 10^{-2}$, the condition number $\kappa_c(x_0)$ cannot be used because the conditions for its validity are not satisfied, and similarly, if $\alpha < 2.56 \times 10^{-2}$, the condition number $\kappa_c(x_1)$ is not valid. Condition estimation for values of α in these ranges requires, therefore, that high order terms be retained in the Taylor series of the perturbed polynomial, and this is shown in the following numerical computations.

The results of the application of this algorithm to the polynomial (2.9) are shown in Figure 3.1 for $\varepsilon_c = 10^{-7}$. This figure shows the perturbation regions, in the complex plane, of the double root $x_0 = 1$ and simple root $x_1 = (1 + \alpha)$, as the separation α of the roots increases. These regions are equal to one another for each of the stated values of α, and the constriction of the perturbation region narrows as α increases, such that it is very thin when $\alpha = 0.017698$ (Figure 3.1(d)).

Figure 3.2 is the same as Figure 3.1, but for $\alpha \geq 0.0177$. In particular, Figures 3.1(d) and 3.2(a) show that a minor change in the value of α from 0.017698 to 0.0177

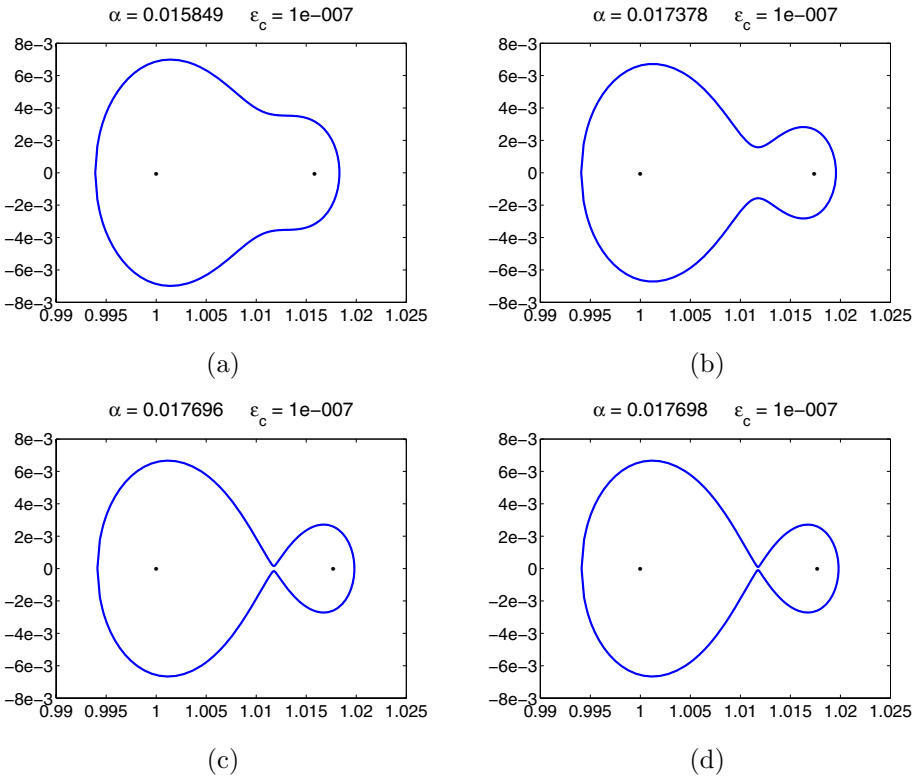


FIG. 3.1. *The perturbation regions, in the complex plane, for the double root* $x_0 = 1$ *and the simple root* $x_1 = (1 + \alpha)$*, for several values of* α*. The roots* x_0 *and* x_1 *are marked by dots.*

causes a significant change because the closed region that defines the perturbation region of both roots for $\alpha \leq 0.017698$ breaks down into two nonintersecting regions. It therefore follows that if $\varepsilon_c = 10^{-7}$ and $\alpha \geq 0.0177$, there does not exist a perturbation in the coefficients of the polynomial such that the simple root can merge into the double root, and thereby create a treble root. The figures show that as α increases, the perturbation region of each root becomes circular, which is expected. (It is noted that the scales on the axes are different, and thus although the perturbation regions appear elliptical with a distortion, they are circular with a distortion.)

Figures 3.1 and 3.2 show that the greatest displacement of the roots of $p(x)$ occurs in the region between them. Specifically, it is seen from these figures that the perturbation region of the double root $x_0 = 1$ is significantly greater on its right side than it is on its left side, and similarly, the perturbation region of the simple root $x_1 = (1 + \alpha)$ is significantly greater on its left side than it is on its right side. Finally, it was noted in section 2.2 that the restriction to the lowest order terms in the Taylor series of the perturbed polynomial necessarily implies that the regions in which the perturbed roots $x_0 + \delta x_0$ and $x_1 + \delta x_1$ lie cannot intersect, even though a merger of these regions would be expected for sufficiently small values of α, and it is necessary for $\alpha = 0$. It is clear from Figures 3.1 and 3.2 that the inclusion of high order terms in this Taylor series solves this problem because the intersection of these regions is not empty for small values of α.

The polynomial (2.9) is relatively simple, and thus the diagrams in Figures 3.1

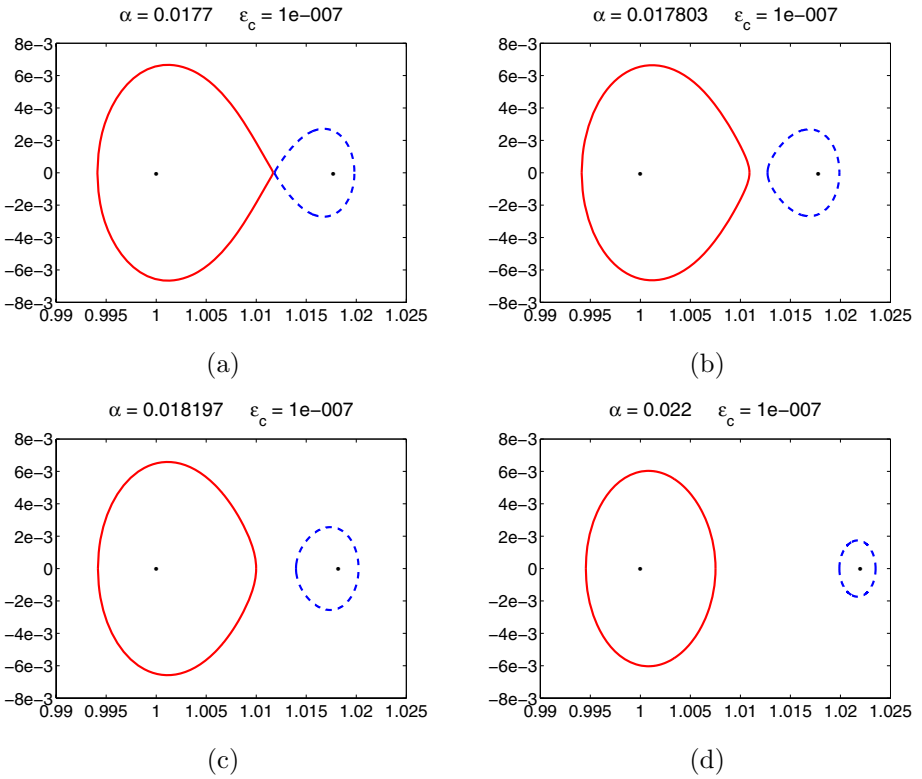


FIG. 3.2. *The perturbation regions, in the complex plane, for the double root* $x_0 = 1$ *(solid line) and the simple root* $x_1 = (1 + \alpha)$ *(dashed line), for several values of* α. *The roots* x_0 *and* x_1 *are marked by dots.*

and 3.2 could also have been generated using standard tools in a graphics package. These tools are not, however, suitable for higher degree polynomials with a more complex root structure, for which the method of high order Taylor series of a perturbed polynomial used in this paper is superior.

It was noted above that an approximation is required to obtain the plots in Figures 3.1 and 3.2, and a simple measure of the error that is introduced by this approximation is obtained by calculating the normalized residual. In particular, if $\delta\hat{x}_0$ is a solution of (3.3), then the normalized residual $r_{\delta\hat{x}_0}$ is defined by

$$r_{\delta\hat{x}_0} := \frac{h_1}{h_2},$$

where h_1 follows from (3.1),

$$h_1 = \left|\delta\hat{x}_0^3 - \alpha\delta\hat{x}_0^2\right| - \varepsilon_c\Big\{\left|(1+\delta\hat{x}_0)^3\right| + (3+\alpha)\left|(1+\delta\hat{x}_0)^2\right| + (3+2\alpha)\left|1+\delta\hat{x}_0\right| + (1+\alpha)\Big\},$$

and the denominator h_2 is equal to the sum of the absolute values of the coefficients of $\delta\hat{x}_0^3$, $\delta\hat{x}_0^2$, $\delta\hat{x}_0$, and 1,

$$h_2 = (1+\alpha) + \varepsilon_c\left\{8 + 4(3+\alpha) + 2(3+2\alpha) + (1+\alpha)\right\}.$$

For example, the coefficients of $(1+\delta\hat{x_0})^3$ are $1,3,3,1$, and their sum is equal to 8. The normalized residual was computed, and its maximum value was less than $10^{-7.5}$ for all values of α, thereby confirming that the plots in Figures 3.1 and 3.2 are excellent approximations to the perturbation regions of the double and simple roots.

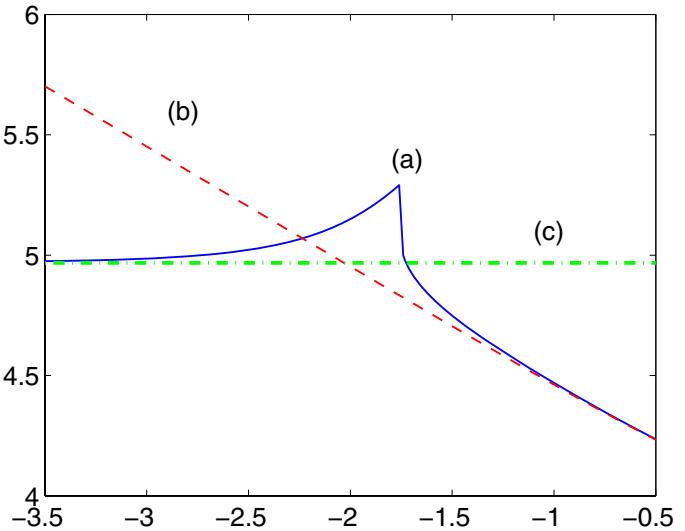


FIG. 3.3. *The variation of the logarithm, to base* 10, *of* (a) *the condition number of the double root* $x_0 = 1$*;* (b) *the condition number* $\kappa_c(x_0)$*, which is defined in* (2.11)*; and* (c) *the condition number* (2.12) *of the treble root* $x_2 = 1$ *of the polynomial* $(x-1)^3 = 0$*, against* $\log_{10}\alpha$.

Figures 3.1 and 3.2 enable the variation of the condition numbers of the double and simple roots with α to be easily computed because they are given by, respectively,

$$\max_{\delta x_0 \in \mathbb{C}} \frac{1}{\varepsilon_c}\frac{|\delta x_0|}{|x_0|} \qquad \text{and} \qquad \max_{\delta x_1 \in \mathbb{C}} \frac{1}{\varepsilon_c}\frac{|\delta x_1|}{|x_1|}.$$

This variation for the double root is shown in Figure 3.3, and the limiting values of this condition number for $\alpha \to 0$ and $\alpha \gg 0$ are also shown. In particular, the double root x_0 approaches the treble root $x_2 = 1$ of the polynomial $(x-1)^3 = 0$ as $\alpha \to 0$, the condition number of which is given in (2.12). If, however, $\alpha \gg 0$, then the condition number of x_0 is given by $\kappa_c(x_0)$, which is defined in (2.11). It is clear that, as expected, these limiting values of the condition number are obtained.

The more important feature from Figure 3.3 is the variation of the condition number with α between these limiting values. In particular, it is seen that as α increases, the condition number increases to a value that is larger than the condition number of the treble root $x_2 = 1$, and then decreases, initially very rapidly and then more slowly. This decrease is easily explained from Figures 3.1(d) and 3.2(a) because a minor change in α from 0.017698 to 0.0177 causes a significant reduction in the maximum distance between the point $(1,0)$ in the complex plane and the boundary of the region in which the perturbed root may lie. This reduction necessarily implies that the condition number of the root x_0 is also reduced significantly.

Figure 3.4 is the same as Figure 3.3 but for the simple root x_1, and it is seen that the condition number of this root also increases initially, and then decreases,

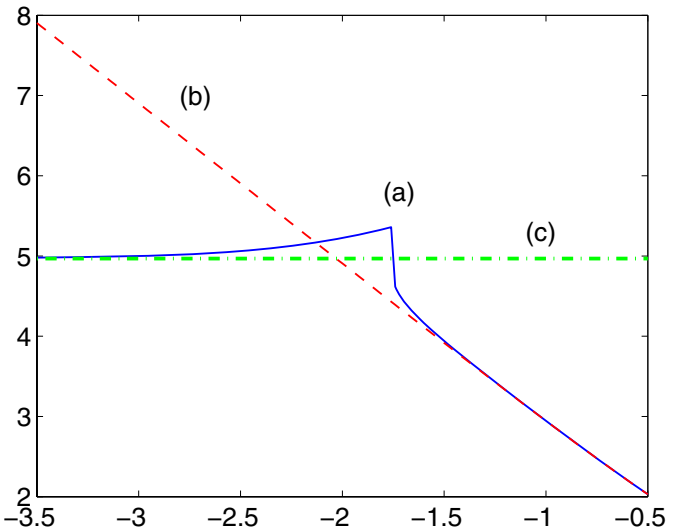


FIG. 3.4. *The variation of the logarithm, to base* 10, *of* (a) *the condition number of the simple root* $x_1 = (1+\alpha)$; (b) *the condition number* $\kappa_c(x_1)$, *which is defined in* (2.11); *and* (c) *the condition number* (2.12) *of the treble root* $x_2 = 1$ *of the polynomial* $(x-1)^3 = 0$, *against* $\log_{10}\alpha$.

as α increases. It is clear that the sharp decrease in the condition number after this initial increase can also be explained with the help of Figures 3.1(d) and 3.2(a). These numerical and geometrical results show that the inclusion of high order terms in the Taylor series of the perturbed form of $p(x)$ adds significant extra information, and not merely more accurate estimates of the condition numbers of the roots of $p(x)$.

4. Principal axes of perturbed roots. The contours that are shown in Figures 3.1 and 3.2 define the regions within which the roots lie when each coefficient is perturbed by a uniformly distributed random variable. It may be thought that the roots of the perturbed polynomial are randomly distributed in these regions, but this does not occur. Specifically, the roots of the perturbed polynomial show preferred directions, or principal axes, along which they lie.

The value $\varepsilon_c = 10^{-7}$ was retained, and the roots of the polynomial (2.9) were computed 1000 times, for a given value α. This experiment was repeated for several values of α, and the results, which are plotted in Figure 4.1, show that the roots of the perturbed polynomials lie along well-defined axes. Figure 4.1(a) shows that if $\alpha = 0$, the roots lie, to a very good approximation, along six radial arms. This computational result follows easily from (2.5), which reduces to

$$\delta x_0^3 = -\sum_{j=0}^{3} \delta a_j$$

to lowest order, and thus

$$\delta x_0 = \exp\frac{(2k+1)\pi i}{3}\left(\sum_{j=0}^{3}\delta a_j\right)^{\frac{1}{3}}, \qquad k = 0, 1, 2.$$

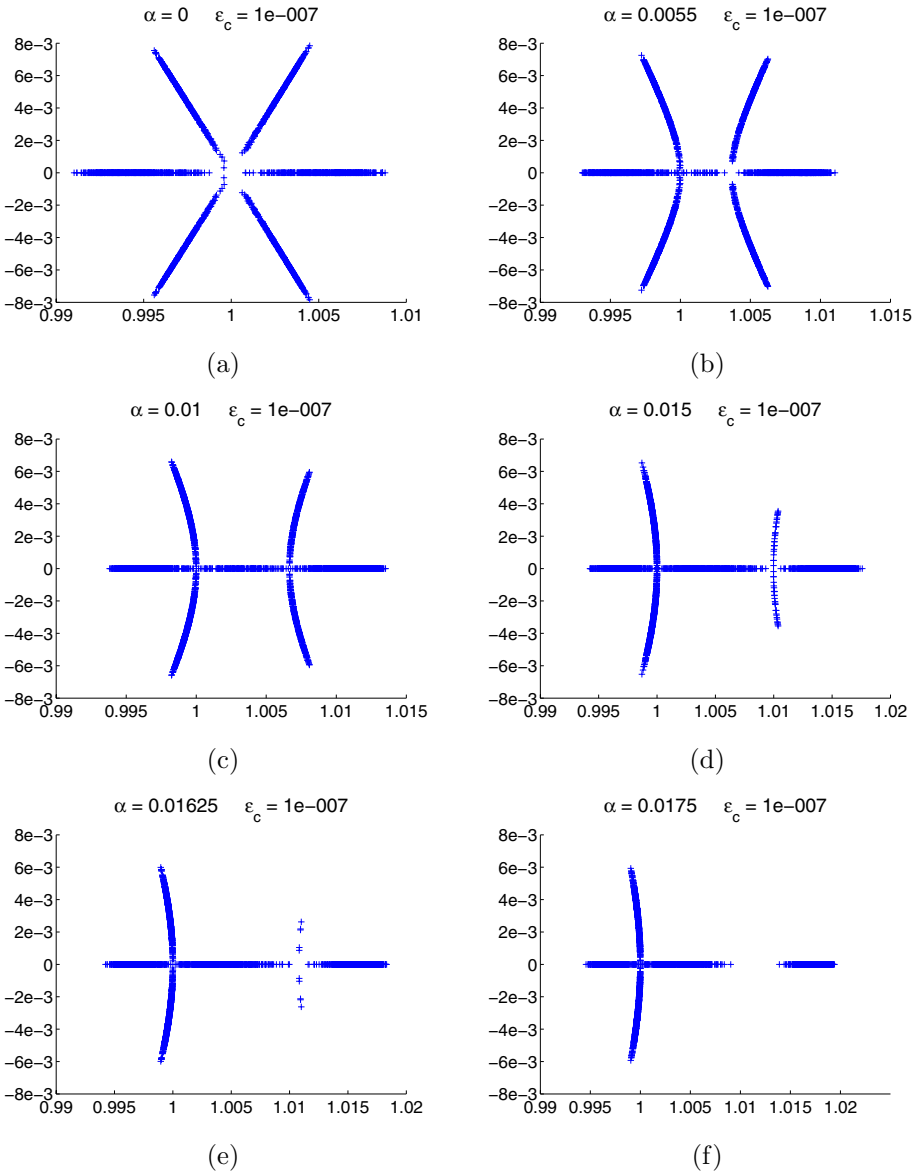


FIG. 4.1. *The principal axes of the roots of the perturbed form of $p(x)$, for six values of α.*

The sum in this expression can take both positive and negative values because each random variable δa_j has zero mean. It follows, therefore, that the set of the maximum values (in magnitude) of the computed solutions that are obtained when the perturbed polynomial is solved a large number of times is equal to

$$\delta x_0^+ \bigcup \delta x_0^-, \tag{4.1}$$

where δx_0^+ and δx_0^- are the sets of the maximum positive and minimum negative

computed values of δx_0, respectively,

$$\delta x_0^+ = \left\{ \left(\sum_{j=0}^{3} |\delta a_j| \right)^{\frac{1}{3}} \exp \frac{(2k+1)\pi i}{3} \exp \frac{2l\pi i}{3}, \qquad k,l = 0,1,2 \right\},$$

$$\delta x_0^- = \left\{ \left(\sum_{j=0}^{3} |\delta a_j| \right)^{\frac{1}{3}} \exp \frac{(2k+1)\pi i}{3} \exp \frac{(2l+1)\pi i}{3}, \qquad k,l = 0,1,2 \right\}.$$

It is readily verified that the union (4.1) of the sets δx_0^+ and δx_0^- yields a set of radial vectors of length $(\sum_{j=0}^{3} |\delta a_j|)^{\frac{1}{3}}$, which are aligned at angles of $\frac{k\pi i}{3}$, $k = 0, \ldots, 5$, to the real axis. Furthermore, it follows from (2.2) and (2.10) that

$$\max \left(\sum_{j=0}^{3} |\delta a_j| \right)^{\frac{1}{3}} = \left(8 \times 10^{-7}\right)^{\frac{1}{3}} = 9.3 \times 10^{-3},$$

and Figure 4.1(a) shows that this theoretical value is in excellent agreement with its computed value.

As α increases, the principal axes that are aligned at $\pm\frac{\pi}{3}$ move to the right and decrease in length. Furthermore, the decreasing number of roots that lie along these principal axes shows that the probability of a root lying along them decreases, and the principal axes of the double and simple roots split at a value $\alpha = \alpha^*$, where $0.01625 < \alpha^* < 0.0175$. It is interesting to note that Figure 3.2(a) shows that, if worst case perturbations are used, then the perturbation regions of the roots become disjoint when $\alpha = 0.0177$. Figure 4.1(f) shows that the principal axes at $\pm\frac{\pi}{2}$ for the double root are slightly curved when $\alpha = 0.0175$, which implies that this root interacts, albeit slightly, with the simple root for this value of α.

Figure 4.2 shows the difference, for six values of α, between the region R that defines the area within which the roots of the perturbed polynomials lie, as shown in Figures 3.1 and 3.2, and the principal axes of the roots. The principal axes in Figure 4.2 were obtained by computing the roots of the perturbed polynomial 1000 times for each value of α. The figure clearly shows that the probability of a root of a perturbed polynomial lying in most of the region R is vanishingly small, and thus a computation based on R may yield results that differ significantly from those that are based on the principal axes of the roots.

Finally, it is noted that computational experiments showed that the principal axes of the roots of a family of perturbed polynomials lose their well-defined property as the multiplicities of the roots of the exact polynomial increase.

5. Discussion and summary. This paper has considered the use of high order Taylor series for the computation of the componentwise condition number of a root of a polynomial that is near one or more other roots. The results in this paper cannot be obtained by its established formula (2.3) because it uses only the lowest order terms in the Taylor series of the perturbed polynomial. It was shown that if a multiple root x_0 splits up into a series of distinct roots x_i, $i = 1, \ldots, r$, of lower multiplicities, then the componentwise condition number of each of these distinct roots increases initially as the distance between them and x_0 increases.

It was stated in section 1 that the numerical and geometric issues that arise when high order Taylor series are used for the computation of the condition number of a

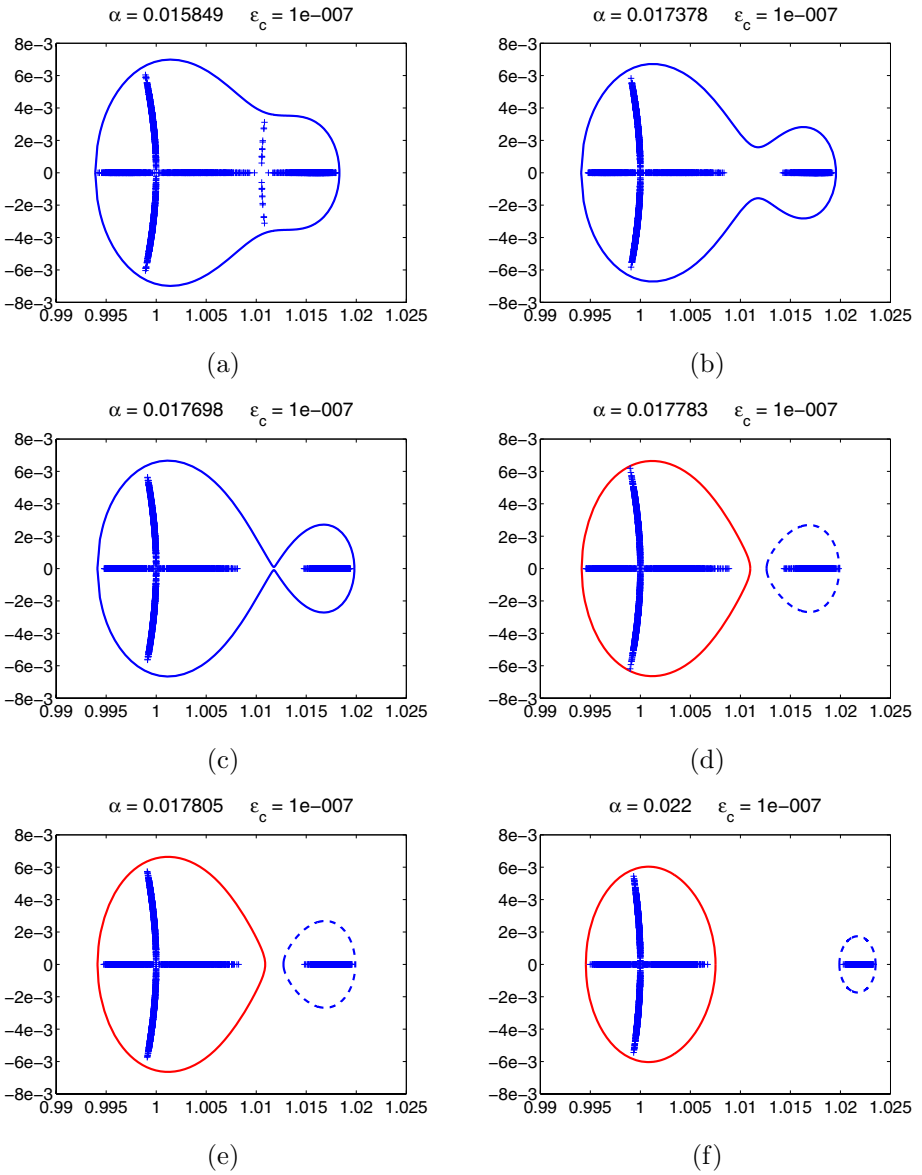


FIG. 4.2. *The regions within which the roots of the perturbed polynomial $p(x)$ lie, and their principal axes, for six values of α. The perturbation regions of the double and simple roots are equal to one another in* (a), (b), *and* (c), *but they are disjoint in* (d), (e), *and* (f), *where the solid line (–) and dashed line (- -) are the boundaries of the perturbation regions for the double and simple roots, respectively.*

root of a polynomial are considerable, and they therefore pose issues that require further work:

- The simple example in the paper showed that it is advantageous to plot the boundary of the region in the complex plane in which the perturbed root lies because it adds geometric insight and enables the condition number to be computed easily and quickly. The computation of this boundary for an

arbitrary polynomial is, however, difficult because it requires the computation of the complex solutions of a nonlinear equation in a complex variable. Also, there exist several methods for tracing a curve $f(z) = 0$, but plotting the boundary of the region in which the perturbed root lies is difficult because $f(z)$ is not differentiable for all values of z.

- It was shown that the roots of a set of perturbed polynomials lie along principal axes, and that a result that is based on them may differ from the equivalent result that is based on the condition number. This difference makes it desirable to be able to compute these principal axes.

It has recently been shown [11] that when it is required to compute the condition number of a point on an algebraic curve, and this point is near a singularity, the restriction to lowest order terms imposes constraints that are very similar to those that are described in section 2.2, and high order terms must be included in order to remove these restrictions.

Acknowledgments. The work that is described in this paper was performed when the author visited Professor Gene Golub at The Department of Computer Science at Stanford University, and he wishes to thank Professor Golub for his hospitality and for the enjoyable environment that he provided during the visit. The author also wishes to thank an anonymous referee whose comments improved the paper significantly.

REFERENCES

[1] W. Gautschi, *On the condition of algebraic equations*, Numer. Math., 21 (1973), pp. 405–424.
[2] W. Gautschi, *The condition of polynomials in power form*, Math. Comput., 33 (1979), pp. 343–352.
[3] W. Gautschi, *Questions of Numerical Condition Related to Polynomials*, MAA Studies in Numerical Analysis 24, G. Golub, ed., Mathematical Association of America, Washington, DC, 1984, pp. 140–177.
[4] R. Mosier, *Root neighborhoods of a polynomial*, Math. Comput., 47 (1986), pp. 265–273.
[5] K. C. Toh and L. N. Trefethen, *Pseudozeros of polynomials and pseudospectra of companion matrices*, Numer. Math., 68 (1994), pp. 403–425.
[6] J. Wilkinson, *The evaluation of zeros of ill-conditioned polynomials*, Numer. Math., 1 (1959), pp. 150–166.
[7] J. Wilkinson, *Rounding Errors in Algebraic Processes*, Prentice-Hall, Englewood Cliffs, NJ, 1963.
[8] J. Wilkinson, *The perfidious polynomial*, in MAA Studies in Numerical Analysis, Vol. 24, G. Golub, ed., Mathematical Association of America, Washington, DC, 1984, pp. 1–28.
[9] J. R. Winkler, *Condition numbers of a nearly singular simple root of a polynomial*, Appl. Numer. Math., 38 (2001), pp. 275–285.
[10] J. R. Winkler, *A statistical analysis of the numerical condition of multiple roots of polynomials*, Comput. Math. Appl., 45 (2003), pp. 9–24.
[11] J. R. Winkler, *Backward errors and condition numbers of regular and singular points on algebraic curves*, in Mathematics of Surfaces XI, R. Martin, H. Bez, and M. Sabin, eds., Springer-Verlag, Berlin, 2005, pp. 413–433.
[12] J. R. Winkler, *The numerical condition of univariate and bivariate degree elevated Bernstein polynomials*, J. Comput. Appl., 191 (2006), pp. 32–49.
[13] H. Zhang, *Numerical condition of polynomials in different forms*, Electron. Trans. Numer. Anal., 12 (2001), pp. 66–87.

SIAM J. SCI. COMPUT.
Vol. 28, No. 4, pp. 1437–1462
© 2006 Society for Industrial and Applied Mathematics

EFFICIENT AND RELIABLE MULTIPRECISION IMPLEMENTATION OF ELEMENTARY AND SPECIAL FUNCTIONS*

ANNIE CUYT†, BRIGITTE VERDONK†, AND HAAKON WAADELAND‡

Abstract. Special functions are pervasive in all fields of science. The most well-known application areas are in physics, engineering, chemistry, computer science, and statistics. Because of their importance, several books and a large collection of papers have been devoted to the numerical computation of these functions. The technique for providing a floating-point implementation of a function differs substantially when going from a fixed finite precision context to a finite multiprecision context. In the former, the aim is to provide an optimal mathematical model, valid on a reduced argument range and requiring as few operations as possible. Here optimal means that, in relation to the model's complexity, the truncation error is as small as it can get. The total relative error, including round-off error and possible argument reduction effect, should not exceed a prescribed threshold. In a finite multiprecision context, the goal is to provide a more generic technique, from which an approximant yielding the user-defined accuracy can be deduced at runtime. Hence best approximants are not an option since these models have to be recomputed every time the precision is altered and a function evaluation is requested. At the same time the generic technique should generate an approximant of as low complexity as possible. In the current approach we point out how continued fraction representations of functions can be helpful in the multiprecision context. The newly developed generic technique is based mainly on the use of sharpened a priori truncation error estimates for real continued fraction representations of a real variable, developed in section 3. As illustrated in section 4, the technique is very efficient and even quite competitive when compared to the traditional fixed precision implementations. The implementation is reliable in the sense that it allows one to return a sharp interval enclosure for the requested function evaluation, at the same cost. The paper follows a recipe style. In section 2 we gather the ingredients for the new results. In section 3 we construct or prepare, for a general function $f(x)$, a continued fraction approximant satisfying all requirements of a proper implementation. In section 4 the procedure is illustrated with results obtained for several specific functions $f(x)$.

Key words. continued fractions, special functions, verified, variable precision

AMS subject classifications. 11A55, 30B70, 41A20, 33B99, 33C99, 65G20

DOI. 10.1137/050629203

1. Introduction.

Virtually all present-day computer systems, from personal computers to the largest supercomputers, implement the IEEE 64-bit floating-point arithmetic standard, which provides 53 binary or approximately 16 decimal digits accuracy. For most scientific applications, this is more than sufficient. However, for a rapidly expanding body of applications, 64-bit IEEE arithmetic is no longer sufficient. These applications range from some interesting new mathematical investigations to large-scale physical simulations performed on highly parallel supercomputers. Moreover, in these applications, portions of the code typically involve numerically sensitive calculations, which produce results of questionable accuracy using conventional arithmetic. These inaccurate results may in turn induce other errors, such as taking the wrong path in a conditional branch. Such blocks of code benefit enormously from a

*Received by the editors April 14, 2005; accepted for publication (in revised form) July 22, 2005; published electronically September 5, 2006.
http://www.siam.org/journals/sisc/28-4/62920.html
†Department of Mathematics and Computer Science, University of Antwerp, Middelheimlaan 1, B2020 Antwerp, Belgium (annie.cuyt@ua.ac.be, brigitte.verdonk@ua.ac.be).
‡Department of Mathematical Sciences, Norwegian University of Science and Technology, N7491 Trondheim, Norway (haakonwa@math.ntnu.no).

combination of reliable numeric techniques and the use of high-precision arithmetic. Indeed, the aim of reliable numeric techniques is to deliver, together with the computed result, a guaranteed upper bound on the total error or, equivalently, to compute an enclosure for the exact result.

To date, even environments such as Maple, Mathematica, and MATLAB and libraries such as IMSL, CERN, and NAG offer no routines for the reliable evaluation of special functions. The following quotes concisely express the need for new developments in the evaluation of special functions:

- [illegible] D. Lozier, general director of the Digital Library for Mathematical Functions project, and F. Olver [3].
- [illegible] 20 [illegible] 30 [illegible] Cleve Moler, founder of MATLAB [13].

2. Ingredients. A lot of well-known constants in mathematics, physics, and engineering, as well as elementary and special functions, enjoy very nice and rapidly converging continued fraction representations. Among these, we almost always find one or more limit-periodic continued fractions. In this paper we focus on the use of these limit-periodic continued fractions in the reliable multiprecision implementation of the functions they represent. We restrict ourselves to the real-valued case. This implementation is built on top of multiprecision IEEE 754-854 compliant computer arithmetic.

2.1. IEEE-based arithmetic. Let us assume to have at our disposal a multiprecision IEEE 754-854 compliant floating-point implementation of the basic operations, comparisons, and base and type conversions. Such an implementation is characterized by four parameters: the internal base β, the precision t, and the exponent range $[L, U]$. In the current generic context, we are at least aiming at implementations for $\beta = 2$ at nonstandard precisions $t > 64$, and at implementations for use with $\beta = 2^i$ where $i > 1$ or $\beta = 10^j$ where $j \geq 1$. An IEEE compliant implementation offers four rounding modes: upward $\triangle$, downward $\triangledown$, truncation $\bowtie$, and round-to-nearest $\bigcirc$. We are especially interested in the first two rounding modes because a correct implementation in these rounding modes provides a reliable enclosure of the exact function value.

To provide an implementation of a function $f(x)$ in a particular precision t, one first needs to develop a mathematical model or approximation $F(x)$ for $f(x)$ that enjoys a very small relative error compared to the precision in use. This is usually a very time-consuming effort because the model changes whenever the precision does. The sum of the truncation error $|f(x) - F(x)|/|f(x)|$ and the rounding error $|F(x) - \mathtt{F}(x)|/|f(x)|$, where $\mathtt{F}(x)$ denotes the machine implementation of the model $F(x)$, should preferably not exceed a few `ulp`, where

$$1\ \mathtt{ulp} = \beta^{-t+1}. \tag{1}$$

A typical double precision implementation ($\beta = 2,\ t = 53$) of the elementary functions achieves this in about 25 basic operations. When analyzing the efficiency of our

multiprecision implementation, we compare the number of basic operations, required in the new approach when the precision is set to $t = 53$, to this reference.

Since the accumulated rounding error is included in the total error which we bound by only a few `ulp` (typically 1 or 2), the actual evaluation of $f(x)$ needs to take place in a slightly larger working precision $s > t$. The optimal working precision is determined dynamically, depending on the rounding error analysis and the required accuracy. If applicable (not for special functions), our rounding error analysis also takes the effect of argument reduction into account. In any case a fully reliable evaluation of $f(x)$ over the domain is guaranteed.

2.2. Continued fraction representations and tails. Let us consider a continued fraction representation of the form

$$f = \frac{a_1}{1+\frac{a_2}{1+\cdots}} = \frac{a_1\rfloor}{\lceil 1}+\frac{a_2\rfloor}{\lceil 1}+\cdots=\sum_{n=1}^{\infty}\frac{a_n\rfloor}{\lceil 1}, \qquad a_n := a_n(x), \qquad f := f(x). \tag{2}$$

Here a_n is called the nth partial numerator. We use the notation f and $f(x)$ interchangeably. The latter is preferred when the dependence on x needs to be emphasized. We denote by the Nth approximant $f_N(w)$ or $f_N(x;w)$ and Nth tail t_N or $t_N(x)$ of (2), respectively, the values

$$f_N(w) = f_N(x;w) = \sum_{n=1}^{N-1}\frac{a_n\rfloor}{\lceil 1}+\frac{a_N\rfloor}{\lceil 1+w}, \tag{3}$$

$$t_N = t_N(x) = \sum_{n=N+1}^{\infty}\frac{a_n\rfloor}{\lceil 1}. \tag{4}$$

We also need approximants of tails and therefore introduce the notation $f_N^{(k)}(w)$ or $f_N^{(k)}(x;w)$ for

$$f_N^{(k)}(w) = f_N^{(k)}(x;w) = \sum_{n=k+1}^{k+N-1}\frac{a_n\rfloor}{\lceil 1}+\frac{a_{N+k}\rfloor}{\lceil 1+w}, \qquad f_N^{(0)}(w) = f_N(w).$$

Sometimes the notation $f^{(k)}$ is used for the tail t_k. A continued fraction is said to converge if $\lim_{N\to\infty} f_N(0)$ exists. Note that convergence to ∞ is allowed. In the present paper we assume the continued fractions to converge. Moreover, we restrict ourselves to the case where some $w \neq 0$ can be chosen such that $\lim_{N\to\infty} f_N(w) = \lim_{N\to\infty} f_N(0)$.

The Nth approximant of a continued fraction can also be written as

$$f_N(w) = (s_1 \circ \cdots \circ s_N)(w), \qquad s_n(w) = \frac{a_n}{1+w}, \qquad n = N,\dots,1.$$

Using the linear fractional transformations s_n, one can define a sequence $\{V_n\}_{n\in\mathbb{N}}$ of value sets for f by

$$s_n(V_n) = \frac{a_n}{1+V_n} \subseteq V_{n-1}, \qquad n \geq 1. \tag{5}$$

The importance of such a sequence of sets lies in the fact that these sets keep track of where certain values lie. For instance, if $w \in V_N$, then $f_N(w) \in V_0$ and $f_{N-k}^{(k)}(w) \in V_k$.

Also $t_N \in \overline{V}_N$ and $f \in \overline{V}_0$, where $\overline{V}$ denotes the closure of V. An equally important role is played by a sequence of convergence sets $\{E_n\}_{n\in\mathbb{N}}$, of which the elements guarantee convergence of the continued fraction (2) as long as each partial numerator a_n belongs to the respective set E_n:

$$\forall n \geq 1 : a_n \in E_n \Rightarrow \sum_{n=1}^{\infty} \frac{a_n\rfloor}{\lceil 1} \text{ converges.} \tag{6}$$

A sequence $\{V_n\}_{n\in\mathbb{N}}$ is called a sequence of value sets for a sequence $\{E_n\}_{n\in\mathbb{N}}$ of convergence sets if (5) holds for all $a_n \in E_n$. Value sets can also be defined for nonconvergent continued fractions (then the E_n are called element sets), but in the current context this form of generality does not interest us.

It is well known that the tail or rest term of a convergent Taylor series expansion converges to zero. It is less well known that the tail of a convergent continued fraction representation does not need to converge to zero; it does not even need to converge at all. We give an example for each of the cases. Take, for instance, the continued fraction expansion

$$\frac{\sqrt{1+4x}-1}{2} = \sum_{n=1}^{\infty} \frac{x\rfloor}{\lceil 1}, \qquad x \geq -\frac{1}{4}.$$

Each tail t_N converges to $\frac{1}{2}(\sqrt{1+4x}-1)$ as well. More remarkable is that the even-numbered tails of the convergent continued fraction

$$\sqrt{2}-1 = \sum_{n=1}^{\infty} \left(\frac{(3+(-1)^n)/2\rfloor}{\lceil 1} \right) = \frac{1\rfloor}{\lceil 1} + \frac{2\rfloor}{\lceil 1} + \frac{1\rfloor}{\lceil 1} + \frac{2\rfloor}{\lceil 1} + \cdots$$

converge to $\sqrt{2}-1$ while the odd-numbered tails converge to $\sqrt{2}$ (hence the sequence of tails does not converge), and that the sequence of tails $\{t_N\}_{N\geq 1} = \{N+1\}_{N\geq 1}$ of

$$1 = \sum_{n=1}^{\infty} \frac{n(n+2)\rfloor}{\lceil 1}$$

converges to $+\infty$.

When carefully monitoring the behavior of these continued fraction tails, very accurate approximants $f_N(w)$ for f can be computed by making an appropriate choice for w. We call a continued fraction of the form (2) limit-periodic with period k if

$$\lim_{p\to\infty} a_{pk+q} = \tilde{a}_q, \qquad q = 1, \ldots, k.$$

More can be said about tails of limit-periodic continued fractions with period one, also called limit-periodic continued fractions. Let (2) converge and be limit-periodic with $a_n \geq -\frac{1}{4}$ and $\lim_{n\to\infty} a_n = \tilde{a} < \infty$. If $\tilde{w}$ is the in modulus smaller fixpoint of the linear fractional transformation $s(w) = \tilde{a}/(1+w)$, then

$$\tilde{w} = -\frac{1}{2} + \sqrt{\tilde{a} + \frac{1}{4}} = \lim_{N\to\infty} t_N$$

and, according to [15],

$$\lim_{N\to\infty} \left| \frac{f(x) - f_N(x;\tilde{w})}{f(x) - f_N(x;0)} \right| = 0. \tag{7}$$

Hence a suitable choice of w in (3) may result in more rapid convergence of the approximants ($w = 0$ is usually used as a reference).

In this paper we replace the condition that (2) converges, in the case of limit-periodic continued fractions, with the conditions $a_n \geq -\frac{1}{4}$ and the sequence $\{a_n\}_{n\in\mathbb{N}}$ is bounded [11, pp. 150–159]. These easier conditions automatically imply that $\tilde{a} \geq -\frac{1}{4}$ and $\tilde{w}$ is real. They also simplify the description of the results, which can, however, be generalized to the particular situations where $a_n \leq -\frac{1}{4}$, $\tilde{a} = -\frac{1}{4}$, and (2) converges because the convergence speed of the partial numerators can be termed safe (see [11, p. 159]).

2.3. The oval sequence theorem. Most truncation error upper bounds for $|f(x) - f_N(x;w)|$ are given for the classical choice $w = 0$. For continued fractions with partial numerators of the form $a_n(x) = \alpha_n x$ with $\alpha_n > 0$ we refer among others to the a priori Gragg–Warner bound

(8)
$$|f(x) - f_N(x;0)| \leq 2\frac{|a_1|}{\cos\phi}\prod_{k=2}^{N}\frac{\sqrt{1+4|a_k|/\cos^2(\phi)}-1}{\sqrt{1+4|a_k|/\cos^2(\phi)}+1}, \qquad -\pi < 2\phi = \arg(x) < \pi,$$

which holds for $N \geq 2$, and the a posteriori Henrici–Pfluger bound

$$|f(x) - f_N(x;0)| \leq \begin{cases} |f_N(x;0) - f_{N-1}(x;0)|, & |\arg(x)| \leq \pi/2, \\ \dfrac{|f_N(x;0) - f_{N-1}(x;0)|}{|\sin(\arg(x))|}, & \pi/2 < |\arg(x)| < \pi. \end{cases}$$

A posteriori bounds are usually slightly sharper because they exploit the information contained in already computed approximants. But they are of no use in a variable precision context: One does not want to scan all approximants until the truncation error threshold is satisfied. The Gragg–Warner a priori bound does not suffer from this disadvantage, but it is only valid when $a_n(x) = \alpha_n x$ with $\alpha_n > 0$. In this paper we develop a practical and sharp truncation error bound for the case $w \neq 0$, which is valid for all continued fractions with real partial numerators $a_n(x)$. Our point of departure for this is the oval sequence theorem from which a priori truncation error estimates can be obtained in case $w \neq 0$.

In the general formulation of the oval sequence theorem (see Theorem 1) [11, pp. 145–147], which holds in the complex plane, the value sets V_n are disks and the convergence sets E_n are Cartesian ovals (the situation where the continued fraction does not necessarily converge can also be considered but is of no interest to us). We denote the complex conjugate of the center C_n by $\overline{C}_n$.

In the real case both V_n and E_n reduce to intervals. We reformulate the more specific real version in Theorem 2 (which allows for point intervals) and provide an elegant short proof.

THEOREM 1. [illegible] $0 < r_n < |1 + C_n|$ [illegible] $n \geq 0$ [illegible] $|C_{n-1}|r_n < |1 + C_n|r_{n-1}$ [illegible] $n \geq 1$ [illegible] $\{V_n\}_{n\in\mathbb{N}}$ [illegible]

$$V_n = \{w \in \mathbb{R} : |C_n - w| < r_n\}$$

[illegible] $\{E_n\}_{n\in\mathbb{N}_0}$ [illegible]

$$E_n = \{a \in \mathbb{R} : |a(1+\overline{C}_n) - C_{n-1}(|1+C_n|^2 - r_n^2)| + r_n|a| \leq r_{n-1}(|1+C_n|^2 - r_n^2)\}.$$

[illegible] $w \in V_N$ [illegible] $|f(x) - f_N(x;w)|$ [illegible]

$$|f(x) - f_N(x;w)| \le 2r_N \frac{|C_0| + r_0}{|1 + C_N| - r_N} \times \prod_{k=1}^{N-1} M_k, \tag{9}$$

[illegible] $M_k = \max\{|u/(1+u)| : u \in \overline{V}_k\}$

In the real case the set E_n given above is actually an interval. The set $\overline{V}_k = [C_k - r_k, C_k + r_k]$ and the maximum M_k is given by

$$M_k = \max\left(\left|\frac{C_k - r_k}{1 + C_k - r_k}\right|, \left|\frac{C_k + r_k}{1 + C_k + r_k}\right|\right)$$

since the function $w/(1+w)$ is a strictly increasing function. Let us simplify the statement and proof of the oval sequence theorem for the case of real interval sequences. At the same time, the new proof delivers a bound for the relative error instead of the absolute error, and we have $M_k \le 1$. In what follows we refer to this theorem as the interval sequence theorem.

THEOREM 2. [illegible] n [illegible] L_n [illegible] R_n [illegible] $-\frac{1}{2} \le L_n \le R_n < \infty$ [illegible]

$$\begin{aligned} b_n &:= (1 + \operatorname{sign}(L_n) \max(|L_n|, |R_n|))\, L_{n-1}, \\ c_n &:= (1 + \operatorname{sign}(L_n) \min(|L_n|, |R_n|))\, R_{n-1} \end{aligned}$$

[illegible] $b_n \le c_n$ [illegible] $0 \le b_n c_n$ [illegible] $\{V_n\}_{n\in\mathbb{N}}$ [illegible] $V_n = [L_n, R_n]$ [illegible] [illegible] $\{E_n\}_{n\in\mathbb{N}}$ [illegible]

$$E_n = [b_n, c_n] = \begin{cases} [(1 + R_n)L_{n-1}, (1 + L_n)R_{n-1}], & b_n \ge 0, \\ [(1 + L_n)L_{n-1}, (1 + R_n)R_{n-1}], & b_n \le 0. \end{cases}$$

[illegible] $w \in V_N$ [illegible] $|f(x) - f_N(x;w)|/|f(x)|$ [illegible]

$$\left|\frac{f(x) - f_N(x;w)}{f(x)}\right| \le \frac{R_N - L_N}{1 + L_N} \prod_{k=1}^{N-1} M_k, \tag{10}$$

[illegible] $M_k = \max\{|u/(1+u)| : u \in V_k\} = \max\{|L_k/(1+L_k)|, |R_k/(1+R_k)|\}$ [illegible]. The relation between E_n and V_n is expressed in (5). For $E_n = [b_n, c_n] \subset \mathbb{R}^+$ this translates to

$$\begin{aligned} L_{n-1} &\le \frac{b_n}{1 + R_n}, \\ R_{n-1} &\ge \frac{c_n}{1 + L_n}, \end{aligned}$$

while for $E_n \subset \mathbb{R}^-$ it translates to

$$\begin{aligned} L_{n-1} &\le \frac{b_n}{1 + L_n}, \\ R_{n-1} &\ge \frac{c_n}{1 + R_n}. \end{aligned}$$

Using the notation $f^{(j)}(x)$ for the jth tail of the continued fraction $f(x)$, we obtain

$$\begin{aligned} f(x) - f_N(x;w) &= \frac{a_1}{1+f^{(1)}(x)} - \frac{a_1}{1+f_{N-1}^{(1)}(x;w)} \\ &= \frac{-f^{(0)}(x)(f^{(1)}(x) - f_{N-1}^{(1)}(x;w))}{1+f_{N-1}^{(1)}(x;w)} \\ &= -f(x)\frac{f^{(N)}(x)-w}{1+w}\prod_{j=1}^{N-1}\left(\frac{-f^{(j)}(x)}{1+f_{N-j}^{(j)}(x;w)}\right). \end{aligned}$$

Hence

$$\left|\frac{f(x)-f_N(x;w)}{f(x)}\right| \le \frac{R_N - L_N}{1+w}\left|\prod_{j=1}^{N-1}\frac{f^{(j)}(x)}{1+f_{N-j}^{(j)}(x;w)}\right|.$$

Now for $1 \le k \le \lfloor (N-1)/2 \rfloor$, we have

$$\begin{aligned} &\frac{f^{(2k)}(x)}{1+f_{N-2k}^{(2k)}(x;w)}\,\frac{f^{(2k-1)}(x)}{1+f_{N-2k+1}^{(2k-1)}(x;w)} \\ &\quad= \frac{f^{(2k)}(x)}{1+f_{N-2k}^{(2k)}(x;w)}\,\frac{a_{2k}}{1+f^{(2k)}(x)}\,\frac{1}{1+f_{N-2k+1}^{(2k-1)}(x;w)} \\ &\quad= \frac{f^{(2k)}(x)}{1+f^{(2k)}(x)}\,\frac{f_{N-2k+1}^{(2k-1)}(x;w)}{1+f_{N-2k+1}^{(2k-1)}(x;w)}. \end{aligned}$$

If $N-1$ is odd, then the last factor can be combined with $1/(1+w)$ into

$$\frac{f^{(N-1)}(x)}{1+f_1^{(N-1)}(x;w)}\,\frac{1}{1+w} = \frac{f_1^{(N-1)}(x;w)}{1+f_1^{(N-1)}(x;w)}\,\frac{1}{1+f^{(N)}(x)}.$$

Since $\max_{u\in V_k}|u/(1+u)| = M_k$, the theorem is proved. □

An upper bound for the truncation error $|f(x) - f_N(x;w)|$ is obtained by multiplying the right-hand side of (10) by R_0 which is an upper bound for $|f(x)|$.

The smaller the sets V_n, in other words, the smaller $R_n - L_n$, and the more precisely one knows the M_k, the smaller the upper bound on the truncation error $|f(x) - f_N(x;w)|/|f(x)|$ becomes. We come back to this issue in section 3.

In section 3 we also combine the interval sequence theorem with the characteristic monotonicity behavior of the tails of some limit-periodic continued fraction representations, which we derive now.

2.4. Monotonicity properties of tails. To prepare these results, we prove some easy lemmas. Lemma 1 mainly serves to establish computable upper bounds for tails of continued fractions, while Lemma 2 provides additional information about the relation between continued fraction approximants and tail estimates. In Lemmas 3 and 4 we obtain some results on the boundedness and monotonicity behavior of the sequence of tails. In the formulation of these lemmas the notions increasing and decreasing for sequences of numbers are never meant in the strict sense: A constant sequence can be considered to be decreasing as well as increasing.

To obtain monotonicity results about the continued fraction tails, we further distinguish between

- the fractions where $a_n \to \tilde{a}$ from one side, say $\{a_n\}_{n\in\mathbb{N}}$ is a decreasing (or increasing) sequence with $\lim_{n\to\infty} a_n = \tilde{a}$, and
- the fractions where $a_n \to \tilde{a}$ in an alternating fashion, say the sequences $\{a_{2n+1}\}_{n\in\mathbb{N}}$ and $\{a_{2n}\}_{n\in\mathbb{N}}$, respectively, decrease and increase toward $\tilde{a}$.

For continued fractions with positive a_n it suffices to study the latter case in detail, while the former case is the basic building block when dealing with continued fractions with negative a_n. This becomes clear in section 4 where all different types of behavior for the partial numerators are illustrated and can be dealt with, starting from the two mentioned basic types. How to deal with a continued fraction containing both positive and negative partial numerators is explained in section 3.4. From section 3.4 it should be clear that the condition in most lemmas for the partial numerators to be either all positive or all negative is not a true restriction but merely reflects a choice made by the authors to simplify the description of the results.

LEMMA 1.

1. [illegible] $a_n > 0$ [illegible] (2) [illegible] $k > 0$ [illegible] $\{f_{2n}^{(k)}(0)\}_{n\in\mathbb{N}}$ [illegible] $\{f_{2n+1}^{(k)}(0)\}_{n\in\mathbb{N}}$ [illegible]

$$f_{2n}^{(k)}(0) \le t_k \le f_{2n+1}^{(k)}(0).$$

2. [illegible] $a_n < 0$ [illegible] $\{a_n\}_{n\in\mathbb{N}}$ [illegible] $\lim_{n\to\infty} a_n = \tilde{a} \ge -\frac{1}{4}$ [illegible] $k > 0$ [illegible] t_k [illegible] (2) [illegible]

$$\frac{-1+\sqrt{4\tilde{a}+1}}{2} \le t_k \le \frac{-1+\sqrt{4a_{k+1}+1}}{2}.$$

[illegible]. Part 1 is well known from continued fraction literature [1, p. 223]. It remains to prove part 2. We give the proof for the upper bound in part 2 because the lower bound is proved analogously. Consider the approximant $f_n^{(k)}$ of t_k. Since $0 \le 1 + a_{k+n-j} \le 1 + a_{k+1}$ for $j = 0, \ldots, n-1$, we find that

$$\begin{aligned}
\frac{a_{k+n-1}}{1+a_{k+n}} &\le \frac{a_{k+1}}{1+a_{k+1}},\\
\frac{a_{k+n-2}}{1+\frac{a_{k+n-1}}{1+a_{k+n}}} &\le \frac{a_{k+1}}{1+\frac{a_{k+1}}{1+a_{k+1}}}\\
&\vdots\\
f_n^{(k)} &\le \frac{a_{k+1}}{\left\lfloor 1\right\rfloor} + \cdots + \frac{a_{k+1}}{\left\lfloor 1\right\rfloor}.
\end{aligned}$$

In the limit the inequality is preserved because of the convergence of (2), leading to

$$t_k \le \frac{-1+\sqrt{4a_{k+1}+1}}{2},$$

which concludes this part of the proof. □

The condition that (2) converges is automatically satisfied in part 2 of Lemma 1 and in Lemmas 2, 3, and 4 because of the boundedness of the partial numerators a_n.

LEMMA 2. [illegible] $a_n \ge -\frac{1}{4}$ [illegible] $\{a_n\}_{n\in\mathbb{N}}$ [illegible]

1. $t_k \ge -\frac{1}{2}$ [illegible] $\operatorname{sign}(t_k) = \operatorname{sign}(a_{k+1})$ [illegible] k

2. [illegible] $a_n > 0$ [illegible] $0 \le w_{1,k} \le t_k \le w_{2,k}$ [illegible] k [illegible]

$$f_{2n}(w_{1,2n}) \le f \le f_{2n+1}(w_{1,2n+1}), \tag{11a}$$

$$f_{2n}(w_{2,2n}) \ge f \ge f_{2n+1}(w_{2,2n+1}). \tag{11b}$$

3. [illegible] $a_n < 0$ [illegible] $-\frac{1}{2} \le w_{1,k} \le t_k \le w_{2,k}$ [illegible] k [illegible]

$$f_n(w_{1,n}) \le f \le f_n(w_{2,n}). \tag{11c}$$

[illegible]. The proof is straightforward. We prove, for instance, the second inequality of part 3. From $0 \ge w_{2,n} \ge t_n \ge -\frac{1}{2}$ follows

$$\begin{aligned} \frac{a_n}{1+w_{2,n}} &\ge \frac{a_n}{1+t_n} \\ &\vdots \\ f_n(w_{2,n}) &\ge f_n(t_n) = f \end{aligned}$$

and similarly for the other equations. □

LEMMA 3.

1. [illegible] (2) [illegible] $a_n > 0$ [illegible] $\{a_{2n+1}\}_{n\in\mathbb{N}}$ [illegible] $\{a_{2n}\}_{n\in\mathbb{N}}$ [illegible] $\{t_{2n}\}_{n\in\mathbb{N}}$ [illegible] $\{t_{2n+1}\}_{n\in\mathbb{N}}$ [illegible]
2. [illegible] k [illegible] $\ell, a_{2k+1} > a_{2\ell}$ [illegible] k [illegible] ℓ [illegible] (2) [illegible] $t_{2k} \ge t_{2\ell+1}$

[illegible]. Since $a_{2n+3} \le a_{2n+1}$ and $a_{2n+4} \ge a_{2n+2}$, we find that

$$\begin{aligned} f_2^{(2n+2)}(0) = \frac{a_{2n+3}}{1+a_{2n+4}} &\le f_2^{(2n)}(0) = \frac{a_{2n+1}}{1+a_{2n+2}} \\ &\vdots \\ f_k^{(2n+2)}(0) &\le f_k^{(2n)}(0), \qquad k > 2. \end{aligned}$$

In the limit the last inequality remains true, yielding $t_{2n+2} \le t_{2n}$. The case $t_{2n+3} \ge t_{2n+1}$ follows in the same way, and we proceed to part 2. Let us first prove the statement for $k = \ell$. Since $a_{2k+1} > a_{2k+2}, a_{2k+2} < a_{2k+3}$ and so on, we have

$$\begin{aligned} f_2^{(2k)}(0) &> f_2^{(2k+1)}(0) \\ &\vdots \\ f_s^{(2k)}(0) &> f_s^{(2k+1)}(0), \qquad s > 2. \end{aligned}$$

Taking limits, the last inequality becomes $t_{2k} \ge t_{2k+1}$. For $k < \ell$, we use

$$t_{2\ell} \ge t_{2\ell+1} \ge t_{2k+1}$$

and for $k > \ell$, we use

$$t_{2\ell} \ge t_{2k} \ge t_{2k+1},$$

which concludes the proof. □

LEMMA 4. [illegible] a_n [illegible] $-\frac{1}{4} \leq a_n < 0$ [illegible] $\{a_n\}_{n\in\mathbb{N}}$ [illegible] $\{t_n\}_{n\in\mathbb{N}}$ [illegible]. Consider for $k \geq 1$ and $n \geq 1$ the approximant $f_n^{(k)}$ of the tail t_k. Since $0 \leq 1 + a_{k+n+1} \leq 1 + a_{k+n}$, we find with $a_{k+n} < 0$ and $a_{k+n-1} < 0$ that

$$\frac{a_{k+n}}{1 + a_{k+n+1}} \leq \frac{a_{k+n-1}}{1 + a_{k+n}}$$

and analogously

$$f_n^{(k+1)} \leq f_n^{(k)}.$$

In the limit this inequality is preserved and becomes

$$t_{k+1} \leq t_k,$$

which concludes the proof. □

2.5. Backward recurrence algorithm. Several algorithms exist for the computation of $f_N(w)$. The easiest to use, because N can be determined concurrently with $f_N(w)$, is the forward recurrence algorithm

$$\begin{aligned} A_{-1} &= 1, \qquad A_0 = 0, \\ B_{-1} &= 0, \qquad B_0 = 1, \\ A_n &= A_{n-1} + a_n A_{n-2}, \qquad n = 1, \ldots, N-1, \\ B_n &= B_{n-1} + a_n B_{n-2}, \qquad n = 1, \ldots, N-1, \\ f_N(w) &= \frac{(1+w)A_{N-1} + a_N A_{N-2}}{(1+w)B_{N-1} + a_N B_{N-2}}. \end{aligned} \tag{12}$$

The most stable, however [5], is the backward recurrence algorithm

$$\begin{aligned} F_{N+1}^{(N)} &= w, \\ F_n^{(N)} &= \frac{a_n}{1 + F_{n+1}^{(N)}}, \qquad n = N, N-1, \ldots, 1, \\ f_N(w) &= F_1^{(N)}. \end{aligned}$$

For the backward recurrence algorithm to be useful in a variable precision context, it must be possible to determine rather easily a priori which approximant to compute, in other words, to decide which N guarantees a prescribed total error upper bound. For the determination of N from the truncation error upper bound, we refer to sections 3.1 and 3.2. The results follow from combining the interval sequence theorem with Lemmas 1–4. Choosing a suitable estimate for the tail t_N is discussed in section 3.3. The rounding error is subsequently controlled by determining a suitable working precision in which to execute the backward recurrence algorithm. The latter is dealt with in detail in section 3.5.

3. Preparation. The realization of a machine implementation of $f(x)$ is a three-step process:

1. When given an argument x for a function f, the evaluation $f(x)$ is often reduced to the evaluation of f for another argument $\tilde{x}$ lying within specified

bounds and for which there exists an easy relationship between $f(x)$ and $f(\tilde{x})$. For instance, for the exponential function in a base β implementation,

$$\exp(x) = \beta^k \exp(\tilde{x}), \qquad \tilde{x} = \operatorname{mod}(x, \ln \beta), \qquad |\tilde{x}| \leq \frac{\ln \beta}{2}.$$

Although the given argument x is known exactly, because it is a given floating-point number, usually the reduced argument $\tilde{x}$ cannot be computed exactly but is subject to a rounding error.

2. After the reduced argument is determined, the mathematical model F for f is constructed and a truncation error comes into play.
3. When implemented, in other words, evaluated, this mathematical model F is also subject to a rounding error.

Finally, the effect of switching from the argument x to the reduced argument $\tilde{x}$ must be taken into account. This introduces a final additional error. Let us now have a look at how all these errors can be controlled.

The issue of argument reduction is a topic in itself and is not the subject of this paper. We mention it here because we want to emphasize that in our implementation this effect is taken into account, by which we mean that the error upper bound we guarantee holds for the entire argument range and not only for the reduced range. Detailed information on argument reduction can be found in [14, 4] and [12]. For each of the functions discussed in detail in section 4, we briefly outline the argument reduction technique used.

For the use of continued fraction approximants in steps 2 and 3 we still need to address the following issues:

- How do we determine N such that the upper bound for the relative error on $f_N(x; w)$ does not exceed a threshold ϵ_T?
- How do we obtain an easily computable tail estimate $w \in V_N$ (or $w_N \in V_N$) for the evaluation of $f_N(x; w)$?

A rounding error analysis of the backward recurrence algorithm is given in section 3.5.

3.1. Determination of N. The a priori Gragg–Warner bound is expressed entirely in terms of the partial numerators a_n and the function's argument x. Except for the result given in [11, p. 151], this cannot be said of the oval sequence or interval sequence bounds. Theorem 1 as well as Theorem 2 assumes that the value sets are given and that convergence sets can be associated with them.

The aim of this section is slightly different from what we have obtained in Theorem 2, where the sets E_n are deduced from the intervals $V_n = [L_n, R_n]$ and the bounds of E_n are formulated in terms of L_n and R_n. Here we want to formulate L_n and R_n in terms of the bounds on a_n in E_n and hence associate intervals V_n with given intervals E_n, instead of the other way around. In Lemma 5 we further specify the bounds L_n and R_n introduced in Theorem 2, for some appropriately chosen convergence sets E_n. We show that the bounds L_n and R_n are in fact tails.

Let $E_n = [b_n, c_n]$ with $-\frac{1}{4} \leq b_n \leq c_n$ and $b_n c_n \geq 0$. The smallest useful E_n equals $[\bigtriangledown(a_n), \bigtriangleup(a_n)]$, where $\bigtriangledown(a_n)$ and $\bigtriangleup(a_n)$ denote the machine representations of the partial numerator a_n obtained using the downward and upward rounding modes, respectively. The usual machine representation of a_n, namely $\bigcirc(a_n)$ which results from round-to-nearest, is contained in such E_n. In case a_n is itself the result of a computation, then the interval E_n can be taken slightly larger. The condition that

b_n and c_n have the same sign means nothing more than that at least sign(a_n) is kept fixed in E_n.

LEMMA 5. [illegible] $\{E_n\}_{n\in\mathbb{N}}$ [illegible] $E_n = [b_n, c_n]$ [illegible] $b_n \geq -\frac{1}{4}$ [illegible] $0 \leq b_n c_n$ [illegible] $\{V_n\}_{n\in\mathbb{N}}$ [illegible] $V_n = [L_n, R_n]$ [illegible] L_n [illegible] R_n [illegible]

$$\begin{aligned} \hat{D} &= \frac{b_1\rfloor}{\lceil 1} + \frac{c_2\rfloor}{\lceil 1} + \frac{b_3\rfloor}{\lceil 1} + \frac{c_4\rfloor}{\lceil 1} + \cdots, \\ \hat{U} &= \frac{c_1\rfloor}{\lceil 1} + \frac{b_2\rfloor}{\lceil 1} + \frac{c_3\rfloor}{\lceil 1} + \frac{b_4\rfloor}{\lceil 1} + \cdots, \end{aligned} \tag{13}$$

$$\begin{aligned} \check{D} &= \frac{b_1\rfloor}{\lceil 1} + \frac{b_2\rfloor}{\lceil 1} + \frac{b_3\rfloor}{\lceil 1} + \frac{b_4\rfloor}{\lceil 1} + \cdots, \\ \check{U} &= \frac{c_1\rfloor}{\lceil 1} + \frac{c_2\rfloor}{\lceil 1} + \frac{c_3\rfloor}{\lceil 1} + \frac{c_4\rfloor}{\lceil 1} + \cdots. \end{aligned} \tag{14}$$

[illegible] $\hat{D}, \check{D}$ [illegible] $\hat{U}, \check{U}$ [illegible] $\hat{D}^{(n)}, \check{D}^{(n)}$ [illegible] $\hat{U}^{(n)}, \check{U}^{(n)}$ [illegible] $b_n \geq 0$

$$\begin{aligned} L_{2j} &= \hat{D}^{(2j)}, \qquad L_{2j-1} = \hat{U}^{(2j-1)}, \\ R_{2j} &= \hat{U}^{(2j)}, \qquad R_{2j-1} = \hat{D}^{(2j-1)}, \end{aligned} \tag{15}$$

[illegible] $b_n \leq 0$

$$L_n = \check{D}^{(n)}, \qquad R_n = \check{U}^{(n)}. \tag{16}$$

[illegible] A sequence of value sets $\{V_n\}_{n\in\mathbb{N}}$ associated with the sequence of convergence sets $\{E_n\}_{n\in\mathbb{N}}$ satisfies

$$\frac{a_n}{1+V_n} \subseteq V_{n-1}, \qquad a_n \in E_n.$$

Hence, if $V_n \subset \mathbb{R}^+$, expressions for the left and right endpoints L_n and R_n of the intervals V_n need to satisfy the condition

$$(1+L_n)R_{n-1} \geq (1+R_n)L_{n-1} \tag{17}$$

and if $V_n \subset \mathbb{R}^-$, they need to satisfy

$$R_{n-1}(1+R_n) \geq L_{n-1}(1+L_n). \tag{18}$$

Let us first write down explicit formulas for suitable bounds L_n and R_n in case we are dealing with a sequence of convergence sets that allow only positive a_n.

Each V_n contains the tail t_n of (2) since $V_n = \overline{V}_n$. The smallest possible value for the tail t_n of (2) is obtained when the partial numerators $a_{n+j} \in E_{n+j}$ with $j \geq 1$ alternatively take the smallest and largest values $b_{n+1} \in E_{n+1}$, $c_{n+2} \in E_{n+2}, \ldots$. A similar result holds for the largest value of the tail which is attained when the a_{n+j} alternatively take the largest and smallest values c_{n+1}, $b_{n+2}, \ldots$. Hence, for the convergence sets $E_n = [b_n, c_n]$, the corresponding value sets V_n are given by $V_n = [L_n, R_n]$ with

$$\begin{aligned} L_n &= \frac{b_{n+1}\rfloor}{\lceil 1} + \frac{c_{n+2}\rfloor}{\lceil 1} + \frac{b_{n+3}\rfloor}{\lceil 1} + \frac{c_{n+4}\rfloor}{\lceil 1} + \cdots, \\ R_n &= \frac{c_{n+1}\rfloor}{\lceil 1} + \frac{b_{n+2}\rfloor}{\lceil 1} + \frac{c_{n+3}\rfloor}{\lceil 1} + \frac{b_{n+4}\rfloor}{\lceil 1} + \cdots. \end{aligned}$$

It is now easy to check that (17) holds.

When the convergence sets allow only negative a_n, then the proof goes as follows. Consider $f_n^{(k)}$. Since $0 \le 1 + b_{k+n} \le 1 + a_{k+n} \le 1 + c_{k+n}$, we find that

$$\frac{b_{k+n-1}}{1+b_{k+n}} \le \frac{a_{k+n-1}}{1+a_{k+n}} \le \frac{c_{k+n-1}}{1+c_{k+n}}$$

and analogously

$$\check{D}_n^{(k)} \le f_n^{(k)} \le U_n^{(k)}.$$

In the limit this inequality is preserved and becomes

$$\check{D}^{(k)} \le t_k \le \check{U}^{(k)}.$$

Again, (18) is easy to check. □

Given the accuracy with which the partial numerators can be obtained, in other words, the width of the convergence sets E_n, we can now determine the bounds L_n and R_n in terms of the bounds on E_n and use these in (10). This way, the right-hand side of (10) can be evaluated for consecutive values of N until the relative error is sufficiently small. Note, however, that L_n and R_n are infinite expressions and hence not computable. The practical computation of $M_k = \max\{|L_k/(1+L_k)|, R_k/(1+R_k)\}$ in the right-hand side of (10) is further detailed in section 3.3.

We also prefer a bound that requires the computation of as few expressions as possible, for instance, either the sequence $\{R_n\}_{n\in\mathbb{N}}$ or the sequence $\{L_n\}_{n\in\mathbb{N}}$ but not both. This can easily be guaranteed when the value sets V_n are small enough to ensure that $L_n R_n \ge 0$. Then in case $L_n \le 0$, the bound $M_n = |L_n/(1+L_n)|$, and in case $L_n \ge 0$, we have $M_n = R_n/(1+R_n)$ since $L_n \le R_n$ in all cases. According to Lemma 2, $L_n R_n \ge 0$ is guaranteed by $b_n c_n \ge 0$ because the sign of R_n and L_n is determined by the sign of either b_n or c_n.

In addition, when $0 \le b = \lim_{n\to\infty} b_n$ and $0 \le c = \lim_{n\to\infty} c_n$, the tails L_n and R_n, respectively, converge to

$$\begin{aligned} \lim_{n\to\infty} L_n = L &= \frac{b-c-1+\sqrt{(c-b)^2+2(c+b)+1}}{2}, \\ \lim_{n\to\infty} R_n = R &= \frac{c-b-1+\sqrt{(c-b)^2+2(c+b)+1}}{2} \end{aligned} \tag{19}$$

with $L \le R$. In case $b \le 0$ and $c \le 0$, we have

$$\begin{aligned} \lim_{n\to\infty} L_n = L &= \frac{-1+\sqrt{4b+1}}{2}, \\ \lim_{n\to\infty} R_n = R &= \frac{-1+\sqrt{4c+1}}{2}. \end{aligned} \tag{20}$$

In section 4 we illustrate that the bound given by (10) combined with the formulas (15) and (16) is as sharp with respect to the true truncation error for the modified approximant $f_N(x;w)$ as the Gragg–Warner bound is when compared to the unmodified approximant $f_N(x;0)$. Both are also of the same a priori form.

3.2. Explicit formula for N. In this section we want to use formula (10) to deduce an explicit formula for N from the truncation error bound, rather than the bound from the knowledge of N. To this end we need to be able to bound the R_n from above by a value $\overline{R}$, from a certain $n_R \leq N$ on, and bound the L_n from below by $\overline{L}$ from a possibly different $n_L \leq N$ on. The continued fractions and convergence sets that satisfy such conditions are discussed in detail in section 4. Sometimes we want to make use of Lemma 6.

LEMMA 6. [illegible] $\{x_n\}_{n\in\mathbb{N}}$ [illegible] $\{y_n\}_{n\in\mathbb{N}}$ [illegible] n $x_n \geq -d$ [illegible] $y_n \geq -d$ [illegible] $d \in \mathbb{R}$ [illegible]

$$\left\{\frac{y_n - x_n}{d + x_n}\right\}_{n\in\mathbb{N}}$$

[illegible]. For fixed ℓ we define the sequence $\{z_n^{(\ell)}\}_{n\in\mathbb{N}}$ with

$$z_n^{(\ell)} = \frac{y_\ell - x_n}{d + x_n}.$$

Because for every ℓ it holds that $y_\ell \geq -d$, the function $(y_\ell - x)/(d+x)$ is an increasing function on $]-d, +\infty[$. Hence the sequence $\{z_n^{(\ell)}\}_{n\in\mathbb{N}}$ is a decreasing sequence and consequently

$$\frac{y_n - x_n}{d + x_n} \geq \frac{y_{n+1} - x_n}{d + x_n} = z_n^{(n+1)} \geq z_{n+1}^{(n+1)} = \frac{y_{n+1} - x_{n+1}}{d + x_{n+1}}. \qquad \square$$

When the R_n and L_n satisfy the required monotonicity properties, we can bound the factor $(R_N - L_N)/(1 + L_N)$ for $N \geq \max(n_R, n_L)$ by $(\overline{R} - \overline{L})/(1 + \overline{L})$. In case the sequence $\{R_n\}_{n\in\mathbb{N}}$ is decreasing and the sequence $\{L_n\}_{n\in\mathbb{N}}$ is increasing, we can take $\overline{R} = R_K$ and $\overline{L} = L_K$ for $K \leq N$. In addition, the product of the factors M_k can then be bounded above by

$$\prod_{k=1}^{N-1} M_k \leq \left(\prod_{k=1}^{K-1} M_k\right) \times M_K^{N-K}, \qquad K \leq N. \tag{21}$$

Although cruder, (21) allows one to determine N explicitly in terms of a given threshold ϵ_T. Taking the logarithm of the new bound (21) for the product $M_1 \cdots M_{N-1}$ in (10) and an imposed threshold ϵ_T results in

$$(N - K)(-\log M_K) \geq \log\left(\frac{R_K - L_K}{1 + L_K}\right) + \sum_{k=1}^{K-1} \log M_k - \log \epsilon_T. \tag{22}$$

How $w \in [L_N, R_N]$ is obtained cheaply (in terms of numerical operations) is discussed together with some practical ways to obtain a computational upper bound for M_k.

3.3. Obtaining M_k and w. Lemmas 1 and 2 show the way to compute an upper bound for M_k. We discuss two cases in somewhat more detail.

When all partial numerators $a_n > 0$, then $L_k \geq 0$ and $M_k = R_k/(1 + R_k)$. Let us denote the jth approximants of R_k and L_k by $R_{k,j}$ and $L_{k,j}$, respectively, and use part 1 of Lemma 1 (the correct notation for these approximants is actually $R_{k,j}(0)$ and $L_{k,j}(0)$, but we drop the suffix (0) to further simplify the notation). Because

the tails R_k and L_k themselves cannot be computed, odd-numbered approximants $R_{k,2j+1} \geq R_k$ and an even-numbered approximant $L_{N,2j} \leq L_N$ are computed. In practice, for instance, formula (10) becomes

$$\frac{|f(x) - f_N(x;w)|}{|f(x)|} \leq \frac{R_{N,2j+1} - L_{N,2j}}{1 + L_{N,2j}} \prod_{k=1}^{N-1} \frac{R_{k,2j+1}}{1 + R_{k,2j+1}}, \qquad w \in [L_N, R_N]. \tag{23}$$

Here $R_{k,2j+1}$ equals in turn $\hat{U}_{2j+1}^{(2\ell)}$ or $\hat{D}_{2j+1}^{(2\ell+1)}$ depending on whether k is even ($k = 2\ell$) or odd ($k = 2\ell+1$). When replacing the odd-numbered approximants $R_{k,2j+1}$ in (23) by approximate values $\breve{R}_{k,2j+1}$ computed in rounded arithmetic, one should be careful not to violate the condition $\breve{R}_{k,2j+1} \geq R_k$.

When all $a_n < 0$, then $M_k = |L_k/(1 + L_k)|$. If, in addition, the sequences $\{a_n\}_{n\in\mathbb{N}}, \{b_n\}_{n\in\mathbb{N}}$ and $\{c_n\}_{n\in\mathbb{N}}$ from Lemma 5 decrease, then the lower bound for L_k given in part 2 of Lemma 1 can be used to obtain an upper bound for M_k, while the factor $(R_N - L_N)/(1 + L_N)$ can be bounded by (here we keep the suffix (w) because it varies between R_N and L_N)

$$\frac{R_N - L_N}{1 + L_N} \leq \frac{R_{N,j}\left(\frac{-1+\sqrt{4a_{N+1}+1}}{2}\right) - L_{N,j}(-\frac{1}{2})}{1 + L_{N,j}(-\frac{1}{2})}.$$

This results for (10) in

$$\frac{|f(x) - f_N(x;w)|}{|f(x)|} \leq \frac{R_{N,j}\left(\frac{-1+\sqrt{4a_{N+1}+1}}{2}\right) - L_{N,j}(-\frac{1}{2})}{1 + L_{N,j}(-\frac{1}{2})} \prod_{k=1}^{N-1} \left|\frac{L_{k,j}(-\frac{1}{2})}{1 + L_{k,j}(-\frac{1}{2})}\right|,$$

$$w \in [L_N, R_N]. \tag{24}$$

Similar bounds can be given when the sequences $\{a_n\}_{n\in\mathbb{N}}, \{b_n\}_{n\in\mathbb{N}}$, and $\{c_n\}_{n\in\mathbb{N}}$ are increasing.

The simplest choice for w, in case of a limit-periodic continued fraction with $\lim_{n\to\infty} a_n = \tilde{a} < \infty$, is given by

$$w = \frac{-1 + \sqrt{4\tilde{a} + 1}}{2}. \tag{25}$$

The question is, of course, whether this w belongs to $V_N = [L_N, R_N]$. For the continued fractions discussed in section 4, we indicate how this can be assured. It comes down to choosing the convergence sets E_n appropriately, by which we mean that the sequence of value sets $\{V_n\}_{n\in\mathbb{N}}$ has a nonempty intersection. When choosing the E_n too small, such that the sequence of value sets $\{V_n\}_{n\in\mathbb{N}}$ has an empty intersection, it may take some additional computations to obtain a suitable $w = w_N \in V_N$.

So in the ideal situation the sets V_n are small enough to bring the truncation error bound (10) down as much as possible, while at the same time the sets V_n are large enough to allow a nonempty intersection.

3.4. Fractions with positive and negative partial numerators. The condition that (2) has either only positive or only negative partial numerators a_n can be

relaxed for $n < N$, as long as it is satisfied from N on. In that case Lemma 2, which gives us a tail estimate, still remains valid, while Lemma 5 is adapted as follows.

LEMMA 7. [illegible] $\{E_n\}_{n\in\mathbb{N}}$ [illegible] $E_n = [b_n, c_n]$ [illegible] $b_n \geq -\frac{1}{4}$ [illegible] $0 \leq b_n c_n$ [illegible] $\{V_n\}_{n\in\mathbb{N}}$ [illegible] $V_n = [L_n, R_n]$ [illegible]

$$L_n = \sum_{k=1}^{\infty} \frac{\beta_{n+k}}{\left\lfloor 1 \right\rfloor}, \qquad R_n = \sum_{k=1}^{\infty} \frac{\gamma_{n+k}}{\left\lfloor 1 \right\rfloor},$$

[illegible] β_{n+k} [illegible] γ_{n+k} [illegible] k [illegible] $a_{n+1}\cdots a_{n+k-1} > 0$ [illegible] k [illegible] $a_{n+1}\cdots a_{n+k-1} < 0$ [illegible] $\beta_{n+k} = b_{n+k}$ [illegible] $\gamma_{n+k} = c_{n+k}$ [illegible] $\beta_{n+k} = c_{n+k}$ [illegible] $\gamma_{n+k} = b_{n+k}$ [illegible] The rule dictating whether β_{n+k} and γ_{n+k} equal either b_{n+k} or c_{n+k}, where $a_{n+k} \in [b_{n+k}, c_{n+k}]$, comes from the following considerations:

- Irrespective of the sign of a_{n+k} we find that the inequalities $b_{n+k} \leq a_{n+k} \leq c_{n+k}$ are flipped around by the division and result in

$$\frac{1}{1+c_{n+k}} \leq \frac{1}{1+a_{n+k}} \leq \frac{1}{1+b_{n+k}}.$$

- For every negative a_{n+k-1} we know that $|c_{n+k-1}| \leq |a_{n+k-1}| \leq |b_{n+k-1}|$ and hence the inequalities are flipped around once more to

$$\frac{c_{n+k-1}}{1+c_{n+k}} \geq \frac{a_{n+k-1}}{1+a_{n+k}} \geq \frac{b_{n+k-1}}{1+b_{n+k}}.$$

Hence the number of divisions plus the number of negative partial numerators encountered before one reaches a_{n+k} determines whether the largest and smallest fraction values are attained with a_{n+k} replaced by either b_{n+k} or c_{n+k}. □

When the restriction that (2) has either only positive or only negative partial numerators a_n holds from a certain $n \geq M$ on, one can also proceed as follows. A truncation error bound for $|f - f_N(w)|$ can be obtained from a truncation error bound for $|f^{(M)} - f^{(M)}_{N-M}(w)|$ by means of the relation [10, p. 420]

$$f - f_N(w) = \frac{(-1)^M a_1 \cdots a_M}{B_{M-1}^2 \left(h_M + f^{(M)}\right)\left(h_M + f^{(M)}_{N-M}(w)\right)} (f^{(M)} - f^{(M)}_{N-M}(w)). \tag{26}$$

Here the sequence $\{h_k\}_{k\in\mathbb{N}}$ is the so-called critical tail sequence given by

$$\begin{aligned} h_1 &= 1, \\ h_k &= 1 + \frac{a_k}{\left\lfloor 1 \right\rfloor} + \ldots + \frac{a_2}{\left\lfloor 1 \right\rfloor}, \qquad k \geq 2, \end{aligned}$$

and B_k is the kth denominator of (2), which can be computed by means of (12). From (26) we obtain the estimate

$$|f - f_N(w)| \leq \frac{|a_1 \cdots a_M|}{|B_{M-1}|^2 \min^2\left(|L_M + h_M|, |R_M + h_M|\right)} \left| f^{(M)} - f^{(M)}_{N-M}(w) \right|.$$

3.5. Rounding error. When implementing $f_N(w)$, we need to take into account that each basic operation $* \in \{+, -, \times, \div\}$ is being replaced by a machine operation $\circledast \in \{\oplus, \ominus, \otimes, \oslash\}$ and hence is subject to a relative error of at most $\frac{1}{2}$ `ulp`. Also each partial numerator a_n needs to be converted to a machine number $\breve{a}_n$, hence entailing a relative rounding error ϵ_n given by

$$\breve{a}_n = a_n(1 + \epsilon_n).$$

Here $|\epsilon_n|$ is mostly bounded by $\frac{1}{2}$ `ulp` if $a_n = a_n(x)$ is not a compound expression. Otherwise $|\epsilon_n|$ may be somewhat larger.

Without loss of generality, we assume that $w \in V_N$ is a machine number estimating t_N. So we need only take into account the distance of w to t_N, which is present in the truncation error via the parameters C_N and r_N determining V_N. We do not have to take into account an additional rounding error on w representing t_N.

When executing the backward recurrence, each computed $\breve{F}_n^{(N)}, n = N, \ldots, 1$, differs from the true $F_n^{(N)}$ by a rounding error $\epsilon_n^{(N)}$:

$$\begin{aligned}
\breve{F}_{N+1}^{(N)} &= w, \qquad \epsilon_{N+1}^{(N)} = 0, \\
\breve{F}_n^{(N)} &= \breve{a}_n \oslash (1 \oplus \breve{F}_{n+1}^{(N)}), \qquad n = N, \ldots, 1, \\
&= \frac{\breve{a}_n}{1 + \breve{F}_{n+1}^{(N)}}(1 + \delta_n) \\
&= F_n^{(N)}(1 + \epsilon_n^{(N)}), \\
\breve{F}_1^{(N)} &= F_1^{(N)}(1 + \epsilon_1^{(N)}).
\end{aligned}$$

Here δ_n is the relative rounding error introduced in step n of the algorithm. The main question is, How large is $|\epsilon_1^{(N)}|$? This question is answered in Lemma 8 and Theorem 3, the latter being a slight generalization of a result proved in [9] for which we omit the simple proof. Let us introduce the notation

$$\gamma_n^{(N)} = F_{n+1}^{(N)}/(1 + F_{n+1}^{(N)}), \qquad n = 1, \ldots, N.$$

LEMMA 8. $\{V_n\}_{n=1}^{\infty}$ (2) $F_{N+1}^{(N)} = w \in V_N$ $1 \le n \le N$

$$|\gamma_n^{(N)}| = \left| \frac{F_{n+1}^{(N)}}{1 + F_{n+1}^{(N)}} \right| \le M = \max_{n=1,\ldots,N} M_n.$$

THEOREM 3. $F_{N+1}^{(N)} = w$ $n = 1, \ldots, N$

$$\begin{aligned}
|\epsilon_n| &\le \epsilon \ \texttt{ulp}, \\
|\delta_n| &\le \delta \ \texttt{ulp}, \\
|\gamma_n^{(N)}| &\le M.
\end{aligned}$$

β t

$$\beta^{-t+1}\left(1 + M(1 + 2\epsilon + 2\delta)\frac{M^{N-1} - 1}{M - 1}\right) < 1.$$

$|\epsilon_1^{(N)}|$

$$|\epsilon_1^{(N)}| \leq \epsilon_R := \frac{1}{2}(1+2\epsilon+2\delta)\frac{M^N-1}{M-1}\ \texttt{ulp}. \tag{27}$$

If the partial numerators a_n of the continued fraction (2) satisfy $a_n \geq -\frac{1}{4}$, then we know that

- in case all $a_n > 0$ and $w \leq t_n$, the even approximants satisfy $f_{2k}(x;w) \leq f(x)$,
- in case all $a_n < 0$ and $w \leq t_n$, all approximants satisfy $f_n(x;w) \leq f$.

From Theorem 3 we then obtain for even $N = 2n$ with $F_1^{(N)} = f_{2n}(x;w)$ and $w \leq t_N$, or for general N with $F_1^{(N)} = f_N(x;w)$ and $w \leq t_N$,

$$\frac{|f_N(x;w) - \breve{F}_1^{(N)}|}{|f(x)|} = \left|\epsilon_1^{(N)}\right| \frac{|F_1^{(N)}|}{|f(x)|} \leq \epsilon_R.$$

Clearly, ϵ_R is a function of the precision because of its dependence on the `ulp`. When targeting a threshold for ϵ_R, a suitable precision in which to evaluate the continued fraction by means of the backward recurrence algorithm can be computed from this condition. In what follows this precision is called the working precision s, to distinguish it from the user-defined precision t from which the targeted threshold ϵ is usually obtained.

4. Savoring. In the following we distinguish a number of special cases for the partial numerators a_n:

- $\{a_{2n}\}_{n\in\mathbb{N}}$ increasing and $\{a_{2n+1}\}_{n\in\mathbb{N}}$ decreasing with

$$0 < a_n, \qquad 0 < \lim_{n\to\infty} a_{2n} = \tilde{a} = \lim_{n\to\infty} a_{2n+1};$$

- $\{a_n\}_{n\in\mathbb{N}}$ decreasing with $\lim_{n\to\infty} a_n = \tilde{a} \geq 0$;
- $\{a_n\}_{n\in\mathbb{N}}$ decreasing with

$$a_n < 0, \qquad \lim_{n\to\infty} a_n = \tilde{a} \geq -1/4;$$

- $\{a_n\}_{n\in\mathbb{N}}$ increasing with $\lim_{n\to\infty} a_n = \tilde{a} \leq 0$.

These cases cover almost all known limit-periodic continued fractions of the elementary functions and a large number of special functions.

4.1. The case $a_{2n} \nearrow \tilde{a}, a_{2n+1} \searrow \tilde{a}, \tilde{a} > 0$. Let the sequences of positive real numbers $\{a_{2n+1}\}_{n\in\mathbb{N}}$ and $\{a_{2n}\}_{n\in\mathbb{N}}$ be decreasing and increasing, respectively. Let for any $p, q \in \mathbb{N}$,

$$a_{2p+1} > a_{2q}.$$

If the a_n can be obtained from a single rounding, then we can choose the convergence sets $E_n = [b_n, c_n] = [\triangledown(a_n), \triangle(a_n)]$. If the a_n are more complicated, then we choose $[b_n, c_n]$ somewhat larger. In both cases the intervals $V_n = [L_n, R_n]$ with L_n and R_n given by (15) are the corresponding value sets, and the bounds (10) and (23) hold.

In addition, we now investigate the monotonicity properties of the sequences $\{L_n\}_{n\in\mathbb{N}}$ and $\{R_n\}_{n\in\mathbb{N}}$. To this end we reconsider the continued fractions $\hat{D}$ and $\hat{U}$ defined in (13) with tails $\hat{D}^{(n)}$ and $\hat{U}^{(n)}$.

Since the sequences $\{\triangledown(a_{2n})\}_{n\in\mathbb{N}}$ and $\{\triangle(a_{2n+1})\}_{n\in\mathbb{N}}$ are, respectively, increasing and decreasing as a consequence of the monotonicity of the rounding functions $\triangledown$

and $\triangle$, Lemma 3 guarantees that the even-numbered tails $\hat{U}^{(2n)} = R_{2n}$ and the odd-numbered tails $\hat{U}^{(2n+1)} = L_{2n+1}$ form a decreasing and increasing sequence, respectively, and that for any $p, q \in \mathbb{N}$, $\hat{U}^{(2q+1)} \leq \hat{U}^{(2p)}$. The same conclusion holds for the tails $\hat{D}^{(2n)} = L_{2n}$ and $\hat{D}^{(2n+1)} = R_{2n+1}$. If the interval $[b_n, c_n]$ is somewhat larger, the same conclusion holds if we can ensure that the sequences $\{b_{2n}\}_{n\in\mathbb{N}}$ and $\{c_{2n+1}\}_{n\in\mathbb{N}}$ are, respectively, increasing and decreasing.

Following the line of proof of Lemma 3, we can show in addition that

$$\begin{aligned} R_{2n+1} &\leq R_{2k}, \qquad k, n \in \mathbb{N}, \\ L_{2n+1} &\leq L_{2k}, \qquad k, n \in \mathbb{N}. \end{aligned}$$

Hence

$$\bigcap_{k=1}^{\infty} V_{2k-1} = L, \qquad \bigcap_{k=1}^{\infty} V_{2k} = R$$

with L and R given by (19). When choosing $E_n = [b_n, c_n]$ equal to

$$[b_{2k}, c_{2k}] = [a_{2k}, \tilde{a}], \qquad [b_{2k-1}, c_{2k-1}] = [\tilde{a}, a_{2k-1}], \qquad k \geq 1, \tag{28}$$

inequalities (10) and (23) remain valid, while, moreover,

$$L = L_{2n} = R_{2n-1} = R = \frac{1}{2}(-1 + \sqrt{4\tilde{a} + 1}).$$

Then, mathematically, a suitable $w \in V_N$ is easy to determine since for all N now $w = -\frac{1}{2} + \sqrt{\tilde{a} + \frac{1}{4}} \in V_N$. In practice, the above is applied with $\tilde{a}$ and a_{2k-1} in E_{2k-1} replaced by $\bigtriangledown(\tilde{a})$ and $\triangle(a_{2k-1})$, respectively, and $\tilde{a}$ and a_{2k} in E_{2k} replaced by $\triangle(\tilde{a})$ and $\bigtriangledown(a_{2k})$ or similar lower and upper bounds. Then L and R differ slightly, because in an implementation they are given by (19) with $b = \bigtriangledown(\tilde{a})$ and $c = \triangle(\tilde{a})$ or similarly. Choosing w alternately equal to L or R depending on whether N is odd or even is a valid choice (care must be taken during the actual computation of w that it is rounded in the appropriate direction, with "appropriate" meaning inside V_N).

Examples of elementary functions illustrating this case include the hyperbolic arcsine

$$\frac{\operatorname{asinh}(x)}{x\sqrt{1+x^2}} = \cfrac{1}{1} + \sum_{n=1}^{\infty} \cfrac{2\lfloor\frac{n+1}{2}\rfloor(2\lfloor\frac{n+1}{2}\rfloor - 1)x^2/(4n^2-1)}{1}, \qquad \lim_{n\to\infty} a_n(x) = \frac{x^2}{4},$$

the hyperbolic arctangent

$$x \operatorname{atanh}(x) = \cfrac{x^2/(1-x^2)}{1} + \sum_{n=1}^{\infty} \cfrac{2\lfloor\frac{n+1}{2}\rfloor(2\lfloor\frac{n+1}{2}\rfloor - 1)x^2/\left((4n^2-1)(1-x^2)\right)}{1},$$

$$\lim_{n\to\infty} a_n(x) = \frac{x^2}{4(1-x^2)},$$

and the natural logarithm

$$\frac{x}{\ln(x+1)} - 1 = \sum_{n=1}^{\infty} \cfrac{nx/(4n-2)}{1} + \cfrac{nx/(4n+2)}{1}, \qquad \lim_{n\to\infty} a_n(x) = x/4.$$

We discuss the natural logarithm in more detail in section 4.2.

Examples of special functions to which the technique applies are the hypergeometric functions

$$ {}_2F_1(a,1;c;x), \qquad a \neq 3/2, \qquad x<0, \qquad \tilde{a} = -x/4. $$

For $a = 3/2$, the technique described in section 4.3 applies.

4.2. Natural logarithm. A useful continued fraction for which the results of section 4.1 hold is

$$ f(x) = \frac{x}{\ln(x+1)} - 1 = \sum_{n=1}^{\infty} \frac{nx/(4n-2)\big\rfloor}{\lceil 1} + \frac{nx/(4n+2)\big\rfloor}{\lceil 1}, \qquad x+1 \geq 0. $$

Since $\ln(0) = -\infty$, $\ln(1) = 0$, and for $0 < x < 1$ the relation $\ln(x) = -\ln(1/x)$ holds, we have to consider only the case $x > 1$. For a given floating-point number

$$ x = +d_0 \,.\, d_1 \cdots d_{t-1} \times \beta^{e_x} = +m_x \times \beta^{e_x}, \qquad d_0 \neq 0,\ 0 \leq d_i \leq \beta - 1,\ i = 0, \ldots, t-1, $$

as operand, and a choosen integer $\ell > 0$, we construct the reduced operand $\tilde{x}$,

$$ \begin{aligned} j_x &= \lfloor \log_{\beta^{1/\ell}} m_x, \rfloor, \\ \tilde{x} &= +d_0 \,.\, d_1 \cdots d_{t-1} \times \beta^{-j_x/\ell} - 1, \end{aligned} $$

which satisfies

$$ \ln(x) = \frac{\tilde{x}}{1+\ln(\tilde{x})} + \left(e_x + \frac{j_x}{\ell} \right) \ln(\beta). \tag{29} $$

For $\beta = 2$ and $\ell = 8$, the range of the reduced argument $\tilde{x}$ is then $[1, 2^{1/8}]$.

With $t = 53$ in (1) and $\epsilon_T = 2^{-52} = 1\mathtt{ulp} \approx 2.22 \times 10^{-16}$, we obtain from (23) that $f_N(w)$ with $N = 9$, $w = \frac{1}{2}(-1 + \sqrt{x+1})$, and $2j+1 = 3$ satisfies

$$ \frac{|f(\sqrt[8]{2}-1) - f_9(w)|}{|f(\sqrt[8]{2}-1)|} = 7.98 \times 10^{-17} \leq \frac{R_{9,3} - L_{9,2}}{1 + L_{9,2}} \prod_{k=1}^{8} \frac{R_{k,3}}{1+R_{k,3}} \leq 1.98 \times 10^{-16} < \epsilon_T. $$

This compares nicely to the targeted complexity of 25 operations for double precision. The backward algorithm uses $2N = 18$ operations for the computation of $f_9(w)$, and the functional relationship (29) employs another 6. In the next section the comparison with the state-of-the-art double precision algorithms is even more favorable. For the luxury of having a multiprecision version we have to pay the price of obtaining N from (10) or (22) though, while in a fixed precision implementation N is fixed a priori. Of course we do not have to add this additional work to the complexity count when comparing the algorithm's complexity to that of a fixed precision algorithm.

Because of the specific behavior of the sequences $\{R_n\}_{n\in\mathbb{N}}$ and $\{L_n\}_{n\in\mathbb{N}}$ when E_n is given by (28), we have with K even in (22) that $\overline{R} = R_K$ and $\overline{L} = L$. With $K = 4$ formula (22) also predicts $N = 9$. With $K = 2$ we obtain a slight overestimate of N, namely $N = 10$ instead of $N = 9$:

$$ \frac{|f(\sqrt[8]{2}-1) - f_{10}(w)|}{|f(\sqrt[8]{2}-1)|} \leq \left(\frac{R_{2,3} - L}{1+L} \right) M_1 \left(\frac{R_{2,3}}{1+R_{2,3}} \right)^4 M_2^4 \leq 1.06 \times 10^{-17} < \epsilon_T. $$

When comparing these results to the classical results obtained for $w = 0$ using (8) (where we approximate $f(\sqrt[8]{2}-1)$ in the denominator of the relative error by $f_K(x;0)$ with K even), we see that

$$\frac{|f(\sqrt[8]{2}-1) - f_{11}(0)|}{|f(\sqrt[8]{2}-1)|} = 1.89\times10^{-17} \le \frac{2a_1}{f_K(x;0)} \prod_{k=2}^{11} \frac{\sqrt{1+4a_k}-1}{\sqrt{1+4a_k}+1} = 3.77\times10^{-17} < \epsilon_T.$$

This implies a gain of about 20% in complexity: $N = 9$ is predicted by (22) and (23) while $N = 11$ is predicted by the Gragg–Warner bound (8). From the multitude of experiments we have conducted we can confirm that this gain is typical when comparing the new technique for hardware precisions. In addition, the precision with which w needs to be computed is determined by the width of V_N, which is at least as wide as $R_{N,2j} - L_{N,2j+1}$. When the precision t is increased while maintaining the rather wide sets E_n specified in (28), often w need not be computed to full precision.

Now let us consider $\epsilon_T = 2^{-511} \approx 1.50 \times 10^{-154}$ and slightly refine the sets E_n. Restricting the variation in the partial numerators to that of the double precision rounding of $a_n(x)$ amounts to choosing E_n like $E_n = [a_n(1-\delta), a_n(1+\delta)]$ with $|\delta| \le 2^{-52}$. We find that for the same argument x and with $N = 84$ and $2j+1 = 11$,

$$\frac{|f(\sqrt[8]{2}-1) - f_{84}(w)|}{|f(\sqrt[8]{2}-1)|} \le 5.82 \times 10^{-156} < \epsilon_T, \qquad w \in V_{84}.$$

Here all factors in the upper bound (23) are computed only in double precision. At the same time, the Gragg–Warner bound predicts $N = 94$. Hence the newly developed technique offers a gain of 20 high precision computations ($\Delta N = 94 - 84$). The practical problem here is to find $w_N \in V_N$. A possible choice is $w_{84} = L_{84,11}$ costing 22 double-precision basic operations ($2j+1 = 11$). This choice for w is valid only if $L_N \le L_{N,2j+1}$ belongs to V_N, meaning that $L_{N,2j+1} \le R_N$.

4.3. The case $a_n \searrow \tilde{a} \ge 0$. Let each partial numerator a_n satisfy $\tilde{a} \le a_n$ where the sequence $\{a_n\}_{n\in\mathbb{N}}$ is a decreasing sequence of positive numbers and $0 \le \tilde{a} = \lim_{n\to\infty} a_n$. So all a_n are strictly positive. For $E_n = [\triangledown(a_n), \triangle(a_n)]$ the corresponding value sets are $V_n = [L_n, R_n]$ with L_n and R_n given by (15). For these V_n the bounds (10) and (23) hold. To guarantee that (25) is in $[L_N, R_N]$, we consider slightly larger convergence sets E_n.

For the convergence sets $E_n = [\tilde{a}, a_n]$ the value sets $V_n = [L_n, R_n]$ are bounded by

$$L_n = \cfrac{\tilde{a}}{\left\lceil 1\right\rfloor} + \cfrac{a_{n+2}}{\left\lceil 1\right\rfloor} + \cfrac{\tilde{a}}{\left\lceil 1\right\rfloor} + \cfrac{a_{n+4}}{\left\lceil 1\right\rfloor} + \cdots,$$
$$R_n = \cfrac{a_{n+1}}{\left\lceil 1\right\rfloor} + \cfrac{\tilde{a}}{\left\lceil 1\right\rfloor} + \cfrac{a_{n+3}}{\left\lceil 1\right\rfloor} + \cfrac{\tilde{a}}{\left\lceil 1\right\rfloor} + \cdots.$$

Following the line of proof of Lemma 3 we find that the sequence $\{R_n\}_{n\in\mathbb{N}}$ is decreasing and that $\{L_n\}_{n\in\mathbb{N}}$ is increasing. Now

$$\bigcap_{k\in\mathbb{N}} V_k = [L, R]$$

with $L = R = (-1 + \sqrt{4\tilde{a}+1})/2$. Hence we can choose for all N the tail estimate $w = (-1 + \sqrt{4\tilde{a}+1})/2$. In practice this is applied to $E_n = [\triangledown(\tilde{a}), \triangle(a_n)]$, replacing

in L_n, R_n, and (22) every occurrence of $\tilde{a}$ by $\bigtriangledown(\tilde{a})$ and every occurrence of a_n by $\bigtriangleup(a_n)$. Use of these roundings makes $L < R$ and, consequently, all $w \in [L, R]$ a valid choice.

With $\tilde{a} = 0$, we have $E_k = [0, a_k], R_k = a_{k+1}$, and $L_k = 0$. The truncation error bound simplifies to

$$\frac{|f(x) - f_N(x; w)|}{|f(x)|} \leq a_{N+1} \prod_{k=1}^{N-1} \frac{a_{k+1}}{1 + a_{k+1}}. \tag{30}$$

Here $w = 0$ can be used, which is a more classical choice.

Examples taken from the class of functions satisfying the conditions imposed in this section involve the arccosine function

$$\frac{x\ \mathrm{acos}(x)}{\sqrt{1-x^2}} = \left\lfloor \frac{1}{1} \right\rfloor + \sum_{n=1}^{\infty} \left\lfloor \frac{n^2(1-x^2)/\left((2n-1)(2n+1)x^2\right)}{1} \right\rfloor, \qquad \lim_{n\to\infty} a_n(x) = \frac{(1-x^2)}{4x^2},$$

the arctangent function

$$x\ \arctan(x) = \left\lfloor \frac{x^2}{1} \right\rfloor + \sum_{n=1}^{\infty} \left\lfloor \frac{n^2x^2/(4n^2-1)}{1} \right\rfloor, \qquad \lim_{n\to\infty} a_n(x) = \frac{x^2}{4},$$

and the arccotangent function

$$\frac{\mathrm{acot}(x)}{x} = \left\lfloor \frac{x^{-2}}{1} \right\rfloor + \sum_{n=1}^{\infty} \left\lfloor \frac{n^2x^{-2}/(4n^2-1)}{1} \right\rfloor, \qquad \lim_{n\to\infty} a_n(x) = \frac{x^{-2}}{4}.$$

The arctangent is discussed in detail in section 4.4. Examples where (30) applies include the hyperbolic tangent function

$$x\ \tanh(x) = \left\lfloor \frac{x^2}{1} \right\rfloor + \sum_{n=1}^{\infty} \left\lfloor \frac{x^2/(4n^2-1)}{1} \right\rfloor$$

and the exponential function

$$\frac{2x}{\exp(x)-1} = 2 - x + \left\lfloor \frac{x^2/6}{1} \right\rfloor + \sum_{n=2}^{\infty} \left\lfloor \frac{x^2/\left(4(4n^2-1)\right)}{1} \right\rfloor.$$

Also, several special functions belong to this class, including the confluent hypergeometric functions ${}_1F_1(1; c; x)$.

4.4. Arctangent function. The conditions imposed in section 4.3 hold for the following continued fraction representation of the arctangent function:

$$\arctan(x) = \left\lfloor \frac{x}{1} \right\rfloor + \sum_{n=1}^{\infty} \left\lfloor \frac{n^2x^2/(4n^2-1)}{1} \right\rfloor, \qquad \lim_{n\to\infty} a_n(x) = \frac{x^2}{4}.$$

Since $\arctan(-x) = -\arctan(x)$, we need to focus on only positive arguments. Moreover, by using the fact that

$$2-\sqrt{3} < x \le 1 \Rightarrow \left|\frac{x\sqrt{3}-1}{x+\sqrt{3}}\right| \le 2-\sqrt{3}, \qquad \arctan(x) = \arctan\left(\frac{x\sqrt{3}-1}{x+\sqrt{3}}\right) + \frac{\pi}{6},$$

$$1 < x \le 2+\sqrt{3} \Rightarrow \left|\frac{\sqrt{3}-x}{x\sqrt{3}+1}\right| \le 2-\sqrt{3}, \qquad \arctan(x) = -\arctan\left(\frac{\sqrt{3}-x}{x\sqrt{3}+1}\right) + \frac{\pi}{3},$$

$$2+\sqrt{3} < x \Rightarrow 1/x < 2-\sqrt{3}, \qquad \arctan(x) = -\arctan(1/x) + \frac{\pi}{2},$$

one can reduce the argument of $\arctan(x)$ to the interval $[0, 2-\sqrt{3}]$. For $x \in [0, 2-\sqrt{3}]$ the value $\arctan(x) \in [0, \pi/12]$. The sequence of partial numerators is a decreasing positive sequence from the very first partial numerator on.

With $t = 53$ in (1) and $\epsilon_T = 2^{-52} = 1\mathtt{ulp}$, we obtain from (23) that $f_N(w)$ with $N = 8$, $w = \frac{1}{2}(-1+\sqrt{x^2+1})$, and $2j+1 = 3$ satisfies

(31)
$$\frac{|f(2-\sqrt{3}) - f_8(w)|}{|f(2-\sqrt{3})|} = 4.74 \times 10^{-17} \le \frac{R_{8,3}-L_{8,2}}{1+L_{8,2}} \prod_{k=1}^{7} \frac{R_{k,3}}{1+R_{k,3}} \le 5.26 \times 10^{-17} < \epsilon_T.$$

When using (22) with $K = 3$ we find $N = 9$. With $K = 4$ we obtain again $N = 8$. The upper bound for the truncation error is then estimated by

$$\frac{|f(2-\sqrt{3}) - f_8(w)|}{|f(2-\sqrt{3})|} \le \left(\frac{R_{8,3}-L_{8,2}}{1+L_{8,2}}\right) M_1\ M_2\ M_3\ M_4^4 < \epsilon_T.$$

Comparing this to the Gragg–Warner bound for the classical approximant $f_N(0)$, we find

(32)
$$\frac{|f(2-\sqrt{3}) - f_9(0)|}{|f(2-\sqrt{3})|} = 2.18 \times 10^{-16} \le \epsilon_T,$$

$$\frac{|f(2-\sqrt{3}) - f_{11}(0)|}{|f(2-\sqrt{3})|} \le \frac{2a_1}{f_K(0)} \prod_{k=2}^{11} \frac{\sqrt{1+4a_k}-1}{\sqrt{1+4a_k}+1} = 7.56 \times 10^{-18} < \epsilon_T.$$

Hence the new technique delivers $N = 8$ while the traditional Gragg–Warner bound estimates $N = 11$, which is an overestimate of N in (32) by 2. Again the use of a nonzero tail estimator in (31) together with the newly developed bounds offers a clear advantage.

Choosing $t = 113$ (as in the quadruple precision supported by Sun) and $\epsilon_T = 2^{-112} \approx 1.93 \times 10^{-34}$, we find with $N = 18, w = \frac{1}{2}(-1+\sqrt{x^2+1})$, and $2j+1 = 3$

$$\frac{|f(2-\sqrt{3}) - f_{18}(w)|}{|f(2-\sqrt{3})|} = 5.65 \times 10^{-39} \le \frac{R_{18,3}-L_{18,2}}{1+L_{18,2}} \prod_{k=1}^{17} \frac{R_{k,3}}{1+R_{k,3}} \le 3.35 \times 10^{-35} < \epsilon_T.$$

The traditional Gragg–Warner bound for use with $w = 0$ delivers $N = 21$ while

$$\frac{|f(2-\sqrt{3}) - f_{20}(0)|}{|f(2-\sqrt{3})|} = 9.35 \times 10^{-36}.$$

Again, this is a clear advantage of a nonzero choice for w over the standard $w = 0$.

4.5. The case $a_n \searrow \tilde{a} \geq -\frac{1}{4}$. For simplicity we take all $a_n < 0$. When this is only the case from an index $n = M$ on, then the technique described in section 3.4 can be used to estimate the truncation error from the beginning of the fraction, given the truncation error from a_M on. Since all $L_n \geq -\frac{1}{2}$, the error bound (10) remains valid.

We choose $E_n = [a_{n+1}, a_n]$ and compute $V_n = [L_n, R_n]$ with L_n and R_n given by (16). Apparently $L_n = t_{n+1}$ and $R_n = t_n$. The sequences $\{R_n\}_{n\in\mathbb{N}}$ and $\{L_n\}_{n\in\mathbb{N}}$ are decreasing with

$$\lim_{n\to\infty} R_n = \frac{-1+\sqrt{4\tilde{a}+1}}{2} = \lim_{n\to\infty} L_n.$$

Combining part 2 of Lemma 1 and part 3 of Lemma 2 provides the computable bounds

$$L_{n,j}\left(\frac{-1+\sqrt{4\tilde{a}+1}}{2}\right) \leq L_n \leq R_n \leq R_{n,j}\left(\frac{-1+\sqrt{4a_{n+1}+1}}{2}\right), \qquad j \geq 1.$$

Under the condition that $L_N \leq R_{N,j}(\frac{1}{2}(-1+\sqrt{4\tilde{a}+1}))$, a valid choice for $w_N \in V_N$ is given by $w_N = R_{N,j}(\frac{1}{2}(-1+\sqrt{4\tilde{a}+1}))$. Computational remarks similar to those in sections 3.3 and 4.1 apply, concerning the effect of rounding errors on the computed $L_{n,j}, R_{n,j}$, and w_N.

Special functions enjoying a limit-periodic continued fraction with $a_n \geq -\frac{1}{4}$ and $\tilde{a} = -\frac{1}{4}$ are

(33)

$$\mathrm{E}_m(x) = \cfrac{\exp(-x)/(x+m)}{1} + \sum_{n=2}^{\infty} \cfrac{\frac{-n(m+n-2)}{(x+m+2(n-1))(x+m+2(n-2))}}{1}, \quad m \neq 1, x \neq 0,$$

$$\operatorname{erfc}(x) = \cfrac{\frac{\exp(-x^2)}{2\sqrt{\pi}x(2x^2+1)}}{1} + \sum_{n=2}^{\infty} \cfrac{\frac{-2n(2n-1)}{(2x^2+1+4n)(2x^2+1+4(n-1))}}{1}, \quad x \neq 0.$$

As an illustration of the technique we compute the function $f(x)$, defined by the first tail of (33), up to 80 bits accuracy, in other words, with $\epsilon_T = 2^{-79} \approx 1.65 \times 10^{-24}$. This function has only negative partial numerators. With E_n and V_n as given above and $x = 2$, the bound (10) is less than ϵ_T for $N \geq 59$. For $j = 12$ we obtain in addition that

$$t_{N+1} = L_N < L_{N,j}\left(\frac{-1+\sqrt{4a_{N+2}+1}}{2}\right) < R_{N,j}\left(-\frac{1}{2}\right) < R_N = t_N$$

and hence that all w satisfying

$$L_{N,j}\left(\frac{-1+\sqrt{4a_{N+2}+1}}{2}\right) < -3.7626 \times 10^{-1} \leq w \leq -3.7527 \times 10^{-1} < R_{N,j}\left(-\frac{1}{2}\right)$$

are valid choices for the approximation of $f(x)$ by $f_N(x; w)$, since they are guaranteed to belong to V_N.

4.6. The case $a_n \nearrow \tilde{a} \leq 0$. When the partial numerators a_n are negative and increasing with $\lim_{n\to\mathbb{N}} a_n = \tilde{a}$, we can choose $E_n = [a_n, \tilde{a}]$. The value sets V_n are then given by $V_n = [L_n, R_n] = [t_n, \frac{-1+\sqrt{4\tilde{a}+1}}{2}]$, where it can be proved as in Lemma

4 that the sequence $\{t_n\}_{n\in\mathbb{N}}$ is increasing. Following the line of proof of part 2 of Lemma 1, we find that

$$\frac{-1+\sqrt{4a_{n+1}+1}}{2} \le L_n.$$

Hence the truncation error can be bounded by

$$|f(x) - f_N(x;w)| \le \frac{1-\sqrt{4a_{N+1}+1}}{1+\sqrt{4a_{N+1}+1}} \prod_{k=1}^{N-1} \frac{1-\sqrt{4a_{k+1}+1}}{1+\sqrt{4a_{k+1}+1}}, \qquad w \in V_N.$$

When $\tilde{a} = 0$ this is very similar to the Gragg–Warner bound on the real line ($\phi = \arg(x) = 0$). Note that $w = 0$ is a valid choice if $\tilde{a} = 0$. This truncation error bound applies, for instance, to the function $f(x)$ given by

$$f(x) = \frac{x}{\tan(x)} - 1 = \sum_{n=1}^{\infty} \cfrac{-x^2/(4n^2-1)}{1}.$$

4.7. Conclusion. The truncation error bounds obtained here for $f_N(x;w)$ are more general than the ones given in [8] or [2] which contain more stringent conditions on the a_n that are not as easily satisfied for all n, and the ones found in [10] where $C_N = \tilde{w}$ with $\tilde{w}$ given by (7) is recommended for (9). Other truncation error bounds that can be found in the literature [7, 6] hold mainly for $w = 0$.

When classifying all these bounds with respect to sharpness, we have to distinguish between bounds for the unmodified approximant $f_N(x;0)$ and bounds for the modified approximant $f_N(x;w)$. As can be expected, the a posteriori error bounds, which take advantage of the information contained in continued fraction approximants computed for smaller values of N, are the most accurate truncation error bounds for $f_N(x;0)$. The newly developed a priori error bound is to the modified approximant $f_N(x;w)$ what the Gragg–Warner bound is to the unmodified approximant $f_N(x;0)$. Both are of comparable quality, which is precisely our goal.

An analysis of the complexity of the proposed implementation (counting additions/subtractions and multiplications/divisions) illustrates that the multiprecision continued fraction method presented here even compares well to the state-of-the-art fixed precision techniques which are in use. When choosing the radix $\beta = 2$ and the precision $t = 53$ (standard IEEE 754 double precision), the proposed continued fraction model allows one to evaluate the elementary functions in about 20 to 25 operations, including the computation of the square root modification $w = \tilde{w}$, which compares favorably to the current double-precision implementations. Moreover, the proposed technique is generic in the sense that it can be used for any user-definable precision.

Acknowledgments. Thanks are due to Stefan Becuwe and Johan Vervloet for implementing a lot of our ideas in either Maple or C++. The insight gained from the numerical illustrations guided us in the selection of the material for this paper and will assist us with the implementation of a reliable special function library in the future.

REFERENCES

[1] G.A. Baker, Jr., and P. Graves-Morris, *Padé Approximants*, 2nd ed., Cambridge University Press, Cambridge, UK, 1996.

[2] Ch. Baltus and W.B. Jones, *Truncation error bounds for modified continued fractions with applications to special functions*, Numer. Math., 55 (1989), pp. 281–307.

[3] B.A. Cipra, *A new testament for special functions?*, SIAM News, 31 (2) (1998), p. 20.
[4] W.J. Cody and W. Waite, *Software Manual for the Elementary Functions*, Prentice–Hall, Englewood Cliffs, NJ, 1980.
[5] W. Gautschi, *Computational aspects of three-term recurrence relations*, SIAM Rev., 9 (1967), pp. 24–82.
[6] W.B. Gragg and D.D. Warner, *Two constructive results in continued fractions*, SIAM J. Numer. Anal., 20 (1983), pp. 1187–1197.
[7] P. Henrici and P. Pfluger, *Truncation error estimates for Stieltjes fractions*, Numer. Math., 9 (1966), pp. 120–138.
[8] L. Jacobsen, *Convergence acceleration and analytic continuation by means of modifications of continued fractions*, in Padé Approximants and Continued Fractions, Kongelige Norske Videnskabers Selskab 1, H. Waadeland and H. Wallin, eds., Universitetsforlaget, Trondheim-Oslo-Bergen-Tromsø, Norway, 1983, pp. 19–33.
[9] W.B. Jones and W.J. Thron, *Numerical stability in evaluating continued fractions*, Math. Comp., 28 (1974), pp. 795–810.
[10] L. Lorentzen, *A priori truncation error bounds for continued fractions*, Rocky Mountain J. Math., 33 (2003), pp. 409–474.
[11] L. Lorentzen and H. Waadeland, *Continued Fractions with Applications*, North–Holland, Amsterdam, 1992.
[12] Sun Microsystems, *fdlibm*, www.netlib.org/fdlibm (1995).
[13] C. Moler, *The tetragamma function and numerical craftsmanship*, MATLAB News & Notes, February (2002).
[14] J.-M. Muller, *Elementary Functions. Algorithms and Implementation*, Birkhäuser Boston, Boston, 1997.
[15] W.J. Thron and H. Waadeland, *Accelerating convergence of limit periodic continued fractions* $K(a_n/1)$, Numer. Math., 34 (1980), pp. 155–170.

SIAM J. SCI. COMPUT.
Vol. 28, No. 4, pp. 1463–1489
© 2006 Society for Industrial and Applied Mathematics

FREQUENCY DOMAIN OPTICAL TOMOGRAPHY BASED ON THE EQUATION OF RADIATIVE TRANSFER*

KUI REN†, GUILLAUME BAL†, AND ANDREAS H. HIELSCHER‡

Abstract. Optical tomography consists of reconstructing the spatial distribution of absorption and scattering properties of a medium from surface measurements of transmitted light intensities. Mathematically, this problem amounts to parameter identification for the equation of radiative transfer (ERT) with diffusion-type boundary measurements. Because they are posed in the phase-space, radiative transfer equations are quite challenging to solve computationally. Most past works have considered the steady-state ERT or the diffusion approximation of the ERT. In both cases, substantial cross-talk has been observed in the reconstruction of the absorption and scattering properties of inclusions. In this paper, we present an optical tomographic reconstruction algorithm based on the frequency-domain ERT. The inverse problem is formulated as a regularized least-squares minimization problem, in which the mismatch between forward model predictions and measurements is minimized. The ERT is discretized by using a discrete ordinates method for the directional variables and a finite volume method for the spatial variables. A limited-memory quasi-Newton algorithm is used to minimize the least-squares functional. Numerical simulations with synthetic data show that the cross-talk between the two optical parameters is significantly reduced in reconstructions based on frequency-domain data as compared to those based on steady-state data.

Key words. optical tomography, photon density waves, equation of radiative transfer, finite volume method, discrete ordinates method, generalized minimal residual algorithm, inverse problems, regularization, numerical optimization

AMS subject classifications. 65N21, 65Z05, 78M25, 78M50

DOI. 10.1137/040619193

1. Introduction.

The inverse problem related to the equation of radiative transfer (ERT) has long been of interest in many fields of applied physics and mathematics. In this problem, one attempts to determine the spatial distribution of parameters (e.g., the spatial distribution of absorption and scattering coefficients) inside a medium from either transmission or reflectance data collected by illuminating the medium from many different directions. Applications include, for instance, astronomy [61], nuclear science [42, 43], and atmospheric science [12]; see the review [50]. Recent advances in this field have been fueled by an increased interest in medical applications. By performing optical measurements on the surface of the skin, optical tomographic techniques attempt to reconstruct the spatial distribution of absorption and scattering coefficients inside the body. These parameters can then be used for many diagnostic purposes [13].

Mathematically, optical tomography reduces to parameter identification problems (inverse problems) for the ERT, also referred to as the linear Boltzmann equation or the transport equation. Most early studies in the field deal with plane-parallel ge-

*Received by the editors November 18, 2004; accepted for publication (in revised form) March 15, 2006; published electronically September 15, 2006. The work of the first and third authors was supported in part by the National Institute of Arthritis and Musculoskeletal and Skin Diseases (grant 2R01 AR046255) and by the National Institute of Biomedical Imaging and Bioengineering (grant R01 EB001900), which are both part of the National Institutes of Health. The work of the second author was supported by NSF grant DMS-0239097 and an Alfred P. Sloan Fellowship.

http://www.siam.org/journals/sisc/28-4/61919.html

†Department of Applied Physics & Applied Mathematics, Columbia University, New York, NY 10027 (kr2002@columbia.edu, gb2030@columbia.edu).

‡Department of Biomedical Engineering and Radiology, Columbia University, New York, NY 10027 (ahh2004@columbia.edu).

ometry [20, 24]. Inversion procedures were also developed in many more general yet still constrained geometries; see, for instance, [3, 42, 43, 61] and the review papers [49, 50]. Because of their geometrical specificities, most of these works do not directly apply to biomedical applications. In the mid-1990s several groups started developing schemes and algorithms that overcame these problems in the framework of the diffusion approximation to the ERT; see, for instance, [4, 34, 57, 63]. Many theoretical and experimental studies have shown that the diffusion approximation was valid only in specific contexts—for instance, when the scattering coefficient is sufficiently large and the absorption coefficient sufficiently small [16, 17, 23, 31]. While this is often the case in biomedical applications there remain many important applications, for which reconstructions based on the ERT are needed. Some examples include brain imaging, where the presence of low-absorbing and low-scattering cerebrospinal fluid leads to nondiffusive light propagation; imaging of joints, where clear synovial fluid has the same effects as cerebrospinal fluid in the brain; and small animal imaging, where small object dimensions (1–2 cm in diameter) invalidate the classical diffusion approximation [30, 32]; see, however, [6, 8] for ways to generalize the diffusion equation in certain situations.

Only quite recently have studies emerged that describe ERT-based reconstruction codes for use in biomedical optical tomography. First, a transport-backtransport method, a nonlinear inversion method, applied to the two-dimensional time-dependent equation of radiative transfer was reported in [18]. New algorithms were developed and experimentally tested for two- and three-dimensional cases using a time-independent ERT in [38, 39, 40, 41]. While these works, which address real-life three-dimensional problems, are an important step towards practical applications, they still suffer from considerable cross-talk between absorption and scattering reconstructions. What we mean by cross-talk is that purely scattering (or purely absorbing) inclusions are often reconstructed with unphysical absorption (or scattering) properties. This behavior is well-understood from the theoretical viewpoint: Different optical distributions inside the medium can lead to the same measurements collected at the surface of the medium [5, 33]. To avoid such cross-talk, which may lead to wrong diagnostics, we need different data. An experimental technique increasingly employed in recent years to obtain additional information is to use frequency domain measurements. In this case the source intensity is modulated (typically between 100–1000 MHz), leading to the propagation of so-called photon density waves. Since frequency domain measurements provide information about both the phase and the intensity of the waves (and not only the intensity as in steady-state measurements), it is expected that the frequency-domain techniques will allow for better separation of absorption and scattering effects [5, 48]. Numerical reconstructions based on the frequency-domain ERT, however, have not yet been developed in the literature. This is the major motivation for the present work.

We will propose in the following sections a numerical procedure that solves the frequency-domain optical tomography problem with angularly averaged measurement. The numerical procedure involves multiple techniques such as the finite volume method for transport discretization, limited memory BFGS for minimization, the adjoint state method for computing Fréchet derivatives, the matrix-free approach for evaluating matrix vector products, and the L-curve method for selecting regularization parameters. The combination of those techniques allows us to perform numerical reconstructions in geometrical settings that are of practical importance.

The remainder of the paper is organized as follows. In section 2 we formulate the inverse problem in frequency-domain optical tomography as a classical regularized

least-squares problem and describe the adjoint method for the computation of the gradient of the least-squares functional. In section 3 we present the details on the discretization of the forward and backward problems, which is based on using a discrete ordinates methods for the directional variables and a finite volume method for the spatial variables. The implementation of the minimization procedure, based on a quasi-Newton optimization algorithm, and the choice of the regularization parameter are described in section 4. Several numerical simulations illustrate the performance of the code in section 5. Reconstructions based on synthetic data indeed show a significant reduction of the cross-talk mentioned above when frequency-domain data are used. Concluding remarks are finally offered in section 6.

2. Problem formulation. We now formulate the optical tomography problem. Let $\mathcal{D} \subset \mathbb{R}^n$ be our domain of interest, with sufficiently regular boundary $\partial\mathcal{D}$. Then the frequency-domain equation of radiative transfer that describes the photon density in the phase space, i.e., as a function of position $\mathbf{x} \in \mathcal{D}$ and direction $\boldsymbol{\theta} \in S^{n-1}$ (unit sphere of $\mathbb{R}^n$) is given by [4]

$$\begin{aligned} \mathcal{T}u \equiv \Big(\frac{i\omega}{v} + \boldsymbol{\theta}\cdot\nabla + \sigma_a(\mathbf{x})\Big)u(\mathbf{x},\boldsymbol{\theta}) + Q(u)(\mathbf{x},\boldsymbol{\theta}) = 0 \qquad & \text{in } \mathcal{D}\times S^{n-1}, \\ u(\mathbf{x},\boldsymbol{\theta}) = f(\mathbf{x},\boldsymbol{\theta}) \qquad & \text{on } \Gamma_-, \end{aligned} \tag{2.1}$$

where $i = \sqrt{-1}$, $n = 2, 3$ is the space dimension, $v \in \mathbb{R}^+$ is the speed of light in the medium, and ω is the modulation frequency of the boundary source $f(\mathbf{x},\boldsymbol{\theta})$. The nonnegative function $\sigma_a(\mathbf{x}) \in L^\infty(\mathcal{D})$ is the absorption coefficient. The unknown quantity, $u(\mathbf{x},\boldsymbol{\theta})$, is the radiant power per unit solid angle per unit area perpendicular to the direction of propagation at $\mathbf{x}$ in the direction $\boldsymbol{\theta}$. Note that $u(\mathbf{x},\boldsymbol{\theta})$ depends also on ω although, for simplicity, we do not write this dependency explicitly. The boundary sets $\Gamma_\pm$ are defined as

$$\Gamma_\pm = \{(\mathbf{x},\boldsymbol{\theta}) \in \partial\mathcal{D}\times S^{n-1} \text{ s.t. } \pm\boldsymbol{\theta}\cdot\boldsymbol{\nu}(\mathbf{x}) > 0\},$$

with $\boldsymbol{\nu}(\mathbf{x})$ the outward unit normal to the domain at $\mathbf{x} \in \partial\mathcal{D}$. The scattering operator Q is defined as

$$Q(u)(\mathbf{x},\boldsymbol{\theta}) = \sigma_s(\mathbf{x})\Big(u(\mathbf{x},\boldsymbol{\theta}) - \int_{S^{n-1}} k(\boldsymbol{\theta}\cdot\boldsymbol{\theta}')u(\mathbf{x},\boldsymbol{\theta}')d\mu(\boldsymbol{\theta}')\Big). \tag{2.2}$$

Here, $\sigma_s(\mathbf{x}) \in L^\infty(\mathcal{D})$ is the scattering coefficient and $d\mu$ is the surface measure on S^{n-1} normalized so that $\int_{S^{n-1}} d\mu(\boldsymbol{\theta}) = 1$. The "collision" kernel $k(\boldsymbol{\theta}\cdot\boldsymbol{\theta}')$, which describes the probability that photons traveling in direction $\boldsymbol{\theta}'$ scatter into direction $\boldsymbol{\theta}$, is a positive function independent of $\mathbf{x}$ and satisfies the normalization condition:

$$\int_{S^{n-1}} k(\boldsymbol{\theta}\cdot\boldsymbol{\theta}')d\mu(\boldsymbol{\theta}') = 1. \tag{2.3}$$

The scattering kernel for light propagation in tissues is highly peaked forward and is chosen as the Henyey–Greenstein phase function [29, 62]

$$k(\boldsymbol{\theta}\cdot\boldsymbol{\theta}') = C\frac{1-g^2}{(1+g^2-2g\cos\phi)^{3/2}}, \tag{2.4}$$

where ϕ is the angle between $\boldsymbol{\theta}$ and $\boldsymbol{\theta}'$, i.e., $\boldsymbol{\theta}\cdot\boldsymbol{\theta}' = \cos\phi$, and where $g \in [0,1]$ is the anisotropy factor, which measures how peaked forward the phase function is.

The larger g is, the more forward the scattering. The anisotropy factor is often used to define the so-called effective scattering coefficient through $\sigma_s' = (1-g)\sigma_s$. C is a normalization constant such that (2.3) hold. We mention that scattering kernels other than (2.4) have also been used in some situations [36] and that simplified (Fokker–Planck) models can also be used to analyze highly peaked scattering in biological tissues [37].

The optical tomography problem thus consists of reconstructing $\sigma_a(\mathbf{x})$ and $\sigma_s(\mathbf{x})$ in (2.1) from boundary current measurements; see (2.6) below. Our objective in this work is to present a numerical scheme that performs the reconstruction.

2.1. Forward problem. The absorption and scattering coefficients σ_a and σ_s cannot take negative values and have to be bounded. We thus introduce the following parameter space $\mathcal{Q}$:

$$\mathcal{Q} := \{(\sigma_a, \sigma_s) \ : \ \sigma_a \geq 0, \ \sigma_s \geq 0, \ \text{ and } (\sigma_a, \sigma_s) \in L^\infty(\mathcal{D}) \times L^\infty(\mathcal{D})\}.$$

We also introduce the functional spaces [2, 16]:

$$L^2_{\boldsymbol{\theta}\cdot\nu}(\Gamma_\pm) := \Big\{u(\mathbf{x},\boldsymbol{\theta}) : \int_{\Gamma_\pm} |u(\mathbf{x},\boldsymbol{\theta})|^2 |\boldsymbol{\theta}\cdot\nu(\mathbf{x})| d\sigma(\mathbf{x}) d\mu(\boldsymbol{\theta}) < +\infty\Big\},$$
$$\mathcal{W}^2(\mathcal{D}\times S^{n-1}) := \big\{u(\mathbf{x},\boldsymbol{\theta}) : u \in L^2(\mathcal{D}\times S^{n-1}) \text{ and } \boldsymbol{\theta}\cdot\nabla u \in L^2(\mathcal{D}\times S^{n-1})\big\}.$$

Adapting well-known results [2, 16] with complex-valued absorption coefficient $\sigma_a + \frac{i\omega}{v}$ in $L^\infty(\mathcal{D})$, we have the following statement about the forward problem.

PROPOSITION 2.1. [illegible] $(\sigma_a, \sigma_s) \in \mathcal{Q}$ [illegible] $\omega < +\infty$ [illegible] $f \in L^2_{\boldsymbol{\theta}\cdot\nu}(\Gamma_-)$ [illegible] (2.1) [illegible] $u(\mathbf{x},\boldsymbol{\theta}) \in \mathcal{W}^2(\mathcal{D}\times S^{n-1})$

We can then define the following [illegible] operator (as well as its adjoint) [14, 51]:

$$\Lambda : \begin{array}{l} f \longmapsto u|_{\Gamma_+} \\ L^2_{\boldsymbol{\theta}\cdot\nu}(\Gamma_-) \longmapsto L^2_{\boldsymbol{\theta}\cdot\nu}(\Gamma_+). \end{array} \tag{2.5}$$

The albedo operator Λ maps the incoming flux on the boundary into the outgoing flux and is a functional of the optical parameters σ_a and σ_s.

A major difficulty in optical tomography comes from the fact that in practice only outgoing currents, which are angular averages of the outgoing flux and are similar to diffusion-type measurements, are available. This prevents us from using classical uniqueness and stability results in inverse transport theory [14]. In fact, the inverse problem we solve in this paper is very similar to the diffusion-based inverse problem [4], on which many more theoretical results exist. To date, we do not know of any theoretical result on the reconstruction of optical properties from outgoing currents for arbitrary geometries. This makes the development of numerical tools all the more important.

To be consistent with existing measurement technologies, we define the following "measurement operator":

$$\begin{aligned} \mathcal{G}u|_{\Gamma_+} &:= \int_{S^{n-1}_+} \boldsymbol{\theta}\cdot\boldsymbol{\nu}(\mathbf{x}) u|_{\Gamma_+} d\mu(\boldsymbol{\theta}) \equiv z(\mathbf{x}), \\ \mathcal{G} &: L^2_{\boldsymbol{\theta}\cdot\nu}(\Gamma_+) \longmapsto L^2(\partial\mathcal{D}) \equiv \mathcal{Z}, \end{aligned} \tag{2.6}$$

with $S_+^{n-1} := \{\boldsymbol{\theta} : \boldsymbol{\theta} \in S^{n-1} \text{ s.t. } \boldsymbol{\theta} \cdot \boldsymbol{\nu}(\mathbf{x}) > 0\}$. We will call $\mathcal{Z}$ the "measurement space." Now the composite operator $\mathcal{G}\Lambda : f \mapsto z$ maps the incoming flux into the tomographic measurements. The adjoint operator $\mathcal{G}^*$ of $\mathcal{G}$ is defined via the identity

$$\left\langle \overline{\mathcal{G}^* g_1}, g_2 \right\rangle_{L^2_{\boldsymbol{\theta}\cdot\boldsymbol{\nu}}(\Gamma_+)} = \left\langle \overline{g_1}, \mathcal{G} g_2 \right\rangle_{\mathcal{Z}} \tag{2.7}$$

for all $g_1 \in \mathcal{Z}$ and $g_2 \in L^2_{\boldsymbol{\theta}\cdot\boldsymbol{\nu}}(\Gamma_+)$, where the symbol $\overline{\mathscr{Y}_1}$ denotes the complex conjugate of $\mathscr{Y}_1$, and $\langle \cdot, \cdot \rangle_X$ is the usual inner product in a Hilbert space X. One observes that $\mathcal{G}^*$ is nothing but the operation of multiplication by $\boldsymbol{\theta} \cdot \boldsymbol{\nu}(\mathbf{x})$.

2.2. Least-squares formulation. The inverse problem of optical tomography can be formulated as follows: determine $(\sigma_a, \sigma_s) \in \mathcal{Q}$ such that

$$\mathcal{G}\Lambda f = z \tag{2.8}$$

holds for all possible source-measurement pairs (f, z). Here $z \in \mathcal{Z} \equiv L^2(\partial\mathcal{D})$ is the measured data corresponding to source f. This problem is in general severely ill-posed (assuming that uniqueness of reconstruction holds as in diffusion theory [4]) in the sense that when no regularization is applied, noise contained in the data z is more amplified during the inversion procedure than what would result from an arbitrary number of differentiations [21]. Another practical difficulty in solving (2.8) lies in the fact that the amount of available data may be quite limited [47]. For example, one may only be able to use a limited number (say, N_q) of light sources. After discretizing (2.8) on a reasonable mesh, we will end up with a very underdetermined nonlinear system. A classical way to resolve the lack of measurements is to turn to the following least-squares formulation: find (σ_a, σ_s) solving

$$\mathcal{F}(\sigma_a, \sigma_s) =: \frac{1}{2} \sum_{q=1}^{N_q} \left\| \mathcal{G}\Lambda f_q - z_q \right\|_{\mathcal{Z}}^2 \to \min . \tag{2.9}$$

Here, $1 \le q \le N_q$ denotes the light source number. For reasons we have mentioned earlier, the least-squares problem (2.9) is usually not stable [4]. To stabilize the problem, we impose additional smoothness restrictions on the coefficients we wish to reconstruct. In other words, we look for optical properties in a space that is much smaller than $\mathcal{Q}$. We call this space the space of admissible solutions:

$$\mathcal{Q}_{ad} := \{(\sigma_a, \sigma_s) \ : (\sigma_a, \sigma_s) \in [\sigma_a^l, \sigma_a^u] \times [\sigma_s^l, \sigma_s^u], \text{ and } (\sigma_a, \sigma_s) \in \mathcal{H}^1(\mathcal{D}) \times \mathcal{H}^1(\mathcal{D})\},$$

where σ_a^l (resp., σ_a^u) and σ_s^l (resp., σ_s^u) are lower (resp., upper) bounds of σ_a and σ_s, respectively, with $\sigma_a^l > 0$ and $\sigma_s^l > 0$. $\mathcal{H}^1(\mathcal{D})$ is the usual Hilbert space of $L^2(\mathcal{D})$ functions with first-order partial derivatives in $L^2(\mathcal{D})$:

$$\|\mathscr{Y}\|^2_{\mathcal{H}^1(\mathcal{D})} := \|\mathscr{Y}\|^2_{L^2(\mathcal{D})} + \|\nabla\mathscr{Y}\|^2_{L^2(\mathcal{D})} \quad \text{for } \mathscr{Y} \in \mathcal{H}^1(\mathcal{D}). \tag{2.10}$$

It is known that $\mathcal{Q}_{ad}$ is a closed and convex subset of $\mathcal{H}^1(\mathcal{D}) \times \mathcal{H}^1(\mathcal{D})$. We can thus introduce the following regularized least-squares functional:

$$\mathcal{F}_\beta(\sigma_a, \sigma_s) := \mathcal{F}(\sigma_a, \sigma_s) + \frac{\beta}{2} \mathcal{J}(\sigma_a, \sigma_s), \tag{2.11}$$

where the last term is a regularization term and β is the regularization parameter [21]. The method for choosing β will be described in section 4.3. We use the Tikhonov regularization functional in our problem:

$$\mathcal{J}(\sigma_a, \sigma_s) = \|\sigma_a - \sigma_a^0\|^2_{\mathcal{H}^1(\mathcal{D})} + \epsilon \|\sigma_s - \sigma_s^0\|^2_{\mathcal{H}^1(\mathcal{D})}, \tag{2.12}$$

where σ_a^0 and σ_s^0 are initial guesses for the σ_a and σ_s profiles, and ϵ is a small constant. The choice of ϵ is addressed in section 4.3. We thus formulate the optical tomography problem as the following regularized least-squares problem:

$$\text{(RLS)} \qquad \begin{aligned} &\min_{(\sigma_a,\sigma_s)} \mathcal{F}_\beta \\ &\sigma_a^l \le \sigma_a \le \sigma_a^u, \\ &\sigma_s^l \le \sigma_s \le \sigma_s^u. \end{aligned}$$

We first observe that problem (RLS) has at least one solution in the sense that the functional $\mathcal{F}_\beta(\sigma_a, \sigma_s)$ admits at least one minimizer. This existence result is classical and follows from the weak lower semicontinuity and coercivity of $\mathcal{F}_\beta(\sigma_a, \sigma_s)$ [46, 60]. However, we cannot show that $\mathcal{F}_\beta(\sigma_a, \sigma_s)$ is strictly convex and cannot conclude that the minimizer is unique [60].

Our implementation of the inverse problem of optical tomography is a gradient-based minimization approach. We thus need to compute the Fréchet derivative of the least-squares functional $\mathcal{F}_\beta(\sigma_a, \sigma_s)$. Direct estimates of the Fréchet derivatives being quite costly because the optical parameters are (at least at the continuous level) infinite-dimensional objects, we adopt the adjoint state (or costate) approach [60] to estimate the derivatives. We have the following result.

THEOREM 2.2 (Fréchet derivatives). [illegible] $\mathcal{F}_\beta(\sigma_a, \sigma_s)$ [illegible] σ_a [illegible] σ_s [illegible] (σ_a, σ_s) [illegible] $(h_a \ h_s)$ [illegible]

(2.13)
$$\begin{pmatrix} \mathcal{F}'_\beta h_a \\ \mathcal{F}'_\beta h_s \end{pmatrix} = \begin{pmatrix} \operatorname{Re} \sum_{q=1}^{N_q} \left\langle \varphi_q, \left(\dfrac{\partial \mathcal{T}}{\partial \sigma_a} h_a \right) u_q \right\rangle_{L^2(\mathcal{D}\times S^{n-1})} + \beta \left\langle \sigma_a - \sigma_a^0, h_a \right\rangle_{\mathcal{H}^1(\mathcal{D})} \\ \operatorname{Re} \sum_{q=1}^{N_q} \left\langle \varphi_q, \left(\dfrac{\partial \mathcal{T}}{\partial \sigma_s} h_s \right) u_q \right\rangle_{L^2(\mathcal{D}\times S^{n-1})} + \beta\epsilon \left\langle \sigma_s - \sigma_s^0, h_s \right\rangle_{\mathcal{H}^1(\mathcal{D})} \end{pmatrix},$$

[illegible] $\mathcal{T}$ [illegible] (2.1). u_q [illegible] φ_q [illegible] (2.1) [illegible] f_q [illegible] (2.16) [illegible] Re [illegible]. Let us denote by r_q the residual $\mathcal{G}\Lambda f_q - z_q = \mathcal{G}u_q|_{\Gamma_+} - z_q$. According to [18, 19], r_q is Fréchet differentiable with respect to both σ_a and σ_s. The L^2-norm is Fréchet differentiable as shown in [46]. By the chain rule, $\|r_q\|_{\mathcal{Z}}^2$ is Fréchet differentiable. Since the summation is finite, we deduce that $\mathcal{F}$ is differentiable. Together with the fact that $\mathcal{J}$ is differentiable, we conclude that $\mathcal{F}_\beta$ is Fréchet differentiable with respect to σ_a and σ_s.

We now compute these Fréchet derivatives. Let us compute the derivative with respect to σ_a:

$$\begin{aligned} \mathcal{F}'_\beta(\sigma_a, \sigma_s) h_a &= \operatorname{Re} \sum_{q=1}^{N_q} \left\langle \overline{r}_q, \mathcal{G} \left(\frac{\partial u_q|_{\Gamma_+}}{\partial \sigma_a} h_a \right) \right\rangle_{\mathcal{Z}} + \beta \left\langle \sigma_a - \sigma_a^0, h_a \right\rangle_{\mathcal{H}^1(\mathcal{D})} \\ &= \operatorname{Re} \sum_{q=1}^{N_q} \left\langle \overline{\mathcal{G}^* r_q}, \frac{\partial u_q|_{\Gamma_+}}{\partial \sigma_a} h_a \right\rangle_{L^2_{\boldsymbol{\theta}\cdot\nu}(\Gamma_+)} + \beta \left\langle \sigma_a - \sigma_a^0, h_a \right\rangle_{\mathcal{H}^1(\mathcal{D})}, \end{aligned} \tag{2.14}$$

where we have used the properties of the adjoint operator (2.7). On the other hand, differentiating the transport equation (2.1) for source f_q gives

$$\begin{aligned} \mathcal{T}\phi_q + \left(\frac{\partial \mathcal{T}}{\partial \sigma_a} h_a\right) u_q = 0 \quad &\text{in } \mathcal{D}\times S^{n-1}, \\ \phi_q = 0 \quad &\text{on } \Gamma_-, \end{aligned} \tag{2.15}$$

where $\phi_q \equiv \frac{\partial u_q}{\partial \sigma_a} h_a$, and $\mathcal{T}$ is the transport operator defined in (2.1). We need also to introduce an adjoint variable φ_q of u_q which is the solution of the following adjoint transport equation:

$$\begin{aligned} \mathcal{T}^*\varphi_q \equiv \Big(\frac{i\omega}{v} - \boldsymbol{\theta}\cdot\nabla + \sigma_a(\mathbf{x})\Big)\varphi_q(\mathbf{x},\boldsymbol{\theta}) + Q(\varphi_q)(\mathbf{x},\boldsymbol{\theta}) = 0 \quad &\text{in } \mathcal{D}\times S^{n-1}, \\ \varphi_q(\mathbf{x},\boldsymbol{\theta}) = -\overline{\mathcal{G}^* r_q} \quad &\text{on } \Gamma_+. \end{aligned} \tag{2.16}$$

Here we have used that $Q^* = Q$, which follows from the definition (2.2). Multiplying (2.15) by φ_q and (2.16) by ϕ_q, then integrating over $\mathcal{D}\times S^{n-1}$, we obtain

$$\langle \overline{\mathcal{G}^* r_q}, \phi_q \rangle_{L^2_{\boldsymbol{\theta}\cdot\nu}(\Gamma_+)} = \left\langle \varphi_q, \left(\frac{\partial \mathcal{T}}{\partial \sigma_a} h_a\right) u_q \right\rangle_{L^2(\mathcal{D}\times S^{n-1})}, \tag{2.17}$$

which leads to

$$\mathcal{F}'_\beta(\sigma_a,\sigma_s)h_a = \mathrm{Re}\sum_{q=1}^{N_q} \left\langle \varphi_q, \left(\frac{\partial \mathcal{T}}{\partial \sigma_a} h_a\right) u_q \right\rangle_{L^2(\mathcal{D}\times S^{n-1})} + \beta \left\langle \sigma_a - \sigma_a^0, h_a \right\rangle_{\mathcal{H}^1(\mathcal{D})}. \tag{2.18}$$

The derivative with respect to σ_s can be computed similarly. ∎

This result shows that in order to compute the Fréchet derivative of the objective functional $\mathcal{F}_\beta(\sigma_a,\sigma_s)$, we need to solve one forward transport problem (2.1) and one adjoint transport problem (2.16).

3. Discretization methods. There is a vast literature on the discretization of radiative transfer equations; see, for instance, [1, 25, 44]. In this paper, we have chosen to use the discrete ordinates method to discretize the directional variables and the finite volume method [22] to discretize the spatial variables.

3.1. The discrete ordinates formulation. In the discrete ordinates method [1, 44], we approximate the total scalar flux, defined as the integral of $u(\mathbf{x},\boldsymbol{\theta})$ over S^{n-1}, by the following quadrature rule:

$$\int_{S^{n-1}} u(\mathbf{x},\boldsymbol{\theta})\, d\mu(\boldsymbol{\theta}) \approx \sum_{j=1}^{J} \eta_j u(\mathbf{x},\boldsymbol{\theta}_j), \tag{3.1}$$

where $\boldsymbol{\theta}_j$ is the jth direction and η_j the associated weight for $1 \le j \le J$. Details on how to choose the set of directions $\{\boldsymbol{\theta}_j\}_{j=1}^J$ and the corresponding weights $\{\eta_j\}_{j=1}^J$ can be found in [44]. To ensure particle conservation, we impose that

$$\sum_{j=1}^{J} \eta_j = 1. \tag{3.2}$$

The equation of radiative transfer is now decomposed as a discrete set of J coupled differential equations that describe the photon flux field along J directions:

$$\nabla \cdot (\boldsymbol{\theta}_j u) + \left(\sigma_t + \frac{i\omega}{v}\right) u(\mathbf{x}, \boldsymbol{\theta}_j) = \sigma_s(\mathbf{x}) \sum_{j'=1}^{J} \eta_{j'} k_{jj'} u(\mathbf{x}, \boldsymbol{\theta}_{j'}) \tag{3.3}$$

for $j = 1, 2, \ldots, J$, where $k_{jj'} = k(\boldsymbol{\theta}_j \cdot \boldsymbol{\theta}_{j'})$, and where $\sigma_t = \sigma_a + \sigma_s$. We impose

$$\sum_{j=1}^{J} \eta_j k_{jj'} = 1, \qquad 1 \le j' \le J, \tag{3.4}$$

so that the number of photons in the system is preserved by the scattering process.

3.2. Spatial discretization. We use a finite volume method to perform the spatial discretization. Finite volume methods [22] ensure the conservation of mass (or momentum, energy) in a discrete sense, which is important in transport calculations. They also have the advantage of easily handling complicated geometries by arbitrary triangulations, which we need in tomographic applications.

We implement a cell-centered version of the finite volume methods. Consider a mesh of $\mathbb{R}^n$, $\mathcal{M}$, consisting of polyhedral bounded convex subsets of $\mathbb{R}^n$, which covers our computational domain $\mathcal{D}$. Let $\mathcal{C} \in \mathcal{M}$ be a [illegible], that is, an element of the mesh $\mathcal{M}$, $\partial\mathcal{C}$ its boundary, and $V_{\mathcal{C}}$ its Lebesgue measure. We assume that the unknown quantity, for example, $u(\mathbf{x}, \theta_j)$, takes its averaged value in $\mathcal{C}$ (and thus is constant). We denote this value by $u_j^{\mathcal{C}}$:

$$u_j^{\mathcal{C}} \equiv \frac{1}{V_{\mathcal{C}}} \int_{V_{\mathcal{C}}} u(\mathbf{x}, \boldsymbol{\theta}_j) d\mathbf{x}. \tag{3.5}$$

Integrating the above discrete ordinates equations (3.3) over cell $\mathcal{C}$ and using the divergence theorem on the first term, we obtain the equations

$$\int_{\partial\mathcal{C}} \boldsymbol{\theta}_j \cdot \mathbf{n}_{\mathcal{C}}(\mathbf{x}) u_j d\gamma(\mathbf{x}) + \left(\sigma_t^{\mathcal{C}} + \frac{i\omega}{v}\right) V_{\mathcal{C}} u_j^{\mathcal{C}} = V_{\mathcal{C}} \sigma_s^{\mathcal{C}} \sum_{j'=1}^{J} \eta_{j'} k_{jj'} u_{j'}^{\mathcal{C}} \tag{3.6}$$

for $1 \le j \le J$, where $\mathbf{n}_{\mathcal{C}}(\mathbf{x})$ denotes the outward normal to $\partial\mathcal{C}$ at point $\mathbf{x} \in \partial\mathcal{C}$, $d\gamma(\mathbf{x})$ denotes the surface Lebesgue measure on $\partial\mathcal{C}$, and $\sigma_s^{\mathcal{C}}$ ($\sigma_t^{\mathcal{C}}$) is the value of σ_s (σ_t) on cell $\mathcal{C}$.

Now we have to approximate the flux through the boundary of $\mathcal{C}$, i.e., the first integral term in (3.6). Let $\{\mathcal{C}_i\}_{i=1}^{I}$ be the set of neighboring cells of $\mathcal{C}$. We denote by $S_{\mathcal{C},i}$ the common edge of cell $\mathcal{C}$ and $\mathcal{C}_i$, i.e., $S_{\mathcal{C},i} = \partial\mathcal{C} \cap \partial\mathcal{C}_i$. We then have

$$\int_{\partial\mathcal{C}} \boldsymbol{\theta}_j \cdot \mathbf{n}_{\mathcal{C}}(\mathbf{x}) u_j d\gamma(\mathbf{x}) = \sum_i \int_{S_{\mathcal{C},i}} \boldsymbol{\theta}_j \cdot \mathbf{n}_{\mathcal{C}}(\mathbf{x}) u_j d\gamma(\mathbf{x}). \tag{3.7}$$

The flux $\int_{S_{\mathcal{C},i}} \boldsymbol{\theta}_j \cdot \mathbf{n}_{\mathcal{C}}(\mathbf{x}) u_j d\gamma(\mathbf{x})$ can be approximated by various numerical schemes. In this work, we take a first-order upwind scheme:

$$F_{j,i}^{\mathcal{C}} := \int_{S_{\mathcal{C},i}} \boldsymbol{\theta}_j \cdot \mathbf{n}_{\mathcal{C}}(\mathbf{x}) u_j d\gamma(\mathbf{x}) = \begin{cases} \boldsymbol{\theta}_j \cdot \mathbf{n}_{\mathcal{C}} |S_{\mathcal{C},i}| u_j^{\mathcal{C}} & \text{if } \boldsymbol{\theta}_j \cdot \mathbf{n}_{\mathcal{C}} \ge 0, \\ \boldsymbol{\theta}_j \cdot \mathbf{n}_{\mathcal{C}} |S_{\mathcal{C},i}| u_j^{\mathcal{C}_i} & \text{if } \boldsymbol{\theta}_j \cdot \mathbf{n}_{\mathcal{C}} < 0, \end{cases} \tag{3.8}$$

where $|S_{\mathcal{C},i}|$ is the Lebesgue measure of $S_{\mathcal{C},i}$. We then obtain a full discretization of the discrete ordinates equations

$$\sum_i F^{\mathcal{C}}_{j,i} + \left(\sigma^{\mathcal{C}}_t + \frac{i\omega}{v}\right) V_{\mathcal{C}} u^{\mathcal{C}}_j = V_{\mathcal{C}} \sigma^{\mathcal{C}}_s \sum_{j'=1}^{J} \eta_{j'} k_{jj'} u^{\mathcal{C}}_{j'} \tag{3.9}$$

for $j = 1, 2, \ldots, J$. Let N denote the total number of control cells. After collecting the discretized transport equation (3.9) on all control cells, we arrive at the following system of complex-valued algebraic equations:

$$\mathbf{AU} = \mathbf{SU} + \mathbf{G}, \tag{3.10}$$

where $\mathbf{A} \in \mathbb{C}^{NJ\times NJ}$ and $\mathbf{S} \in \mathbb{C}^{NJ\times NJ}$ are the discretized streaming-collision and scattering operators, respectively. The boundary source $f(\mathbf{x}, \boldsymbol{\theta})$, which comes into the discretized system via the flux approximation (3.8), is denoted by $\mathbf{G}$. The vector $\mathbf{U} \in \mathbb{C}^{NJ\times 1}$, which contains the values of $u(\mathbf{x}, \boldsymbol{\theta})$ on the cell $\mathcal{C}$ in the direction $\boldsymbol{\theta}_j$, is organized as

$$\mathbf{U} = \begin{pmatrix} \mathbf{U}_1 \\ \vdots \\ \mathbf{U}_J \end{pmatrix}, \text{ with } \mathbf{U}_j = \begin{pmatrix} u^1_j \\ \vdots \\ u^N_j \end{pmatrix} \in \mathbb{C}^N. \tag{3.11}$$

The matrices $\mathbf{A}$ and $\mathbf{S}$ have sparse structures. In fact, they are sparse block matrices. $\mathbf{A}$ is a block diagonal matrix that can be written as

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_1 & & \\ & \ddots & \\ & & \mathbf{A}_J \end{pmatrix} + \begin{pmatrix} \mathbf{C}_0 & & \\ & \ddots & \\ & & \mathbf{C}_0 \end{pmatrix}, \tag{3.12}$$

where $\mathbf{A}_j \in \mathbb{C}^{N\times N}$ is the discretization of the advection operator $\mathcal{A}$ defined by $\mathcal{A}u := \boldsymbol{\theta}_j \cdot \nabla u$. From (3.8) we can deduce that $\mathbf{A}_j$ has no more than $N \times N_E$ nonzero elements, where N_E is the total number of edges (surfaces in three dimensions) each control cell has.

Matrix $\mathbf{C}_0 \in \mathbb{C}^{N\times N}$ is diagonal:

$$\mathbf{C}_0 = \begin{pmatrix} V_1(\sigma^1_t + \frac{i\omega}{v}) & & \\ & \ddots & \\ & & V_N(\sigma^N_t + \frac{i\omega}{v}) \end{pmatrix},$$

where we recall $\sigma^i_t \equiv \sigma^i_a + \sigma^i_s$ $(i = 1, \ldots, N)$.

The matrix $\mathbf{S}$ can be expressed as the direct product of two smaller matrices:

$$\mathbf{S} = \mathbf{K} \otimes \mathbf{D}_0, \tag{3.13}$$

with $\mathbf{D}_0 \in \mathbb{C}^{N\times N}$ a diagonal matrix given by

$$\mathbf{D}_0 = \begin{pmatrix} V_1\sigma^1_s & & \\ & \ddots & \\ & & V_N\sigma^N_s \end{pmatrix},$$

and $\mathbf{K} \in \mathbb{C}^{J\times J}$ a dense matrix with component $(\mathbf{K})_{jj'} = \eta_{j'} k_{jj'}$. In practical applications, the number of directions is much smaller than the number of spatial mesh elements ($J \ll N$). So although $\mathbf{K}$ is dense, the scattering matrix $\mathbf{S}$ is sparse. However, in general the matrix $\mathbf{K}$ is not symmetric unless we choose η_j to be constant. The matrix $\mathbf{A} - \mathbf{S}$ is thus neither symmetric nor positive definite, which is the reason for us to choose a GMRES solver in section 4.2.

Let us remark here that our finite volume discretization reduces to an upwind finite difference scheme on usual finite difference grids. We refer to our earlier work [54] for some numerical tests on the finite volume discretization of the transport equation.

3.3. Discrete adjoint problem. We present in this section the numerical method we have employed to compute the gradient of discrete objective function with respect to the optical properties on each cell.

To simplify the notation, from now on we denote by $\sigma_a \in \mathbb{R}^{N\times 1}$ the absorption coefficient vector $(\sigma_a^1, \ldots, \sigma_a^{\mathcal{C}}, \ldots, \sigma_a^N)^T$ and by $\sigma_s \in \mathbb{R}^{N\times 1}$ the scattering coefficient vector $(\sigma_s^1, \ldots, \sigma_s^{\mathcal{C}}, \ldots, \sigma_s^N)^T$.

We want to minimize the discrepancy between model predictions and measurements over a set of source and detector pairs. Let N_q denote the number of sources used in an experiment, and let N_d denote the number of detectors used for each source. Then the following objective function we employed takes the form

$$\mathcal{F}_\beta(\sigma_a, \sigma_s) = \frac{1}{2}\sum_{q=1}^{N_q}\sum_{d=1}^{N_d} |\mathcal{P}_d \mathbf{U}^q - z_{q,d}^\delta|^2 + \frac{\beta}{2}\mathcal{J}(\sigma_a, \sigma_s), \tag{3.14}$$

where $z_{q,d}^\delta$ denote the dth measurement of the qth source. The superscript δ is used to denote the level of noise contained in the measurements. $\mathbf{U}^q$ is a solution of the transport equation for the qth source. $\mathcal{P}_d \in \mathbb{R}^{1\times N}$ is a discretized version of the measurement operator. It takes the outgoing flux at detector d and averages over S_+^{n-1}. The discretized regularization term is given by

$$\begin{aligned}\mathcal{J}(\sigma_a, \sigma_s) = \sum_{\mathcal{C}=1}^{N}\Big(\sum_{\kappa=\{x,y,z\}} [\mathcal{D}_\kappa^{\mathcal{C}}(\sigma_a - \sigma_a^0)]^2 + (\sigma_a^{\mathcal{C}} - \sigma_a^{0,\mathcal{C}})^2\Big) \\ + \epsilon\sum_{\mathcal{C}=1}^{N}\Big(\sum_{\kappa=\{x,y,z\}} [\mathcal{D}_\kappa^{\mathcal{C}}(\sigma_s - \sigma_s^0)]^2 + (\sigma_s^{\mathcal{C}} - \sigma_s^{0,\mathcal{C}})^2\Big),\end{aligned} \tag{3.15}$$

where $\mathcal{D}_\kappa^{\mathcal{C}} \in \mathbb{R}^{1\times N}$ denotes the discretized partial differential operator at cell $\mathcal{C}$ in the κ $(= x, y, z)$ direction.

We now start to compute the gradient of objective function (3.14) with respect to optical properties on each mesh element. It is straightforward to check that

$$\frac{\partial \mathcal{F}_\beta}{\partial \sigma_a^{\mathcal{C}}} = \Big[\sum_{q=1}^{N_q}\sum_{d=1}^{N_d} \overline{\mathbf{r}}_d^q \mathcal{P}_d \frac{\partial \mathbf{U}^q}{\partial \sigma_a^{\mathcal{C}}}\Big]_{\mathrm{Re}} + \frac{\beta}{2}\frac{\partial \mathcal{J}}{\partial \sigma_a^{\mathcal{C}}}, \tag{3.16}$$

with $\mathbf{r}_d^q = \mathcal{P}_d \mathbf{U}^q - z_{q,d}^\delta$, and $[\cdot]_{\mathrm{Re}}$ denotes the real part of $[\cdot]$.

At the same time, we notice from (3.10) that

$$\frac{\partial \mathbf{A}}{\partial \sigma_a^{\mathcal{C}}}\mathbf{U}^q + \mathbf{A}\frac{\partial \mathbf{U}^q}{\partial \sigma_a^{\mathcal{C}}} = \frac{\partial \mathbf{S}}{\partial \sigma_a^{\mathcal{C}}}\mathbf{U}^q + \mathbf{S}\frac{\partial \mathbf{U}^q}{\partial \sigma_a^{\mathcal{C}}} \tag{3.17}$$

for source $q = 1, \ldots, N_q$, which is equivalent to saying that

$$\frac{\partial \mathbf{U}^q}{\partial \sigma_a^{\mathcal{C}}} = -(\mathbf{A} - \mathbf{S})^{-1} \frac{\partial (\mathbf{A} - \mathbf{S})}{\partial \sigma_a^{\mathcal{C}}} \mathbf{U}^q, \tag{3.18}$$

since $\mathbf{A} - \mathbf{S}$ is invertible. It is very important to note that the matrices $\mathbf{A}$ and $\mathbf{S}$ are independent of the source used. Thus, there are no superscripts q associated with them. We thus have

$$\frac{\partial \mathcal{F}_\beta}{\partial \sigma_a^{\mathcal{C}}} = -\Big[\sum_{q=1}^{N_q} \sum_{d=1}^{N_d} \bar{\mathbf{r}}_d^q \mathcal{P}_d (\mathbf{A} - \mathbf{S})^{-1} \frac{\partial (\mathbf{A} - \mathbf{S})}{\partial \sigma_a^{\mathcal{C}}} \mathbf{U}^q \Big]_{\mathrm{Re}} + \frac{\beta}{2} \frac{\partial \mathcal{J}}{\partial \sigma_a^{\mathcal{C}}}. \tag{3.19}$$

We now introduce a new state variable $\mathbf{\Psi}^q \in \mathbb{C}^{N \times 1}$ (called the adjoint variable of $\mathbf{U}^q$) given by

$$-\sum_{d=1}^{N_d} \bar{\mathbf{r}}_d^q \mathcal{P}_d (\mathbf{A} - \mathbf{S})^{-1} = \mathbf{\Psi}^{qT}, \tag{3.20}$$

where $\mathbf{\Psi}^{qT}$ denotes the transpose of $\mathbf{\Psi}^q$. We then say that $\mathbf{\Psi}^q$ is the solution of the following [illegible] equation of (3.10):

$$(\mathbf{A} - \mathbf{S})^T \mathbf{\Psi}^q = -\sum_{d=1}^{N_d} \bar{\mathbf{r}}_d^q \mathcal{P}_d^T. \tag{3.21}$$

One then arrives at

$$\frac{\partial \mathcal{F}_\beta}{\partial \sigma_a^{\mathcal{C}}} = \Big[\sum_{q=1}^{N_q} \mathbf{\Psi}^{qT} \frac{\partial (\mathbf{A} - \mathbf{S})}{\partial \sigma_a^{\mathcal{C}}} \mathbf{U}^q \Big]_{\mathrm{Re}} + \frac{\beta}{2} \frac{\partial \mathcal{J}}{\partial \sigma_a^{\mathcal{C}}}, \tag{3.22}$$

with

$$\frac{\partial \mathcal{J}}{\partial \sigma_a^{\mathcal{C}}} = 2\Big(\sum_{\kappa = \{x,y,z\}} \mathcal{D}_\kappa^{\mathcal{C}} (\sigma_a - \sigma_a^0)(\mathcal{D}_\kappa^{\mathcal{C}} \mathcal{I}_{\mathcal{C}}) + (\sigma_a^{\mathcal{C}} - \sigma_a^{0,\mathcal{C}}) \Big),$$

where the unit direction vector $\mathcal{I}_{\mathcal{C}} \in \mathbb{R}^{N \times 1}$ is a vector whose $\mathcal{C}$th element is 1 and all other components are zero.

Very similar computation leads to the fact that the derivatives of the objective functional with respect to $\sigma_s^{\mathcal{C}}$ are given by

$$\frac{\partial \mathcal{F}_\beta}{\partial \sigma_s^{\mathcal{C}}} = \Big[\sum_{q=1}^{N_q} \mathbf{\Psi}^{qT} \frac{\partial (\mathbf{A} - \mathbf{S})}{\partial \sigma_s^{\mathcal{C}}} \mathbf{U}^q \Big]_{\mathrm{Re}} + \frac{\beta}{2} \frac{\partial \mathcal{J}}{\partial \sigma_s^{\mathcal{C}}}, \tag{3.23}$$

with

$$\frac{\partial \mathcal{J}}{\partial \sigma_s^{\mathcal{C}}} = 2\epsilon\Big(\sum_{\kappa = \{x,y,z\}} \mathcal{D}_\kappa^{\mathcal{C}} (\sigma_s - \sigma_s^0)(\mathcal{D}_\kappa^{\mathcal{C}} \mathcal{I}_{\mathcal{C}}) + (\sigma_s^{\mathcal{C}} - \sigma_s^{0,\mathcal{C}}) \Big).$$

Formulas (3.22) and (3.23) are what we used to compute the derivatives of objective function with respect to optical properties on each element. Note that we did not

form explicitly the matrix $\frac{\partial(\mathbf{A}-\mathbf{S})}{\partial\sigma_a^{\mathcal{C}}}$ (resp., $\frac{\partial(\mathbf{A}-\mathbf{S})}{\partial\sigma_s^{\mathcal{C}}}$) in the evaluations of $\mathbf{\Psi}^{qT}\frac{\partial(\mathbf{A}-\mathbf{S})}{\partial\sigma_a^{\mathcal{C}}}\mathbf{U}^q$ (resp., $\mathbf{\Psi}^{qT}\frac{\partial(\mathbf{A}-\mathbf{S})}{\partial\sigma_s^{\mathcal{C}}}\mathbf{U}^q$) because this matrix has a very simple sparse structure according to (3.12) and (3.13). Instead, a matrix-free method was adopted. In fact, since

$$\left[\frac{\partial(\mathbf{A}-\mathbf{S})}{\partial\sigma_a^{\mathcal{C}}}\right]_{ij} = \begin{cases} V_{\mathcal{C}}, & i=j \text{ and } \mod(i,N)=\mathcal{C}, \\ 0 & \text{otherwise}, \end{cases} \tag{3.24}$$

where we recall that N is the total number of volume cells that cover our computational domain, we have

$$\mathbf{\Psi}^{qT}\frac{\partial(\mathbf{A}-\mathbf{S})}{\partial\sigma_a^{\mathcal{C}}}\mathbf{U}^q = \sum_{j=1}^{J}\mathbf{\Psi}^q_{(j-1)\times N+\mathcal{C}}V_{\mathcal{C}}\mathbf{U}^q_{(j-1)\times N+\mathcal{C}}. \tag{3.25}$$

Note that here $\mathbf{\Psi}^q_{(j-1)\times N+\mathcal{C}}$ (resp., $\mathbf{U}^q_{(j-1)\times N+\mathcal{C}}$) denotes the $[(j-1)\times N+\mathcal{C}]$th element of $\mathbf{\Psi}^q$ (resp., $\mathbf{U}^q$).

The same observation on $\frac{\partial(\mathbf{A}-\mathbf{S})}{\partial\sigma_s^{\mathcal{C}}}$ leads to

$$\begin{aligned}\mathbf{\Psi}^{qT}\frac{\partial(\mathbf{A}-\mathbf{S})}{\partial\sigma_s^{\mathcal{C}}}\mathbf{U}^q = &\sum_{j=1}^{J}\mathbf{\Psi}^q_{(j-1)\times N+\mathcal{C}}V_{\mathcal{C}}\mathbf{U}^q_{(j-1)\times N+\mathcal{C}} \\ &- \sum_{j'=1}^{J}\sum_{j=1}^{J}(\mathbf{K})_{j'j}\mathbf{\Psi}^q_{(j'-1)\times N+\mathcal{C}}V_{\mathcal{C}}\mathbf{U}^q_{(j-1)\times N+\mathcal{C}},\end{aligned} \tag{3.26}$$

where the (j',j)th component of matrix $\mathbf{K}$, $(\mathbf{K})_{jj'}=\eta_{j'}k_{jj'}$, is given in (3.13). We can thus evaluate (3.22) and (3.23) without forming any intermediate matrices.

4. Numerical implementation. We have implemented the quasi-Newton optimization algorithm to solve the regularized least-squares problem (RLS) introduced in section 2.2. We have found in practice that this method converged much faster (in terms of function evaluations) than the nonlinear conjugate gradient method with either the Fletcher–Reeves or the Polak–Ribière updating formula [52]. This is expected from theory [52] and is consistent with practical applications tested in [41]. We have also implemented a Gauss–Newton method [52] to solve the least-squares problem (without the bounds constraints) and found that the method converges extremely slowly in our case. This is probably due to the fact that our problem is highly nonlinear and the Gauss–Newton method usually does not work well in this kind of situation [27, 52]. Detail comparison between various methods of solving the least-squares reconstruction problem is an ongoing project.

In this work, we employ the BFGS update rule [52] of the inverse Hessian matrix for our quasi-Newton method. The usual BFGS method, however, requires the explicit construction of the Hessian matrix, which is unrealistic for large problems. The memory size required to store the Hessian matrix is roughly proportional to the square of the memory used for the unknown parameters. We have thus resorted to a limited-memory version of the BFGS method which avoids the explicit construction of the inverse Hessian matrix.

4.1. Numerical optimization. The BFGS algorithm can be viewed as a special case of the quasi-Newton method [52]. With $\boldsymbol{\sigma}$ denoting the vector of discretized

optical properties, the quasi-Newton methods can be characterized by the following iterative process:

$$\boldsymbol{\sigma}_{k+1} = \boldsymbol{\sigma}_k + \alpha_k \mathbf{p}_k, \qquad k \in \mathbb{N}^+, \tag{4.1}$$

where $\mathbf{p}_k$ is a descent direction vector and α_k is the step length. The BFGS algorithm chooses $\mathbf{p}_k$ to be the solution of an approximated solution of Newton-type optimality equation, i.e.,

$$\mathbf{p}_k = H_k \mathbf{g}_k, \tag{4.2}$$

where $\mathbf{g}_k$ is the gradient of the least-squares functional, $\mathbf{g}_k = -\nabla_{\boldsymbol{\sigma}} \mathcal{F}_\beta(\boldsymbol{\sigma}_k)$. H_k is the inverse Hessian matrix of $\mathcal{F}_\beta$ at step k. Instead of computing real inverse Hessian matrices, which is very time consuming, the BFGS algorithm chooses to approximate H_k by the following updating rule:

$$H_{k+1} = W_k^T H_k W_k + \rho_k \mathbf{s}_k \mathbf{s}_k^T, \tag{4.3}$$

with $W_k = \mathbf{I} - \rho_k \mathbf{y}_k \mathbf{s}_k^T$, $\mathbf{s}_k = \boldsymbol{\sigma}_{k+1} - \boldsymbol{\sigma}_k$, $\mathbf{y}_k = \mathbf{g}_{k+1} - \mathbf{g}_k$, and $\rho_k = \frac{1}{\mathbf{y}_k^T \mathbf{s}_k}$. $\mathbf{I}$ is the identity matrix. As we mentioned above, forming (4.3) takes a tremendous amount of computer memory for large problems. To overcome this shortcoming, the limited-memory version of BFGS stores only the vectors $\mathbf{y}_k$ and $\mathbf{s}_k$ obtained in the last m ($3 \le m \le 7$ usually) iterations [35] and discards the rest. Thus after the first m iterations, (4.3) can be expressed as

$$\begin{aligned} H_{k+1} &= (W_k^T \cdots W_{k-m}^T) H_{k+1}^0 (W_{k-m} \cdots W_k) \\ &+ \rho_{k-m} (W_k^T \cdots W_{k-m+1}^T) \mathbf{s}_{k-m} \mathbf{s}_{k-m}^T \times (W_{k-m+1} \cdots W_k) \\ &+ \rho_{k-m+1} (W_k^T \cdots W_{k-m+2}^T) \mathbf{s}_{k-m+1} \mathbf{s}_{k-m+1}^T \times (W_{k-m+2} \cdots W_k) \\ &\vdots \\ &+ \rho_k \mathbf{s}_k \mathbf{s}_k^T, \end{aligned} \tag{4.4}$$

with the sparse initial guess H_{k+1}^0 given by $H_{k+1}^0 = \frac{\mathbf{y}_{k+1}^T \mathbf{s}_{k+1}}{\mathbf{y}_{k+1}^T \mathbf{y}_{k+1}} \mathbf{I}$.

We refer interested readers to [35, 52] for more details on the limited-memory BFGS algorithms, and to [41] for applications of those algorithms to optical tomographic problems. Convergence of BFGS algorithms has been proved under certain conditions and has been tested on many applications [11, 52].

To impose bounds on optical parameters, we have to modify the relation (4.2) slightly. We adopt a gradient projection method [11, 35, 52] to do this. At the beginning of each iteration, we use the gradient projection method to find a set of active bounds. We then solve a subminimization problem,

$$\min_{\boldsymbol{\sigma}} Q_k(\boldsymbol{\sigma}) \equiv \mathcal{F}_\beta(\boldsymbol{\sigma}_k) + \mathbf{g}_k^T (\boldsymbol{\sigma} - \boldsymbol{\sigma}_k) + \frac{1}{2} (\boldsymbol{\sigma} - \boldsymbol{\sigma}_k)^T H_k^{-1} (\boldsymbol{\sigma} - \boldsymbol{\sigma}_k), \tag{4.5}$$

on the set of free variables to find an approximation solution $\overline{\sigma}_{k+1}$, treating the active bounds as equality constraints by Lagrange multipliers. After we find an approximation solution $\overline{\boldsymbol{\sigma}}_{k+1}$, a line search along $\mathbf{p}_k = \overline{\boldsymbol{\sigma}}_{k+1} - \boldsymbol{\sigma}_k$ is done to find the step length α_k in (4.1). We use a line search method that enforces the Wolfe conditions [52]. That is, we look for an α_k that solves

$$\min_{\alpha_k > 0} \mathcal{F}_\beta(\boldsymbol{\sigma}_k + \alpha_k \mathbf{p}_k) \tag{4.6}$$

and satisfies

$$\mathcal{F}_\beta(\boldsymbol{\sigma}_k + \alpha_k \mathbf{p}_k) \le \mathcal{F}_\beta(\boldsymbol{\sigma}_k) + c_1 \alpha_k \nabla \mathcal{F}_\beta^T(\boldsymbol{\sigma}_k)\mathbf{p}_k, \tag{4.7}$$

$$\nabla \mathcal{F}_\beta^T(\boldsymbol{\sigma}_k + \alpha_k \mathbf{p}_k)\mathbf{p}_k \ge c_2 \nabla \mathcal{F}_\beta^T(\boldsymbol{\sigma}_k)\mathbf{p}_k, \tag{4.8}$$

where $c_1 = 10^{-4}$, $c_2 = 0.1$ in our case. More details on how to impose bound constraints in BFGS algorithms can be found in [35, sect. 5.5] and [11, 64].

4.2. Solving algebraic systems. As we have mentioned before, at each step of the minimization process, we have to solve both a discretized transport equation (3.10) and its adjoint problem (3.21) to compute the Fréchet derivatives (3.22) and (3.23) of the objective functional, forming the gradient vector $\mathbf{g}_k$ in (4.2). In fact, almost all of the computational time in the reconstruction process is devoted to the solution of these transport equations. In this work, instead of using the popular source iteration method, which converges very slowly in diffusive regimes unless it is properly accelerated [1], we choose to solve the forward problems by a preconditioned GMRES(n) algorithm [55, 56], where n denotes the number of iterative steps after which GMRES is restarted. Our general principle is to choose n large when the problem size is small and n small when the problem size is large. The implementation of the algorithm is based on the template provided in [9]. The preconditioner we employ is the zero fill-in incomplete LU factorization (ILU(0)) [53, 55] that has been proved to be efficient in transport calculations [53]. Details about this factorization can be found in [55]. In all of the numerical examples in section 5, we pick $n = 7$, and the GMRES algorithm is stopped if the relative residual is small enough. For example, the stopping criteria $\|\mathbf{G} - (\mathbf{A} - \mathbf{S})\mathbf{U}_k\|_{l^2}/\|\mathbf{G} - (\mathbf{A} - \mathbf{S})\mathbf{U}_0\|_{l^2} \le 10^{-10}$ is used to solve (3.10). Here $\mathbf{U}_0$ is the initial guess and $\mathbf{U}_k$ is the $\mathbf{U}$ value at the kth GMRES iteration.

4.3. Selecting regularization parameter. To choose the optimal regularization parameter β in (2.11), we adopt the L-curve method in this study. Although there exist proofs that the L-curve method fails to convergence for certain classes of inverse problems [59], we have observed satisfactory results in our applications. We plot the log of the regularization functional against the squared norm of the regularized residual, say, r_β, for a range of values of the regularization parameter. The right parameter β is the one at which the L-curve reaches the maximum of its curvature [28, 60]. One can show that the right β maximizes the curvature function [60]

$$\kappa(\beta) = -\frac{R(\beta)S(\beta)[\beta R(\beta) + \beta^2 S(\beta)] + [R(\beta)S(\beta)]/S'(\beta)}{[R^2(\beta) + \beta^2 S^2(\beta)]^{3/2}}, \tag{4.9}$$

where $R(\beta)$ and $S(\beta)$ are defined by

$$R(\beta) := \|r_\beta\|_{L^2}^2, \qquad S(\beta) := \mathcal{J}(\sigma_a, \sigma_s).$$

We recall that β is not included in $\mathcal{J}(\sigma_a, \sigma_s)$. One notices immediately that the L-curve method requires several reconstructions for any single problem, and thus is very time consuming. A simple [illegible] method is suggested in [27] to reduce the computational cost of regularization parameter selection. In this method, one starts the first reconstruction with a relatively large β. The result of this reconstruction is then taken to be the initial guess of next reconstruction with a smaller β. If the two

β are not dramatically different from each other, then the two reconstructions should converge to similar results. Thus, the reconstruction with smaller β is supposed to converge quickly since its initial guess is chosen to be close enough to its real solution. The process can be repeated to perform reconstructions with several values of β. In practice, this continuation method saves tremendous computational time for finding optimal regularization parameters. It now takes only about 80% extra work to find the optimal parameters in most cases. We adopt this continuation method in our three-dimensional numerical example (Example 4) in the next section. We present in Figures 5.2 (B) and 5.6 (B) the L-curve we have used in Examples 1 and 3, respectively, to choose the optimal regularization parameter β. Note that in the Figure 5.2 (B), $\mathcal{J}(\sigma_a, \sigma_s)$ simplifies to $\|\sigma_a\|^2_{\mathcal{H}^1}$ since we reconstruct only σ_a and have chosen $\sigma_a^0 = 0$ in that case.

Another important issue is to choose an appropriate weight ϵ in the regularization functional defined in (2.12). This weight is necessary because, in practice, σ_s takes values that are about two orders of magnitude larger than σ_a. The weight is used to bring the two terms in the regularization functional to the same level so that the regularization term has an effect on both σ_a and σ_s. In all our numerical simulations in section 5, we choose ϵ to be the ratio $(\sigma_a^b/\sigma_s^b)^2$, where σ_a^b and σ_s^b are optical properties of background media.

Note that the main purpose for introducing the scaling factor ϵ is to reduce computational complexity in choosing optimal regularization parameters. It is by no means the only way to perform regularizations on both absorption and scattering coefficients. For example, one can regularize σ_a and σ_s separately, which will involve selecting multiple regularization parameters such as those in [10, 26, 45].

We remark finally that the $\mathcal{H}^1$-norm we use in the regularization functional can be replaced by other norms or seminorms. For example, we have performed reconstructions on numerical Example 1 in section 5.3 with stricter bounds on σ_a and $\|\nabla\sigma_a\|_{L^2}$ (instead of $\|\sigma_a\|_{\mathcal{H}^1}$) as the regularization functional. We have obtained very similar results (although the optimal regularization parameter changes). The main reason for us to use the $\mathcal{H}^1$-norm is that in many practical applications, we want to find solutions near some reference (σ_a^0, σ_s^0), for example, some known background.

4.4. Cost of the numerical method. The computational cost of our method consists of two main parts. The first part is the evaluation of the objective function and its gradient in the optimization process. The second part is the updating of the BFGS matrices and vectors.

The costs of the function evaluation and of its gradient scale linearly with the number of optical sources N_q. Since each forward problem and its corresponding adjoint problem cost about the same, each gradient calculation (about $2N_q$ forward solves) is approximately twice as expensive as a function evaluation (about N_q forward solves). The cost in updating BFGS matrices and vectors can be neglected compared to function and gradient evaluations. The reason is that BFGS vectors (in $\mathbb{R}^{2N}$) are dramatically smaller than the vectors appearing in the forward and adjoint problems (in $\mathbb{C}^{JN}$).

In our computations, we store the nonzero elements of the matrix $\mathbf{A} - \mathbf{S}$ by using the compressed row storage scheme [9] whenever it is possible to do so. When it is not possible to store $\mathbf{A} - \mathbf{S}$, we store $\mathbf{A}$, $\mathbf{C}_0$, $\mathbf{K}$, and $\mathbf{D}_0$ defined in (3.12) and (3.13). This requires much less memory, but with the price that extra effort has to be paid to evaluate matrix vector products in GMRES. We use the following procedure to compute $\mathbf{Y} \equiv (\mathbf{A} - \mathbf{S})\mathbf{X}$ for any vector $\mathbf{X}$ with the same structure as $\mathbf{U}$ in (3.11):

1. For $j = 1, \ldots, J$,
   - Compute $\mathbf{X}'_j = \mathbf{D}_0 \mathbf{X}_j$;
   - Compute $\mathbf{X}''_j = \mathbf{C}_0 \mathbf{X}_j$;
   - Compute $\mathbf{X}''_j = \mathbf{X}''_j + \mathbf{A}_j \mathbf{X}_j$;
2. For $j = 1, \ldots, J$, $\mathbf{Y}_j = \mathbf{X}''_j - \sum_{j'} \mathbf{K}_{jj'} \mathbf{X}'_j$.

We prefer to store the matrix $\mathbf{A} - \mathbf{S}$ because it saves computational time when matrix-vector products are calculated. In all the numerical examples shown in the following section, we were able to store $\mathbf{A} - \mathbf{S}$. Note that the storage requirement does not increase with the number of sources (N_q) because we solve the transport equation (and its adjoint) with different sources sequentially. The storage cost of BFGS vectors can be neglected compared with the storage of the forward and adjoint matrices and vectors.

5. Numerical examples. We provide in this section four numerical examples that illustrate the performance of our numerical method. In the first example, we reconstruct the spatial distribution of the absorption coefficient while keeping the scattering coefficient fixed. In the second example, the spatially varying scattering coefficient is reconstructed, while the absorption coefficient is fixed. We then show an example in which both optical properties are reconstructed simultaneously. All the first three examples are done in two-dimensional settings. In the fourth example, we show three-dimensional simultaneous reconstructions of both optical properties.

5.1. Domain partition and measurement setup. For our two-dimensional simulations, we consider a computational domain of size 2×2 cm^2, denoted by $\overline{\mathcal{D}} \equiv \mathcal{D} \cup \partial\mathcal{D} = [0, 2] \times [0, 2]$. We cover the domain by 80×80 cells of uniform size whose nodes are given by

$$\mathcal{D}_h = \{\mathbf{x}_{i,j} = (x_i, y_j),\ x_i = i\Delta x,\ y_j = j\Delta y,\ i, j = 0, 1, \ldots, 80\},$$

with $\Delta x = \Delta y = 0.025$. The direction space is discretized into 128 uniformly distributed (over $[0, 2\pi)$) directions with identical quadrature weight:

$$S^1_{\Delta\theta} = \{\boldsymbol{\theta}_i : \boldsymbol{\theta}_i = (i-1) * \Delta\theta,\ i = 1, \ldots, 128\},$$

where $\Delta\theta = 2\pi/128$. The above discretizations yield a total number of 819,200 unknowns for one forward problem (solving for $\mathbf{U}$ in (3.10)), which is also true for the corresponding adjoint problem ($\boldsymbol{\Psi}$ in (3.21)). In all two-dimensional simulations, four sources ($N_q = 4$) are used. They are located at $(0, 1)$, $(1, 0)$, $(2, 1)$, and $(1, 2)$, respectively. For each sources, 20 detectors ($N_d = 20$) are used. The detectors are uniformly distributed along the sides of the square.

For the three-dimensional simulation, we consider a cylindrical domain given by

$$\overline{\mathcal{D}} := \{(x, y, z) : x^2 + y^2 \leq 1; 0 \leq z \leq 2\}.$$

We cover the domain by 22,022 tetrahedral elements. For the integration over S^2, we employ the full level symmetric S_8 discrete ordinate set of [44]. This set consists of 80 directions. A total number of 16 sources ($N_q = 16$) are used in the simulation. The sources are uniformly distributed on the two circles defined by $\Gamma^q_i = \{(x, y, z) : x^2 + y^2 = 1;\ z = z_i\}$ ($i = 1, 2$), where $z_1 = 0.5$, $z_2 = 1.5$. We arrange 8 layers of detectors on the boundary of the domain. Each layer has 16 detectors Those detectors are located at $\Gamma^d_j = \{(x, y, z) : x^2 + y^2 = 1;\ z = z_j\}$ ($j = 1, \ldots, 8$), with $z_j = 0.3 + (j-1) * 0.2$. Note that some detectors are placed on the same xy-plane as sources (but they do not overlap each other).

5.2. Generating synthetic data. In all the numerical reconstructions presented in section 5, the "measurements" are synthetic rather than obtained from real experimental data. A straightforward way of generating synthetic data is to use the same discretization for the forward model and for the inversion procedure. This may lead to somewhat simplified inversions of the finite-dimensional problem, which is often referred to as [illegible] in the inverse problems community [15, p. 133]. To avoid committing these "crimes," the data are generated with a discretization about twice as fine (in all variables) as the discretization used in the inversions.

In the following sections, our "exact data" will thus be those obtained from the fine mesh calculations. By noisy data, we mean the "exact data" polluted by additional multiplicative noise. Let z_d be the exact data; the "noisy" data are simulated according to the rule $z_d^\delta = (1 + \delta * \text{random}) z_d$, where "random" is a uniformly distributed random variable in $[-1, 1]$ and $\delta \geq 0$ will vary in our numerical simulations.

It should be noted that the "exact data" seen on the coarse grid used in the inversion actually already contain some "noise" because they were generated by the fine mesh calculations.

5.3. Single parameter reconstructions. We start with a simpler case where only one optical parameter needs to be reconstructed. Such reconstructions are often useful in practical applications. For example, it is generally believed that changes in the oxygenation of tissues correspond mainly to changes in the absorption property of tissues [58].

[illegible] 1: [illegible] σ_a. We first reconstruct a small absorbing disc centered at (1.15 cm, 1.15 cm) of radius 0.2 cm embedded in the two-dimensional computational domain. The optical parameters for the background and the disc are $\sigma_a = 0.1$ cm^{-1}, $\sigma_s = 80$ cm^{-1} and $\sigma_a = 0.2$ cm^{-1}, $\sigma_s = 80$ cm^{-1}, respectively. The anisotropy factor $g = 0.90$. Those are typical parameter values as they are encountered in biomedical applications. The modulation frequency of the source is $\omega = 600$ MHz. Each reconstruction here takes approximately 5 hours on a 2.4 GHz Pentium XEON processor.

Figure 5.1 shows the real absorption map and the reconstructed maps for different noise levels. We also provide in Figure 5.2 (A) (solid line) the evolution of the normalized objective function versus the iteration step in the case of reconstructions using noise-free synthetic data. Note that although the total number of BFGS iterations can be reduced by using a stricter line search scheme in the optimization algorithm, the total computational time remains almost constant. We have tested many classical line search algorithms. The results shown here use the one that works best for our application.

The quality of the reconstructions is measured as follows. Denote by $M^o \in \mathbb{R}^N$ ($M^r \in \mathbb{R}^N$) an exact (reconstructed) quantity, which can be either the absorption or the scattering map. We then define the relative l^2 error between M^o and M^r by

$$\mathcal{E}_{l^2} = \frac{\|M^r - M^o\|_{l^2}}{\|M^o\|_{l^2}} := \frac{\sqrt{\sum_{i=1}^N (M_i^r - M_i^o)^2}}{\sqrt{\sum_{i=1}^N (M_i^o)^2}}. \tag{5.1}$$

The quality of the reconstruction in the case of an absorbing disc is given by the parameters shown in the second column of Table 5.1. In Figure 5.2 (B), we display the L-curve we use to choose the optimal parameter β in the above reconstructions. We mention again that due to the acquisition of our data presented in section 5.2, the noise-free data actually contain "noise" on the coarse grid.

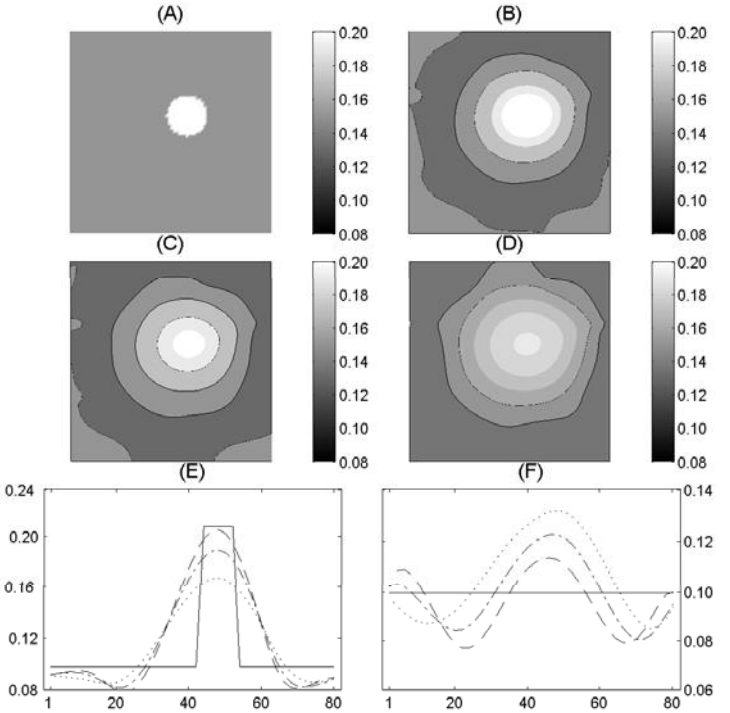


FIG. 5.1. *Maps of the reconstructed absorption coefficient* σ_a $[cm^{-1}]$ *in Example* 1. (A) *Real absorption map.* (B) *Reconstructed absorption map with exact synthetic data.* (C) *Reconstruction with* 10% *random noise.* (D) *Reconstruction with* 20% *random noise.* (E) *Cross sections of map* (A) *(solid line),* (B) *(dashed line),* (C) *(dash-dotted line), and* (D) *(dotted line) along the bottom left to top right diagonal.* (F) *Same as* (E) *except that the cross section is along* $y = 0.4$.

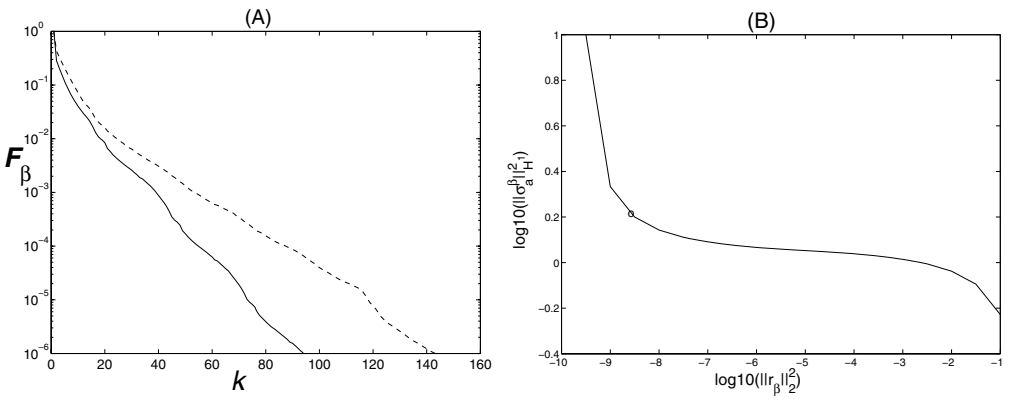


FIG. 5.2. (A) *Evolution of normalized objective functional* $\mathcal{F}_\beta$ *with respect to the number of iteration steps* k *for Example* 1. *Solid line: reconstruction of an absorbing disc; dashed line: reconstruction of a scattering disc.* (B) *L-curve used to choose optimal regularization parameter* β *for reconstruction with noise-free data in the reconstruction of an absorbing disc. The circle* (○) *denotes the place where* β *is chosen. Note that here* $\mathcal{J} = \|\sigma_a\|^2_{\mathcal{H}^1}$ *because we reconstruct only* σ_a *and we have set* $\sigma_a^0 = 0$.

TABLE 5.1
Optimal regularization parameters β and errors in reconstructions for different cases in Examples 1 *and* 2.

Cases	Absorbing disc			Scattering disc		
Noise level	0%	10%	20%	0%	10%	20%
$\beta \times 10^8$	1.0	2.8	4.6	1.3	2.9	5.7
$\mathcal{E}_{l^2}$	0.064	0.069	0.072	0.065	0.074	0.080

The reconstructions are classical examples of what we may expect in the field. Because of the severe ill-posedness of the inverse problem, it is difficult to reconstruct localized objects unless additional information is included in the reconstruction (which we do not want to do here). Note that the center and the integral of the absorption inclusion are more or less well reconstructed, even if the localized absorption map is somewhat smeared over a relatively large domain (whose size increases as the noise level, whence the regularization parameter β, increases). This is consistent with results obtained from asymptotic theories; see, for instance, [7].

2: σ_s. We now replace the highly absorbing disc in the previous case with a highly scattering disc in the same position and with the same size. The optical parameters for the background and the disc are $\sigma_a = 0.1$ cm^{-1}, $\sigma_s = 70$ cm^{-1} and $\sigma_a = 0.1$ cm^{-1}, $\sigma_s = 80$ cm^{-1}, respectively. Again, the anisotropy factor $g = 0.9$, and the modulation frequency $\omega = 600$ MHz. Each reconstruction here takes approximately 6 hours on a 2.4 GHz Pentium XEON processor. Figure 5.3 shows the exact scattering map and the reconstructed maps for different noise levels. Error estimates are presented in the third column of Table 5.1.

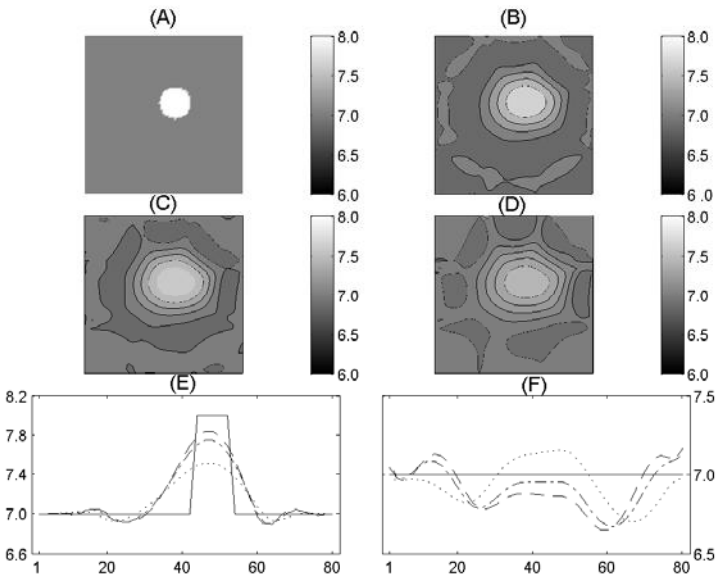


FIG. 5.3. *Maps of reconstructed reduced scattering coefficients $\sigma_s' = (1-g)\sigma_s$ [cm^{-1}] for Example* 1. (A) *Real scattering map.* (B) *Reconstructed scattering map with noise-free synthetic data.* (C) *Reconstruction with* 10% *random noise.* (D) *Reconstruction with* 20% *random noise.* (E) *Cross sections of map* (A) *(solid line),* (B) *(dashed line),* (C) *(dash-dotted line), and* (D) *(dotted line) along the bottom left to top right diagonal.* (F) *Same as* (E) *except that the cross section is along $y = 0.4$.*

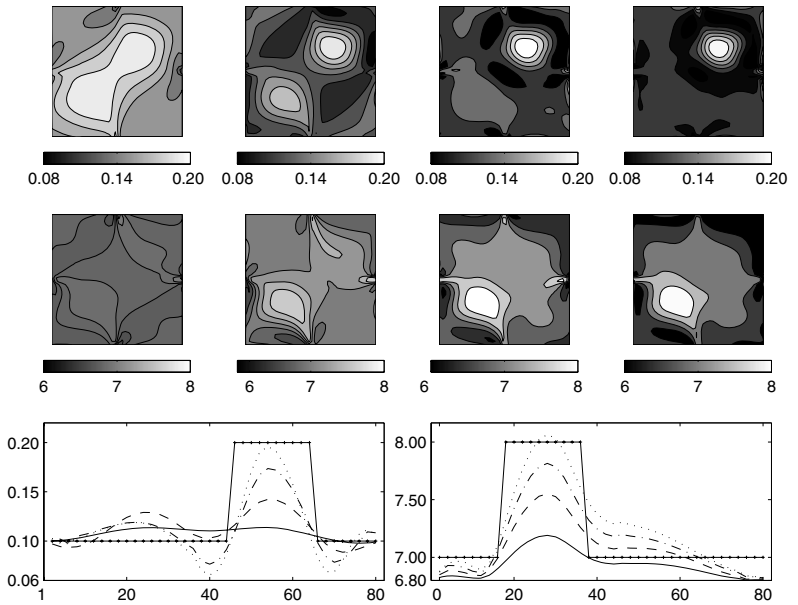


FIG. 5.4. *Top row: Maps of reconstructed absorption coefficients* σ_a $[cm^{-1}]$ *at BFGS iteration* $k = 40$, 80, 120, *and* 156 *(final), respectively, for the frequency domain reconstruction in Example* 3. *Middle row: same as the top row but for the reduced reconstructed scattering coefficients* $\sigma_s' = (1-g)\sigma_s$ $[cm^{-1}]$. *Bottom row: cross section of real (solid line with +) and reconstructed absorption maps (left), reduced scattering maps (right) along the diagonal at iterations* $k = 40$ *(solid line),* 80 *(dashed line),* 120 *(dash-dotted line), and* 156 *(dotted line). The reconstructions are done with noise-free synthetic data.*

The quality of the reconstructions is very similar in the above two examples and is quite satisfactory. In spite of the fact that stronger regularizations have to be imposed as the noise level increases, the localization and the estimate for the optical parameters in the presence of moderate noise indeed allow us to obtain reasonably accurate information toward diagnostic purposes.

5.4. Frequency-domain versus steady-state. One of the main reasons for introducing frequency-domain reconstructions is that they allow for a better separation between the scattering and absorption properties of the inclusions [48]. Diffusion-based theories show that both coefficients cannot be reconstructed simultaneously without additional geometrical hypotheses [5, 33]. We now show in two numerical examples that frequency-domain data indeed substantially improve the reconstruction of both coefficients.

[illegible] 3: [illegible]. We reconstruct here in the square domain two small discs of radius 0.2 cm and centered at (1.35 cm, 1.35 cm) and (0.65 cm, 0.65 cm), respectively. The first disc is highly absorbing and the second one is highly scattering. Optical properties for the two discs are $\sigma_a = 0.2$ cm^{-1}, $\sigma_s = 70$ cm^{-1} and $\sigma_a = 0.1$ cm^{-1}, $\sigma_s = 80$ cm^{-1}, respectively. The background parameters are $\sigma_a = 0.1$ cm^{-1} and $\sigma_s = 70$ cm^{-1}. As before, $g = 0.9$ and $\omega = 600$ MHz. Each reconstruction takes approximately 8 hours on a 2.4 GHz Pentium XEON processor.

We compare the reconstructions based on the frequency-domain ERT with those based on the steady-state ERT. The latter is obtained by setting the frequency $\omega = 0$

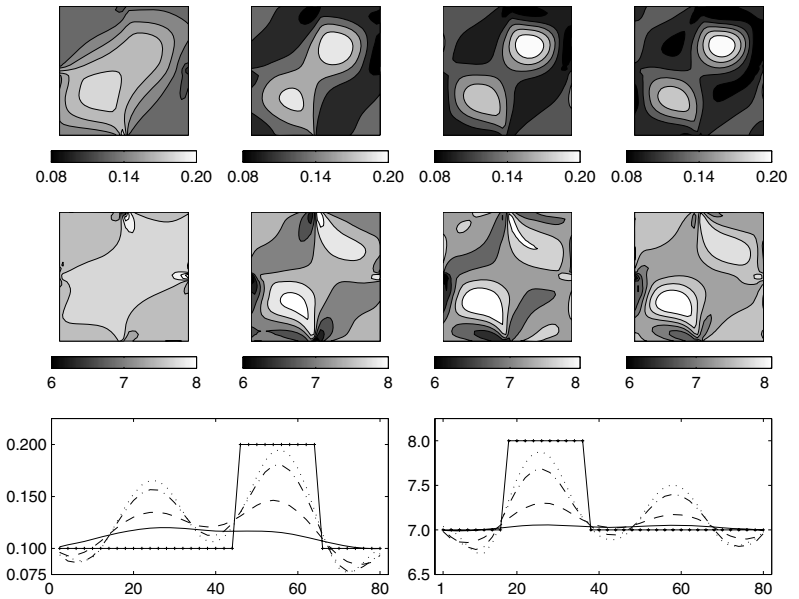


FIG. 5.5. *Top row: Maps of reconstructed absorption coefficients* σ_a $[cm^{-1}]$ *at BFGS iteration* $k = 40$, 80, 120, *and* 354 *(final), respectively, for the steady-state reconstruction in Example* 3. *Middle row: same as the top row but for the reduced reconstructed scattering coefficients* $\sigma_s' = (1-g)\sigma_s$ $[cm^{-1}]$. *Bottom row: cross section of real (solid line with +) and reconstructed absorption maps (left), reduced scattering maps (right) along the diagonal at iterations* $k = 40$ *(solid line),* 80 *(dashed line),* 120 *(dash-dotted line), and* 354 *(dotted line). The reconstructions are done with noise-free synthetic data.*

TABLE 5.2

Error estimates for the reconstructions of Example 3 (*E*3) *and Example* 4 (*E*4) *for several iteration steps* (k) *in the optimization process. Here, "f" refers to frequency-domain calculations and "s" to steady-state calculations.*

		$k = 40$		$k = 80$		$k = 120$		Final	
		σ_a	σ_s	σ_a	σ_s	σ_a	σ_s	σ_a	σ_s
*E*3	$\mathcal{E}_{l^2}(f)$	0.121	0.144	0.092	0.112	0.080	0.092	0.063	0.076
	$\mathcal{E}_{l^2}(s)$	0.181	0.224	0.127	0.132	0.113	0.112	0.094	0.106
*E*4	$\mathcal{E}_{l^2}(f)$	0.194	0.252	0.173	0.208	0.147	0.189	0.131	0.171
	$\mathcal{E}_{l^2}(s)$	0.342	0.422	0.287	0.366	0.245	0.322	0.210	0.292

in our formulation and keeping everything else the same. Note that we observe a slight difference (below 5%) in the computational times required for the case when $\omega = 0$ and the case when $\omega \neq 0$. We present in Figures 5.4 and 5.5 the reconstructions obtained by the frequency-domain method and the steady-state method, respectively. We also list the parameters which measure the quality of the reconstructions at different iteration steps in Table 5.2.

4: . In the last numerical test, we show simultaneous reconstructions of two optical properties in a three-dimensional setting. We try to reconstruct a small cylinder $\overline{\mathcal{D}}_s = \{(x, y, z) : (x - 0.5)^2 + y^2 \leq 0.2^2,\ 0 \leq z \leq 2\}$ embedded in the cylindrical domain defined in section 5.1. Both the absorption and the scattering coefficients of the small cylinder are

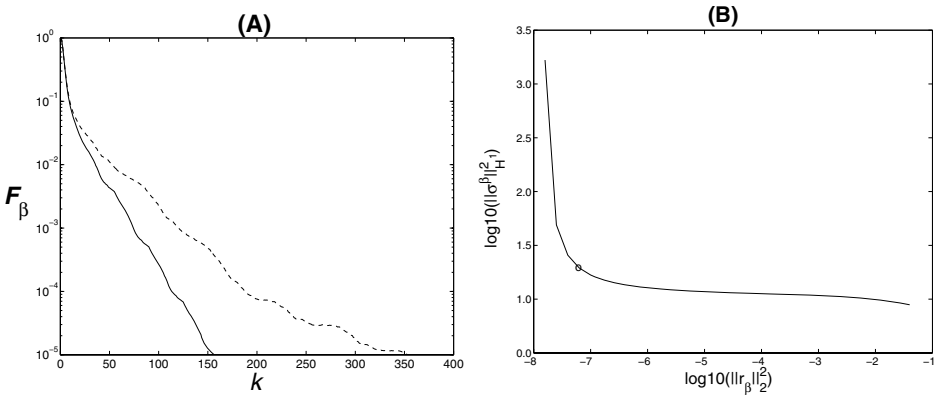


FIG. 5.6. (A) *Evolution of the normalized objective functional $\mathcal{F}_\beta$ with respect to the number of iteration steps k for Example* 3. *Solid line: frequency domain reconstruction of both coefficient simultaneously; dashed line: steady-state reconstruction of both coefficients simultaneously.* (B) *L-curve used to choose optimal regularization parameter β for reconstruction with noise-free data in the frequency domain simultaneous reconstruction of absorbing and scattering coefficients. The circle* ($\circ$) *denotes the place where β is chosen. Note that $\mathcal{J} \equiv \|\boldsymbol{\sigma}^\beta\|^2_{\mathcal{H}^1} := \|\sigma_a^\beta\|^2_{\mathcal{H}^1} + \epsilon\|\sigma_s^\beta\|^2_{\mathcal{H}^1}$ because we have set $\sigma_a^0 = \sigma_s^0 = 0$.*

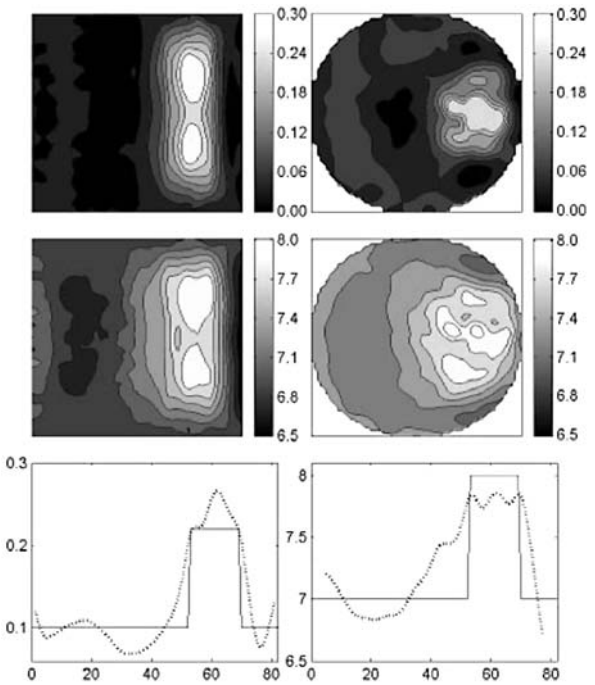


FIG. 5.7. *Top row: xy cross section (at $z = 1$) and xz cross section (at $y = 0$) of the reconstructed absorption coefficient σ_a $[cm^{-1}]$ for the frequency-domain reconstruction in Example* 4. *Middle row: same as the top row but for the reduced reconstructed scattering coefficients $\sigma_s' = (1-g)\sigma_s$ $[cm^{-1}]$. Bottom left: real (solid) and reconstructed (dotted) absorption coefficient along line $y = 0$, $z = 1$. Bottom right: same as bottom left but for reconstructed reduced scattering coefficient.*

different from those of the background. This is different from the case in Example 3 where absorption and scattering anomalies are located at different places. Optical properties for the small cylinder are $\sigma_a = 0.2$ cm^{-1} and $\sigma_s = 80$ cm^{-1}, while those for the background are $\sigma_a = 0.1$ cm^{-1} and $\sigma_s = 70$ cm^{-1}. The anisotropy factor $g = 0.9$ and the modulation frequency $\omega = 600$ MHz. Each reconstruction takes approximately 22 hours on a 3 GHz Pentium XEON processor. As in Example 3, we compare the frequency-domain reconstructions with the steady-state reconstructions. Cross sections of reconstructions obtained by the frequency-domain method and the steady-state method are presented in Figures 5.7 and 5.8, respectively. Qualities of the reconstructions at different iteration steps are again listed in Table 5.2.

We first observe that in both the two-dimensional (Example 3) and the three-dimensional (Example 4) reconstructions, the frequency-domain reconstruction converges faster (in terms of BFGS iterations) than the steady state reconstruction; see, for example, the results in Figure 5.6 (A) and Table 5.2. This has been confirmed in many other geometrical settings we have tested: the speed of convergence of the steady-state reconstruction presented here is one of the most favorable we have obtained, whereas the speed of convergence of the frequency-domain reconstructions was very often similar to what we have presented here.

As far as quality of the reconstruction is concerned, we observe a significant improvement in the frequency domain reconstructions compared to the steady-state

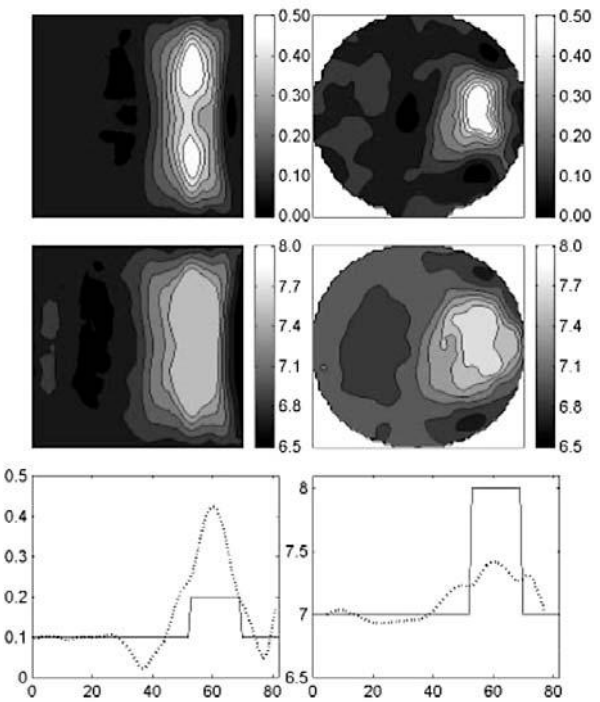


FIG. 5.8. *Top row: xy cross section (at $z = 1$) and xz cross section (at $y = 0$) of the reconstructed absorption coefficient σ_a $[cm^{-1}]$ for the steady-state reconstruction in Example 4. Middle row: same as the top row but for the reduced reconstructed scattering coefficients $\sigma_s' = (1-g)\sigma_s$ $[cm^{-1}]$. Bottom left: real (solid) and reconstructed (dotted) absorption coefficient along line $y = 0$, $z = 1$. Bottom right: same as bottom left but for reconstructed reduced scattering coefficient.*

reconstructions. In all simulations, the stopping criterion is the same:

$$\frac{\mathcal{F}_\beta^k(\sigma_a, \sigma_s)}{\mathcal{F}_\beta^0(\sigma_a, \sigma_s)} \leq 10^{-5}.$$

Although the L^2 errors may not enjoy a dramatic improvement (see Table 5.2), they are still significantly reduced. More importantly, the last rows in Figures 5.4 and 5.5 show dramatic reductions (at least by a factor of 2) of the cross-talk between the absorption and scattering reconstructions: the spurious bumps (left of the left picture on the bottom row and right of the right picture on the bottom row in Figures 5.4 and 5.5) are clearly much stronger in the steady-state calculations than in the frequency-domain calculations. This is the major advantage of the frequency-domain calculations. Very similar phenomena are observed in the three-dimensional simulations; see Figures 5.7 and 5.8. Although the absorption coefficient is still overestimated in the frequency-domain case (last row of Figure 5.7), it is better than the situation in the steady-state case (last row of Figure 5.8) where the absorption coefficient σ_a is severely overestimated, while the reduced scattering coefficient σ_s' is severely underestimated. As predicted by theory, we have observed that an increase in ω led to reduced cross-talk. How much this effect depends on the choice of the frequency (as well as on possible combinations of different frequencies) and on the geometrical setting will be explored in future works.

6. Concluding remarks. We have formulated an inverse problem in optical tomography as a regularized least-squares problem based on the frequency-domain equation of radiative transfer to model light propagation in biological tissues. In the inversion procedure, the forward model is discretized by using a finite volume method and a discrete ordinates method. We solve the regularized least-squares problem by using a limited-memory quasi-Newton method with BFGS-type updating rule for the Hessian matrix and have incorporated positivity constraints and L^∞ bounds on the optical parameters. Numerical reconstructions based on synthetic data provide results that are in agreement with the expected reconstructions. Notably, the crosstalk between the two optical parameters is significantly reduced in frequency-domain reconstructions.

The method presented here also overcomes several of the shortcomings of diffusion-equation-based optical tomography [4, 16], which provides a very useful tool in many problems but fails to adequately model strongly absorbing regions (e.g., large blood-filled spaces such as brain hematoma), low-scattering void-like inclusions (e.g., spaces filled with cerebrospinal fluid, amniotic fluid, or synovial fluid), and optically relatively thin media such as fingers and small animals. Main domains of application of the proposed method will be functional imaging of rheumatoid arthritis of human finger joints and small animal imaging [30, 32], where the diffusion equation typically fails to generate accurate forward predictions. Extensive numerical reconstructions in practically relevant geometries for these applications and based on possibly multifrequency experimental data will be performed in the near future.

Acknowledgment. We would like to thank the two anonymous referees for their many constructive comments and suggestions.

REFERENCES

[1] M. L. Adams and E. W. Larsen, *Fast iterative methods for discrete-ordinates particle transport calculations*, Prog. Nucl. Energy, 40 (2002), pp. 3–150.

[2] V. Agoshkov, *Boundary Value Problems for the Transport Equations*, Birkhäuser, Boston, 1998.
[3] R. Aronson, R. L. Barbour, J. Lubowsky, and H. Graber, *Application of transport theory to infra-red medical imaging*, in Modern Mathematical Methods in Transport Theory, W. Greenberg and J. Polewczak, eds., Birkhäuser, Basel, 1991, pp. 64–75.
[4] S. R. Arridge, *Optical tomography in medical imaging*, Inverse Problems, 15 (1999), pp. R41–R93.
[5] S. R. Arridge and W. R. B. Lionheart, *Nonuniqueness in diffusion-based optical tomography*, Opt. Lett., 23 (1998), pp. 882–884.
[6] G. Bal, *Transport through diffusive and nondiffusive regions, embedded objects, and clear layers*, SIAM J. Appl. Math., 62 (2002), pp. 1677–1697.
[7] G. Bal, *Optical tomography for small volume absorbing inclusions*, Inverse Problems, 19 (2003), pp. 371–386.
[8] G. Bal and K. Ren, *Generalized diffusion model in optical tomography with clear layers*, J. Opt. Soc. Amer. A, 20 (2003), pp. 2355–2364.
[9] R. Barrett, M. Berry, T. F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. van der Vorst, *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*, SIAM, Philadelphia, 1994.
[10] M. Belge, M. Kilmer, and E. Miller, *Efficient determination of multiple regularization parameters in a generalized L-curve framework*, Inverse Problems, 18 (2002), pp. 1161–1183.
[11] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu, *A limited memory algorithm for bound constrained optimization*, SIAM J. Sci. Comput., 16 (1995), pp. 1190–1208.
[12] M. T. Chahine, *Inverse problems in radiative transfer: Determination of atmospheric parameters*, J. Atmos. Sci., 27 (1970), pp. 960–967.
[13] B. Chance, R. R. Alfano, B. J. Tromberg, and A. Katzir, eds., *Optical Tomography and Spectroscopy of Tissue*, Vol. V, SPIE, Bellingham, WA, 2003.
[14] M. Choulli and P. Stefanov, *Reconstruction of the coefficients of the stationary transport equation from boundary measurements*, Inverse Problems, 12 (1996), pp. L19–L23.
[15] D. Colton and R. Kress, *Inverse Acoustic and Electromagnetic Scattering Theory*, Springer-Verlag, New York, 1998.
[16] R. Dautray and J.-L. Lions, *Mathematical Analysis and Numerical Methods for Science and Technology*, Vol. 6, Springer-Verlag, Berlin, 1993.
[17] H. Dehghani, D. T. Delpy, and S. R. Arridge, *Photon migration in non-scattering tissue and the effects on image reconstruction*, Phys. Med. Biol., 44 (1999), pp. 2879–2906.
[18] O. Dorn, *A transport-backtransport method for optical tomography*, Inverse Problems, 14 (1998), pp. 1107–1130.
[19] O. Dorn, *Scattering and absorption transport sensitivity functions for optical tomography*, Optics Express, 7 (2000), pp. 492–506.
[20] R. A. Elliott, T. Duracz, N. J. McCormick, and D. R. Emmons, *Experimental test of a time-dependent inverse radiative transfer algorithm for estimating scattering parameters*, J. Opt. Soc. Amer. A, 5 (1988), pp. 366–373.
[21] H. W. Engl, M. Hanke, and A. Neubauer, *Regularization of Inverse Problems*, Kluwer Academic, Dordrecht, The Netherlands, 1996.
[22] R. Eymard, T. Gallouët, and R. Herbin, *Finite volume methods*, in Handbook of Numerical Analysis VII, P. Ciarlet and J. L. Lions, eds., North-Holland, Amsterdam, 2000, pp. 713–1020.
[23] M. Firbank, S. R. Arridge, M. Schweiger, and D. T. Delpy, *An investigation of light transport through scattering bodies with non-scattering regions*, Phys. Med. Biol., 41 (1996), pp. 767–783.
[24] L. Fukshansky, N. Fukshansky-Kazarinova, and A. M. v. Remisowsky, *Estimation of optical properties in a living tissue by solving the inverse problem of the multiflux radiative transfer*, Appl. Opt., 30 (1991), pp. 3145–3153.
[25] F. Golse, S. Jin, and C. D. Levermore, *The convergence of numerical transfer schemes in diffusive regimes* I: *Discrete-ordinate method*, SIAM J. Numer. Anal., 36 (1999), pp. 1333–1369.
[26] C. Gu and G. Wahba, *Minimizing GCV/GML scores with multiple smoothing parameters via the Newton method*, SIAM J. Sci. Statist. Comput., 12 (1991), pp. 383–398.
[27] E. Haber, U. Ascher, and D. Oldenburg, *On optimization techniques for solving nonlinear inverse problems*, Inverse Problems, 16 (2000), pp. 1263–1280.
[28] P. C. Hansen and D. P. O'Leary, *The use of the L-curve in the regularization of discrete ill-posed problems*, SIAM J. Sci. Comput., 14 (1993), pp. 1487–1503.

[29] L. G. Henyey and J. L. Greenstein, *Diffuse radiation in the galaxy*, Astrophys. J., 90 (1941), pp. 70–83.
[30] A. Hielscher, A. Bluestone, G. Abdoulaev, A. Klose, J. Lasker, M. Stewart, U. Netz, and J. Beuthan, *Near-infrared diffuse optical tomography*, Disease Markers, 18 (2002), pp. 313–337.
[31] A. H. Hielscher, R. E. Alcouffe, and R. L. Barbour, *Comparison of finite-difference transport and diffusion calculations for photon migration in homogeneous and heterogeneous tissue*, Phys. Med. Biol., 43 (1998), pp. 1285–1302.
[32] A. H. Hielscher, A. D. Klose, A. Scheel, B. Moa-Anderson, M. Backhaus, U. Netz, and J. Beuthan, *Sagittal laser optical tomography for imaging of rheumatoid finger joints*, Phys. Med. Biology, 49 (2004), pp. 1147–1163.
[33] V. Isakov, *Inverse Problems for Partial Differential Equations*, Springer-Verlag, New York, 1998.
[34] H. Jiang, K. D. Paulsen, U. L. Österberg, B. W. Pogue, and M. S. Patterson, *Simultaneous reconstruction of optical absorption and scattering maps turbid media from near-infrared frequency-domain data*, Opt. Lett., 20 (1995), pp. 2128–2130.
[35] C. T. Kelley, *Iterative Methods for Optimization*, Frontiers Appl. Math. 18, SIAM, Philadelphia, 1999.
[36] A. Kienle, F. K. Forster, and R. Hibst, *Influence of the phase function on determination of the optical properties of biological tissue by spatially resolved reflectance*, Opt. Lett., 26 (2001), pp. 1571–1573.
[37] A. D. Kim and J. B. Keller, *Light propagation in biological tissue*, J. Opt. Soc. Amer. A, 20 (2003), pp. 92–98.
[38] A. D. Klose, U. Netz, J. Beuthan, and A. H. Hielscher, *Optical tomography using the time-independent equation of radiative transfer. Part* 1: *Forward model*, J. Quant. Spectrosc. Radiat. Transfer, 72 (2002), pp. 691–713.
[39] A. D. Klose and A. H. Hielscher, *Iterative reconstruction scheme for optical tomography based on the equation of radiative transfer*, Med. Phys., 26 (1999), pp. 1698–1707.
[40] A. D. Klose and A. H. Hielscher, *Optical tomography using the time-independent equation of radiative transfer. Part* 2: *Inverse model*, J. Quant. Spectrosc. Radiat. Transfer, 72 (2002), pp. 715–202.
[41] A. D. Klose and A. H. Hielscher, *Quasi-Newton methods in optical tomographic image reconstruction*, Inverse Problems, 19 (2003), pp. 387–409.
[42] E. W. Larsen, *Solution of the inverse problem in multigroup transport theory*, J. Math. Phys., 22 (1981), pp. 158–160.
[43] E. W. Larsen, *Solution of three dimensional inverse transport problems*, Transport Theory Statist. Phys., 17 (1988), pp. 147–167.
[44] E. E. Lewis and W. F. Miller, *Computational Methods of Neutron Transport*, American Nuclear Society, La Grange Park, IL, 1993.
[45] A. Li, E. Miller, M. Kilmer, T. Brukilacchio, T. Chaves, J. Stott, Q. Zhang, T. Wu, M. Chorlton, R. Moore, D. Kopans, and D. Boas, *Tomographic optical breast imaging guided by* 3*-D mammography*, Appl. Opt., 42 (2003), pp. 5181–5190.
[46] E. H. Lieb and M. Loss, *Analysis*, 2nd ed., AMS, Providence, RI, 2001.
[47] V. Markel and J. Schotland, *Effects of sampling and limited data in optical tomography*, App. Phys. Lett., 81 (2002), pp. 1180–1182.
[48] T. O. McBride, B. W. Pogue, U. L. Österberg, and K. D. Paulsen, *Separation of absorption and scattering heterogeneities in NIR tomographic imaging of tissue*, in OSA Technical Digest, Biomedical Topical Meetings, Optical Society of America, Washington, DC, 2000,
[49] N. J. McCormick, *Recent developments in inverse scattering transport methods*, Transport Theory Statist. Phys., 13 (1984), pp. 15–28.
[50] N. J. McCormick, *Inverse radiative transfer problems: A review*, Nuclear Sci. Engrg., 112 (1992), pp. 185–198.
[51] M. Mokhtar-Kharroubi, ed., *Mathematical Topics in Neutron Transport Theory: New Aspects*, Ser. Adv. Math. Appl. Sci. 46, World Scientific, Singapore, 1997.
[52] J. Nocedal and S. J. Wright, *Numerical Optimization*, Springer-Verlag, New York, 1999.
[53] B. W. Patton and J. P. Holloway, *Application of preconditioned GMRES to the numerical solution of the neutron transport equation*, Ann. Nuclear Energy, 29 (2002), pp. 109–136.
[54] K. Ren, G. S. Abdoulaev, G. Bal, and A. H. Hielscher, *Algorithm for solving the equation of radiative transfer in the frequency domain*, Opt. Lett., 29 (2004), pp. 578–580.
[55] Y. Saad, *Iterative Methods for Sparse Linear Systems*, 2nd ed., SIAM, Philadelphia, 2003.
[56] Y. Saad and M. H. Schultz, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 856–869.

[57] J. C. Schotland, *Continuous-wave diffusion imaging*, J. Opt. Soc. Amer. A, 14 (1997), pp. 275–279.

[58] B. J. Tromberg, N. Shah, R. Lanning, A. Cerussi, J. Espinoza, T. Pham, L. Svaasand, and J. Butler, *Non-invasive in vivo characterization of breast tumors using photon migration spectroscopy*, Neoplasia, 2 (2000), pp. 26–40.

[59] C. R. Vogel, *Non-convergence of the L-curve regularization parameter selection method*, Inverse Problems, 12 (1996), pp. 535–547.

[60] C. R. Vogel, *Computational Methods for Inverse Problems*, Frontiers Appl. Math. 23, SIAM, Philadelphia, 2002.

[61] A. P. Wang and S. Ueno, *An inverse problem in a three-dimensional radiative transfer*, Astrophys. Space Sci., 155 (1989), pp. 105–111.

[62] A. J. Welch and M. J. C. Van-Gemert, *Optical-Thermal Response of Laser Irradiated Tissue*, Plenum Press, New York, 1995.

[63] A. G. Yodh and B. Chance, *Spectroscopy and imaging with diffusing light*, Phys. Today, 48 (1995), pp. 34–40.

[64] C. Zhu, R. H. Byrd, P. Lu, and J. Nocedal, *L-BFGS-B–FORTRAN subroutines for large-scale bound constrained optimization*, ACM Trans. Math. Software, 23 (1997), pp. 550–560.

SIAM J. SCI. COMPUT.
Vol. 28, No. 4, pp. 1490–1508
© 2006 Society for Industrial and Applied Mathematics

MOVING MESH METHODS FOR SINGULAR PROBLEMS ON A SPHERE USING PERTURBED HARMONIC MAPPINGS*

YANA DI†, RUO LI†, TAO TANG‡, AND PINGWEN ZHANG†

Abstract. This work is concerned with developing moving mesh strategies for solving problems defined on a sphere. To construct mappings between the physical domain and the logical domain, it has been demonstrated that harmonic mapping approaches are useful for a general class of solution domains. However, it is known that the curvature of the sphere is positive, which makes the harmonic mapping on a sphere not unique. To fix the uniqueness issue, we follow Sacks and Uhlenbeck [*Ann. of Math.* (2), 113 (1981), pp. 1–24] to use a perturbed harmonic mapping in mesh generation. A detailed moving mesh strategy including mesh redistribution and solution updating on a sphere will be presented. The moving mesh scheme based on the perturbed harmonic mapping is then applied to the moving steep front problem and the Fokker–Planck equations with high potential intensities on a sphere. The numerical experiments show that with a moderate number of grid points our proposed moving mesh algorithm can accurately resolve detailed features of singular problems on a sphere.

Key words. moving mesh methods, singularity, harmonic mapping, perturbed harmonic mapping, spherical domain

AMS subject classifications. 65M50, 65N22, 58E20

DOI. 10.1137/050642514

1. Introduction.

In this work, we will consider efficient moving mesh methods for solving singular problems on a sphere. Partial differential equations (PDEs) defined on a sphere are seen in many practical cases such as polymer rheological analysis, ocean modeling, and global climate simulation. The moving mesh technique is one of the natural choices for solving problems with localized moving singularities. Examples include moving internal boundaries or interfaces separating different regions which are present in many problems in nature, such as interfaces separating two different fluids in a multiphase flow, and between solid and liquid regions in a solidification process. Such interfaces are in general dynamic, i.e., they evolve in time. Moving mesh methods have been proved powerful for solving time-dependent PDEs in two-dimensional domains and simple three-dimensional domains [2, 3, 4, 5, 24, 30, 31, 34].

In general, a moving mesh method always involves a logical domain and a physical domain, together with a transformation between them. In [12], Dvinsky suggests that harmonic function theory may provide a general framework for developing useful mesh generators. His method can be viewed as a generalization and extension of Winslow's method [33]. However, unlike most other generalizations which add terms

*Received by the editors October 12, 2005; accepted for publication (in revised form) March 21, 2006; published electronically September 15, 2006.

http://www.siam.org/journals/sisc/28-4/64251.html

†LMAM & School of Mathematical Sciences, Peking University, Beijing 100871, People's Republic of China (dyn@math.pku.edu.cn, rli@math.pku.edu.cn, pzhang@math.pku.edu.cn). The research of the first two authors was supported in part by the Joint Applied Mathematics Research Institute between Peking University and Hong Kong Baptist University. The research of the fourth author was partially supported by the special funds for Major State Research Projects (2005CB1704) and the National Science Foundation of China for Distinguished Young Scholars (10225103).

‡Mathematics Department, Hong Kong Baptist University, Kowloon Tong, Hong Kong (ttang@math.hkbu.edu.hk). The research of this author was supported by CERG grants of Hong Kong Research Grant Council, FRG grants of Hong Kong Baptist University, NSAF grant 10476032 of the National Science Foundation of China, and by the International Research Team on Complex System of Chinese Academy of Sciences.

or functionals to the basic Winslow grid generator, his approach uses a single functional to accomplish the adaptive mapping. The critical points of this functional are [illegible]. Meshes obtained by Dvinsky's method enjoy desirable properties of harmonic maps, particularly regularity, or smoothness [15, 27]. Motivated by the work of Dvinsky, a moving mesh finite element strategy based on harmonic mapping was proposed and studied by the authors in [21]. The key idea of this strategy is to construct the harmonic map between the physical space and a parameter space by an iteration procedure.

For two-dimensional domains or three-dimensional cubic domains as considered in [21, 22], the metric on the logical domain is Euclidean. Consequently, the construction of the harmonic mapping only solves some linear equations. However, this is not the case for spherical problems. In fact, if the physical domain is a sphere, then it is natural to set the logical domain as a sphere rather than a convex polygon as used in [21]. In this case, the fact that the curvature of a sphere is positive makes harmonic mappings on spheres not unique. Another difficulty is that the metric on the unit sphere is not an identity anymore, which is different from the cases considered before; see, e.g., [21]. Our goal is to construct a transformation which preserves the good properties of harmonic mappings such as smoothness and uniqueness. Moreover, a scheme for solving the nonlinear equation induced by the metric on a sphere will be designed.

Motivated by the work of Sacks and Uhlenbeck [26], we will use the so-called perturbed harmonic mapping to construct a unique mapping between the physical and logical spaces. It will be seen that perturbed harmonic mappings are useful for handling spherical domains. The nonlinear Euler–Lagrange equations from minimizing the mesh energy functional will be solved by a local Gauss–Seidel-like iterative method. The numerical experiments indicate that a satisfactory approximation to the perturbed harmonic mapping can be obtained using only a few iteration steps. Furthermore, a key step for the moving mesh method is to obtain an improved new mesh based on the underlying numerical solutions and the old meshes. With the spherical geometry, the mesh redistribution has to be designed carefully: efforts have to be made to ensure that the mesh movement occurs only on the sphere, and that all the grid points stay on the sphere during the whole process of mesh redistribution. In this work, appropriate mesh redistribution and solution updating strategies will be investigated.

The rest of the paper is arranged as follows. In section 2, the theory of the perturbed harmonic mapping on a sphere will be briefly reviewed. Section 3 describes the general framework of the moving mesh method on a sphere, including some details of the mesh-redistribution and solution-updating strategies. In section 4, the proposed moving mesh scheme will be applied to solve moving shocks on a sphere and the Fokker–Planck equations derived from liquid crystal simulations. For the latter problem, the solutions of the problem tend to a delta function when the strength of the excluded volume potential is large (see [20]). With a uniform mesh, an extremely fine mesh is required in order to obtain reasonable resolution, but the efficiency can be enhanced significantly by using moving mesh methods. Some concluding remarks will be given in the final section.

2. Perturbed harmonic mapping. Let Ω and Ω_c be compact Riemannian manifolds of dimension n with metric tensors d_{ij} and $r_{\alpha\beta}$ in some local coordinates $\vec{x}$ and $\vec{\xi}$, respectively. The energy density of a mapping $\vec{\xi}: \vec{x} \mapsto \vec{\xi}$ is defined by

$$e(\vec{\xi}) = G^{ij} g_{\alpha\beta} \frac{\partial \xi^\alpha}{\partial x^i} \frac{\partial \xi^\beta}{\partial x^j}, \tag{2.1}$$

where the standard summation convention is assumed. The corresponding energy functional is then defined by

$$E(\vec{\xi}) = \int_\Omega e(\vec{\xi})dx. \tag{2.2}$$

The harmonic mapping is the mapping which minimizes the above energy functional.

Existence and uniqueness of the harmonic map are guaranteed, provided the Riemannian curvature of Ω_c is nonpositive and its boundary convex (see Hamilton [15] and Schoen and Yau [27]). However, this result does not apply to the case when the physical and logical domains are spheres, i.e., $\Omega = \Omega_c = \mathcal{S}^2$; in this case, the Riemannian curvature of Ω_c is positive. In fact, the energy functional (2.2) has the conformal invariance [26], which leads to nonuniqueness of the harmonic mappings. To recover the uniqueness, we follow the work of Sacks and Uhlenbeck [26] to employ a so-called perturbed harmonic mapping approach. More precisely, let

$$E_\alpha(\vec{\xi}) = \int_\Omega (1 + e(\vec{\xi}))^\alpha dx, \tag{2.3}$$

with $\alpha > 1$. It is noted that when $\alpha = 1$ the functionals (2.3) and (2.2) are equivalent. It is known that the Sobolev space of mappings,

$$L_1^{2\alpha}(M, N) = \{\xi \in L_1^{2\alpha}(M, R^k) : \xi(x) \in N\} \subset C^0(M, N), \tag{2.4}$$

is a C^2 separable Banach manifold for $\alpha > 1$ [25]. In particular, it is shown in [26] that if $\alpha > 1$, then the critical mappings of E_α in $L_1^{2\alpha}(M, N)$ are C^∞ and are unique in the sense of isometric transformation. Such regularity and uniqueness results provide a sound theoretical background for the moving mesh method based on the perturbed harmonic mappings.

The Euler–Lagrange equation of the energy functional (2.3) is given by

$$-\Delta_{\mathrm{L-B}}\xi^l - (\alpha - 1)\frac{\nabla|\nabla\xi|^2 \cdot \nabla\xi^l}{|\nabla\xi|^2} + [A(\xi)(d\xi, d\xi)]^l = 0, \tag{2.5}$$

where $\Delta_{\mathrm{L-B}}$ stands for the Laplace–Beltrami operator,

$$\Delta_{\mathrm{L-B}} f = \frac{1}{\sqrt{G}} \sum_{i,j} \frac{\partial}{\partial x^i} \left(\sqrt{G} G^{ij} \frac{\partial f}{\partial x^j} \right),$$

and $A(\xi)$ is the second fundamental form of Ω_c. For a sphere, it is known that

$$A(\xi)(d\xi, d\xi) = -|\nabla\xi|^2 \xi. \tag{2.6}$$

Moreover, the derivatives in curvilinear coordinates are given by

$$(\nabla\xi^m)^j = G^{ij} \frac{\partial \xi^m}{\partial x^i},$$
$$\left(\nabla|\nabla\xi|^2\right)^j = G^{ij} \frac{\partial}{\partial x^i} \left(\sum_t G^{kl} \frac{\partial \xi^t}{\partial x^k} \frac{\partial \xi^t}{\partial x^l} \right).$$

Consequently, the Euler–Lagrange equation (2.5) can be written in the following divergence form:

$$-\nabla \cdot \left(\alpha |\nabla\xi|^{2(\alpha-1)} \nabla\xi^l \right) + \alpha |\nabla\xi|^{2(\alpha-1)} [A(\xi)(d\xi, d\xi)]^l = 0. \tag{2.7}$$

Using the Stokes law, we can get the weak form of the above equation:

$$\int_{\mathcal{S}^2} \left\{ \sum_l |\nabla\xi|^{2(\alpha-1)} \nabla\xi^l \cdot \nabla\lambda^l - \sum_l |\nabla\xi|^{2\alpha} \xi^l \lambda^l \right\} dV = 0 \qquad \forall \lambda \in L_1^{2\alpha}(M,N). \tag{2.8}$$

The way of obtaining the mapping for the perturbed equation (2.7) is similar to that used for obtaining the conventional harmonic mappings described in [21, 22]. It is shown that with the harmonic mappings the metric distortion can be minimized [13, 26]. Moreover, it is expected that the mesh associated with the mapping (2.7) has certain smoothness and uniqueness.

In this work, the perturbed harmonic mapping will be used to generate adaptive meshes for unstructured triangles in spherical domains. We point out that there are several other successful ways for adaptive remeshing also by minimizing the mesh energy function that depends on the local physical scales; see, e.g., Anderson, Zheng, and Cristini [1], Liao et al. [23], Sun and Huang [17], and Wang and Wang [32].

3. Moving mesh methods for spherical domain. Consider the time-dependent PDEs on a sphere $\Omega = \mathcal{S}^2$,

$$\Psi_t = \mathcal{L}(\Psi) \qquad \text{in } \Omega \times (0,T], \tag{3.1}$$

where $\mathcal{L}$ is some spatial partial differential operator, with initial value

$$\Psi|_{t=0} = \Psi_0 \qquad \text{in } \Omega. \tag{3.2}$$

Let us consider a moving mesh method based on the perturbed harmonic mappings to solve this time-dependent problem. We triangulate the domain Ω into a mesh $\mathcal{T}_h$ with spherical triangular elements. Equation (3.1) is discretized with finite element approximation on the mesh $\mathcal{T}_h$. Assume a PDE solver is given so that the numerical solution at time level t_{n+1} can be obtained by using the numerical approximation at $t = t_n$. Since our mesh varies with respect to time, we denote the mesh at time step t_n as $\mathcal{T}_h^{(n)}$. The procedure to combine the mesh motion and the PDE solutions is illustrated below:

- [illegible] 0. [illegible] Set $n = 0$ and $t_0 = 0$. Project the initial function on the initial mesh $\mathcal{T}_h^{(0)}$, which gives $\Psi_h^{(0)}$.
- [illegible] 1 [illegible] Obtain the numerical solutions at $t = t_{n+1}$ on the mesh $\mathcal{T}_h^{(n)}$ by an appropriate PDE solver to obtain $\Psi_h^{(n+1)}$.
- [illegible] 2. [illegible] Check the mesh quality by certain criteria; if satisfactory, set $\mathcal{T}_h^{(n+1)} = \mathcal{T}_h^{(n)}$, $n = n+1$, and go to Step 1.
- [illegible] 3. [illegible] Obtain a more appropriate mesh by some mesh moving techniques, as described later in this section.
- [illegible] 4. [illegible] Update the numerical solution $\Psi_h^{(n+1)}$ on the new mesh obtained in Step 3 and go to Step 2.

We point out that the mesh quality (mentioned in Step 2 above) takes into account the shape, alignment, and adaptation features of the elements. For example, the minimum angle, the maximum angle, and the aspect ratio have been used widely to characterize the mesh quality. More details can be found in Huang [18].

3.1. Finite element space on sphere. The sphere is triangulated into a mesh with spherical triangular elements. The Voronoi spherical geodesic grid (see, e.g., [11]) is used in our work. Here we give a finite element space for the discretization

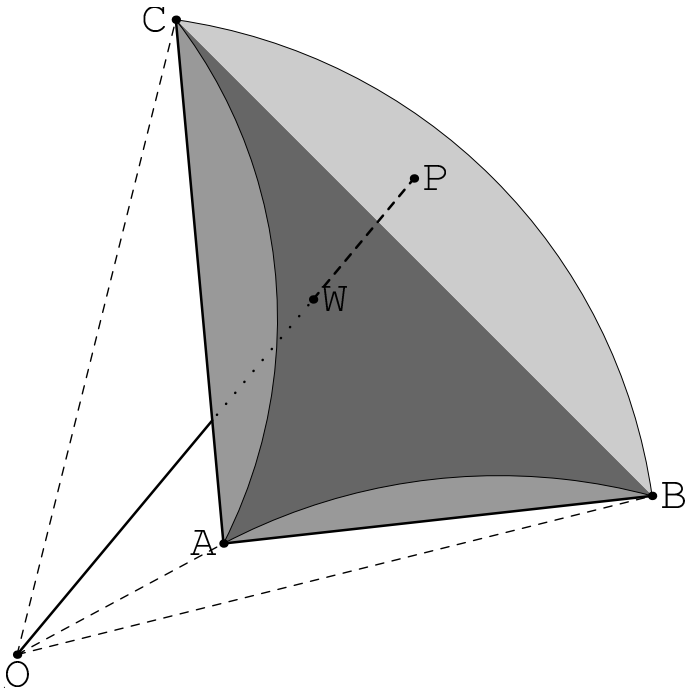


FIG. 1. *The shape function defined on a spherical triangle.*

of the problem (3.1)-(3.2). We will project the linear basis functions of the triangles on polyhedron onto the spherical geodesic triangles. The vertices (counter-clockwise) of a spherical geodesic triangle K are denoted by A, B, and C. The plane triangle K_p has the same vertices. For a point $P \in K$, the intersection point of line $\overrightarrow{OP}$ and K_p is denoted by W (see Figure 1). The three basis functions on spherical geodesic triangle K are defined by

$$\begin{aligned} \lambda_1(\vec{x}) &= \frac{1}{D}(\vec{w} \cdot \vec{a}_2 \times \vec{a}_3), \\ \lambda_2(\vec{x}) &= \frac{1}{D}(\vec{w} \cdot \vec{a}_3 \times \vec{a}_1), \\ \lambda_3(\vec{x}) &= \frac{1}{D}(\vec{w} \cdot \vec{a}_1 \times \vec{a}_2), \end{aligned} \tag{3.3}$$

where

$$\begin{aligned} &\overrightarrow{OA} = \vec{a}_1, \ \ \overrightarrow{OB} = \vec{a}_2, \ \ \overrightarrow{OC} = \vec{a}_3, \ \ \overrightarrow{OP} = \vec{x}, \\ &\overrightarrow{OW} = \vec{w}, \ \ D = (\vec{a}_1 \cdot \vec{a}_2 \times \vec{a}_3), \ \ \vec{w}(\vec{x}) = D\frac{\vec{x}}{\vec{x} \cdot \vec{d}}, \\ &\vec{d} = \vec{a}_1 \times \vec{a}_2 + \vec{a}_2 \times \vec{a}_3 + \vec{a}_3 \times \vec{a}_1. \end{aligned}$$

With this notation, a finite element space on sphere is presented:

$$H_h \triangleq \left\{ \Psi_h \in C^0(\Omega) : \ \Psi_h|_K \in \operatorname{span}\{\lambda_1, \lambda_2, \lambda_3\} \, \forall K \right\}. \tag{3.4}$$

3.2. Mesh redistribution on sphere. A key step for the moving mesh method is to obtain an improved new mesh based on the underlying numerical approximations. In our computations, we simply choose the given physical domain, i.e., the sphere, as the logical domain, and we use the Voronoi spherical geodesic grid (see, e.g., [11]) to generate a quasi-uniform mesh with triangular elements on the logical domain. Obviously, this also gives the initial partition for the physical domain. The nodes in the physical mesh are denoted by X_i, $1 \le i \le N$, and the nodes in the logical mesh by Ξ_i, $1 \le i \le N$. Initially, since we choose the logical mesh the same as the physical mesh, we have $\Xi_i = X_i$. The physical mesh will be redistributed at each time step, while the logical mesh will be kept unchanged during the entire process of the numerical computations. For each time step, by numerically solving the Euler–Lagrange equation (2.7), we can obtain a discrete transformation between the physical domain $\vec{x}_h$ and the logical domain $\vec{\xi}_h$. The logical mesh is then given such that $\Xi_i = \vec{\xi}_h(X_i)$. The metric on the physical domain is given according to the given PDEs and their numerical solution, which is called the [illegible].

On the sphere, there are no local coordinates to cover the whole domain $\mathcal{S}^2$. Therefore, we have to choose suitable local coordinates. The metric for the logical domain is the natural metric of sphere as a submanifold embedded in $\mathbb{R}^3$. Since the Euler–Lagrange equation (2.7) is nonlinear, it will be solved numerically by a point-by-point iterative approach. Assume that we are considering a node in the physical mesh, X_k, whose corresponding node in the logical mesh is denoted by Ξ_k. Denote the set of triangles in the physical domain with X_k as a vertex by $\mathbf{S}_k$ and the corresponding set in the physical domain by $\mathbf{S}_{c,k}$. We will set the local coordinate around X_k in the physical domain and the local coordinate around Ξ_k in the logical domain. It is natural to eliminate one component whose absolute value is the largest among the three components. This will give two-dimensional local coordinates. For example, for $X_k = (x_k^1, x_k^2, x_k^3)$ set, if $|x_k^1| \ge |x_k^2|$ and $|x_k^1| \ge |x_k^3|$, then we will let $(x_k^1, x_k^2) \leftarrow (x_k^2, x_k^3)$ to be the local coordinate around X_k. The same strategy is employed to choose the local coordinate around Ξ_k. We then have the Euler–Lagrange equation (2.7) around the node Ξ_k:

$$\begin{aligned} \nabla \cdot \left(|\nabla \xi|^{2(\alpha-1)} \nabla \xi^l \right) + |\nabla \xi|^{2\alpha} \xi^l = 0 \quad \text{for } l = 1, 2, 3, \\ \xi|_{\partial \mathbf{S}_k} = \xi_b. \end{aligned} \tag{3.5}$$

In the physical domain, around X_k the patch of surface on the unit sphere can be expressed as

$$\left(x^3\right)^2 = 1 - \left(x^1\right)^2 - \left(x^2\right)^2 , \tag{3.6}$$

and the metric of the physical domain is taken as a submanifold embedded in $\mathbb{R}^3$:

$$\tilde{G}^{ij}_{i,j=1,2} = \begin{pmatrix} 1 - \left(x^1\right)^2 & x^1 x^2 \\ x^1 x^2 & 1 - \left(x^2\right)^2 \end{pmatrix}. \tag{3.7}$$

The metric G^{ij} on the physical domain around X_k is then set as

$$G^{ij} = m(u_h)\tilde{G}^{ij}, \tag{3.8}$$

where $m(u_h)$ is the monitor function given for the problem under consideration. The choice of the monitor function m is very subtle, which is an indicator of the solution

singularity and is problem dependent. A general discussion of the monitor function can be found in [6, 29].

Now on the patch of triangles $\mathbf{S}_k$, we will solve a Dirichlet boundary-value problem for (3.5), with linear boundary mapping segmentwise on $\partial\mathbf{S}_k$ determined by the vertex mapping $\Xi_n = \vec{\xi}(X_n)\ \forall X_n \in \mathbf{S}_k, n \neq k$. The mapping in $\mathbf{S}_k$ is approximated by a piecewise linear mapping on every triangle in $\mathbf{S}_k$. Thus the mapping can be given only if $\vec{\xi}(X_k)$ is given. Using the standard Galerkin approximation for (3.5) in $\mathbf{S}_k$, we have only one test function, which is the pyramid-like basis function located at X_k. This gives

$$\int_{\mathbf{S}_k} \left\{ |\nabla\xi|^{2(\alpha-1)} \nabla\xi^l \cdot \nabla\lambda^k - |\nabla\xi|^{2\alpha} \xi^l \lambda^k \right\} dV = 0 \quad \text{for } l = 1,\, 2,\, 3, \tag{3.9}$$

where λ^k is the test function for X_k, which is an algebraic system with order $2\alpha + 1$ for three unknowns. We will solve (3.9) approximately. On $\mathbf{S}_k$, we denote $\vec{\xi}_h$ by

$$\vec{\xi}_h = \sum_n \Xi_n \lambda_h^n + \Xi_k \lambda^k\,, \tag{3.10}$$

where the summation for n is taken over all vertices in $\mathbf{S}_k$ except X_k. By taking $|\nabla\xi|$ in (3.9) explicitly, i.e., ξ_h in $|\nabla\xi|$ takes the currently available Ξ_k as its value at X_k, we then get an explicit approximation for (3.9):

$$\Xi_k^l \approx \frac{\int_{\mathbf{S}_k} \sum_n |\nabla\xi|^{2\alpha} \Xi_n^l \lambda^l \lambda^k dV - \sum_n \int_{\mathbf{S}_k} |\nabla\xi|^{2(\alpha-1)} G^{ij} \Xi_n^l \frac{\partial\lambda^n}{\partial x^i} \frac{\partial\lambda^k}{\partial x^j} dV}{\int_{\mathbf{S}_k} \big(- |\nabla\xi|^{2\alpha} \lambda^k \lambda^k + |\nabla\xi|^{2(\alpha-1)} G^{ij} \frac{\partial\lambda^k}{\partial x^i} \frac{\partial\lambda^k}{\partial x^j} \big) dV}. \tag{3.11}$$

Below we will describe the algorithm for mesh redistribution, which is similar to the one proposed in [21]. In the first step, we iterate for k from 0 to N, using (3.11) to update all logical nodes Ξ_k until convergence (i.e., the difference between the obtained Ξ_k and initial (fixed) $\Xi_k^{(0)}$ is smaller than a given tolerance). The tolerance for the convergence criteria is of the order $\mathcal{O}(h_l)$, where h_l is the typical element size in the logical mesh. The converged result is denoted by $\Xi_k^\star$, and we then get the difference between $\Xi_k^\star$ and the initial logical mesh as

$$\delta\Xi_k = \Xi_k^{(0)} - \Xi_k^\star. \tag{3.12}$$

In the second step, we use the above difference to obtain the mesh redistribution information in the physical mesh difference. Again we will choose the same local coordinate as described above. For a node X_k in the physical mesh, we first compute the mesh motion vector:

$$\delta X_k^l = \frac{\sum_{K\in\mathbf{S}_k} |E| \left.\frac{\partial x^l}{\partial\vec{\xi}}\right|_K \delta\Xi_k}{\sum_{K\in\mathbf{S}_k} |E|} \qquad \text{for } l = 1, 2, \tag{3.13}$$

where the Jacobian matrix $\partial\vec{x}/\partial\vec{\xi}\big|_K$ is computed by using the fact that the linear element is used (see also [21]):

$$\begin{pmatrix} \Xi_{E_1}^1 - \Xi_{E_0}^1 & \Xi_{E_2}^1 - \Xi_{E_0}^1 \\ \Xi_{E_1}^2 - \Xi_{E_0}^2 & \Xi_{E_2}^2 - \Xi_{E_0}^2 \end{pmatrix} \begin{pmatrix} \dfrac{\partial x^1}{\partial\xi^1} & \dfrac{\partial x^1}{\partial\xi^2} \\ \dfrac{\partial x^2}{\partial\xi^1} & \dfrac{\partial x^2}{\partial\xi^2} \end{pmatrix} = \begin{pmatrix} X_{E_1}^1 - X_{E_0}^1 & X_{E_2}^1 - X_{E_0}^1 \\ X_{E_1}^2 - X_{E_0}^2 & X_{E_2}^2 - X_{E_0}^2 \end{pmatrix}; \tag{3.14}$$

and E_0, E_1, and E_2 are the subscripts of the three vertices of the element K in the physical mesh. The coordinate in the third dimension is excluded, as explained earlier (i.e., with the way of choosing the local coordinate). On the other hand, to ensure that the mesh movement occurs only on the sphere, the third component can be recovered by using

$$\delta X_k^3 \to \operatorname{sgn}(X_k^3)\sqrt{1-\left(X_k^1+\delta X_k^2\right)^2-\left(X_k^2+\delta X_k^2\right)^2}-X_k^3. \tag{3.15}$$

In the third step, we will control the magnitude of each redistribution to avoid mesh tangling. For the element $K \in \mathcal{T}_h$ with vertices E_0, E_1, and E_2, we solve the cubic equation (in terms of μ)

$$\det\begin{pmatrix} X_{E_0}+\mu\cdot\delta X_{E_0} & X_{E_1}+\mu\cdot\delta X_{E_1} & X_{E_2}+\mu\cdot\delta X_{E_2} \end{pmatrix}=0, \tag{3.16}$$

where δX_{E_k} is determined by (3.13). Denote the smallest positive root of (3.16) by μ_K. The mesh motion parameter is then set as

$$\mu^\star = \min_{K\in\mathcal{T}_h}\left\{1;\, \mu_K/2\right\}. \tag{3.17}$$

It is obvious that the mesh tangling can be avoided if we move the mesh using the formula $X_k \leftarrow X_k + \mu^\star \delta X_k$.

In the last step, we update the mesh moving direction by

$$\delta X_k \leftarrow \frac{X_k+\mu^\star\delta X_k}{|X_k+\mu^\star\delta X_k|}-X_k\,. \tag{3.18}$$

This is to make sure that all the grid points are moving along the sphere. The new mesh nodes are then given by $X_k+\delta X_k$. It can be verified that the new nodes are all on the sphere.

3.3. Solution interpolation on the new mesh. After redistributing the mesh in the physical domain Ω, we need to transfer the solution information on the old mesh to the new mesh. Let $\delta\vec{x} := \sum_{i=1}^N \delta X_i \lambda^i$, where λ^i is the basis function of H_h. Assume that a finite element solution $\Psi_h^{(n)} = \sum_{i=1}^N \Psi_i^{(n)}\lambda^i$ is given at $t=t_n$. Let τ be a virtual time variable. We will regard the mesh motion from the old mesh to the new mesh as a linear homotopy for τ goes from 0 to 1. Consequently, the mesh flow with the virtual time τ is $\vec{x}(\tau) \triangleq \vec{x}_{\text{old}} + \tau\delta\vec{x}$. For an arbitrary point $\vec{x}_0 \in \mathcal{T}_h$, the basis function $\lambda^i(\vec{x})$ associated with $\vec{x_0}$ will be varying linearly with τ. Since $\lambda^i(\vec{x})$ is defined on the physical mesh, it is denoted by $\lambda^i(\vec{x};\tau)$. Direct calculation shows that

$$\frac{d\lambda^i(\vec{x};\tau)}{d\tau} = \delta\vec{x}\cdot\nabla_{\vec{x}}\lambda^i(\vec{x};\tau). \tag{3.19}$$

Assume that the numerical solution on mesh $\vec{x}(\tau)$ is of the form

$$\Psi_h(\vec{x};\tau) = \sum_{i=1}^N \Psi_i(\tau)\lambda^i(\vec{x};\tau), \tag{3.20}$$

which takes the starting value $\Psi_h(\cdot;0) = \Psi_h^{(n)}$. The main idea of the solution updating is to let the variation of $\Psi_h(\cdot;\tau)$ with respect to τ vanish in the test function space:

$$\left\langle \frac{d\Psi_h(\cdot;\tau)}{d\tau}, \Phi_h \right\rangle = 0 \quad \forall \Phi_h \in H_h. \tag{3.21}$$

Using (3.19) and (3.20) gives

$$\sum_{i=1}^{N}\int_{\Omega}\left\{\frac{d\Psi_i}{d\tau}\lambda^i(\cdot;\tau)\lambda^j(\cdot;\tau)+\Psi_i\delta\vec{x}\cdot\nabla_{\vec{x}}\lambda^i(\cdot;\tau)\lambda^j(\cdot;\tau)\right\}dx=0,\quad 1\le j\le N. \tag{3.22}$$

The above ODE system can be solved by some appropriate ODE solver, e.g., a three-step Runge–Kutta method [21, 22].

4. Numerical experiments. We first discuss the perturbation parameter α in (3.5): in the numerical computations reported in this section it is chosen as 1.5. It is known [25] that α must be larger than 1 in order to ensure the uniqueness of the mapping. If α is close to 1, the critical mapping of E_α is close to a harmonic mapping. This implies that α should be chosen to be approximately 1. On the other hand, it is found that larger values of α increase the solution efficiency for (2.8). In fact, larger values of α can speed up the convergence of the mesh movement: note that the nonlinear Euler–Lagrange equations are solved by a local Gauss–Seidel-like iterative method. With a few numerical tests, it is found that one of the good choices for the parameter α should be approximately 1.5.

Example 4.1 (moving steep front on a sphere). Our first example is to compute a moving steep front along the direction $\vec{b}:=(1,1,1)^T$ governed by the equation

$$\partial\Psi_t+\Psi\vec{b}\cdot\nabla^*\Psi=\epsilon\Delta_{L-B}\Psi, \tag{4.1}$$

which is defined on the unit sphere, where ∇^* is the surface gradient. The initial condition is given by

$$\Psi(x,y,z;0)=\left(1+\exp\left(\frac{x+y+z+1}{2\epsilon}\right)\right)^{-1}. \tag{4.2}$$

We first briefly discuss the PDE algorithm to be used. For this example, a simple finite element discretization as described in section 3.1 is employed. The weak formulation of (4.1) reads as follows: Find $\Psi\in H^1(\Omega)$ such that

$$\int_{\Omega}\frac{\partial\Psi}{\partial t}vdx+\int_{\Omega}\Psi\vec{b}\cdot\nabla^*\Psi vdx=-\epsilon\int_{\Omega}\nabla^*\Psi\cdot\nabla^*vdx\quad\forall v\in H^1(\Omega). \tag{4.3}$$

In the finite element space H_h, the above weak form can be approximated as follows: Find $\Psi_h\in H_h$, such that

$$\int_{\Omega}\frac{\partial\Psi_h}{\partial t}v_hdx+\int_{\Omega}\Psi_h\vec{b}\cdot\nabla^*\Psi_hv_hdx=-\epsilon\int_{\Omega}\nabla^*\Psi_h\cdot\nabla^*v_hdx\quad\forall v_h\in H_h(\Omega). \tag{4.4}$$

For the temporal approximation, an explicit three-step Runge–Kutta scheme is applied; see, e.g., [7].

In our computations, the viscosity coefficient ϵ is chosen as 0.01. For the convection dominant problem, more mesh points are required around the wave front, which makes the use of the moving mesh methods meaningful. The monitor function m defined in (3.8) is taken as the standard gradient-based one, i.e.,

$$m(\Psi)=\sqrt{1+\beta|\nabla\Psi|^2}. \tag{4.5}$$

For problems with a moving steep front, the parameter β has to be set as a large number, say between 10^2 to 10^3; see, e.g., [6]. A large value of β can usually cluster

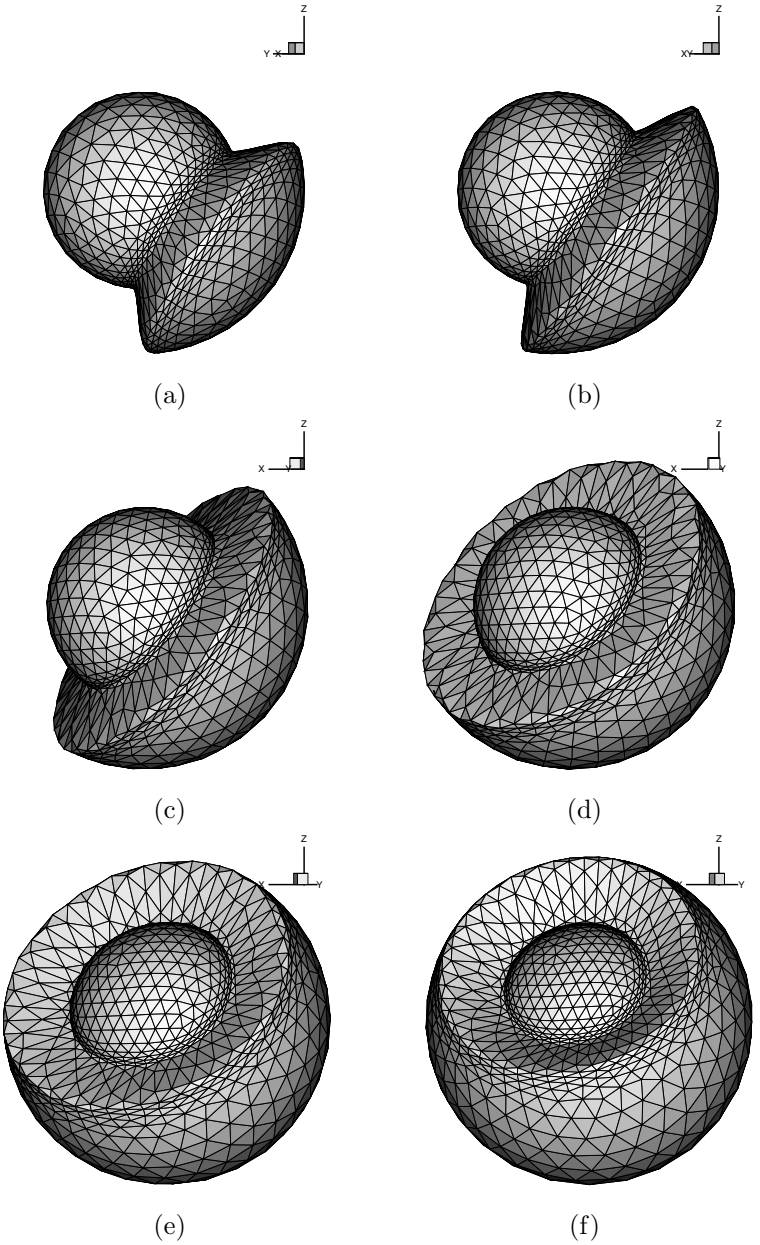


FIG. 2. *Example* 4.1. *The adaptive meshes on a sphere with* 800 *nodes from* $t = 0$ *to* $t = 1.5$.

more grid points around the steep front. On the other hand, if $|\nabla\Psi|$ in (4.5) is replaced by $|\nabla\Psi|/\max_\Omega|\nabla\Psi|$, then the parameter β can be chosen as an $\mathcal{O}(1)$ number; see, e.g., [7]. In our computations, the parameter β in (4.5) is chosen as 10^3.

In Figure 2, the moving meshes obtained by using our moving mesh scheme with 800 nodes (about 1600 triangular elements) are plotted. Together in this figure are the numerical solutions for Ψ at various time levels. Figure 2(a) shows the initial

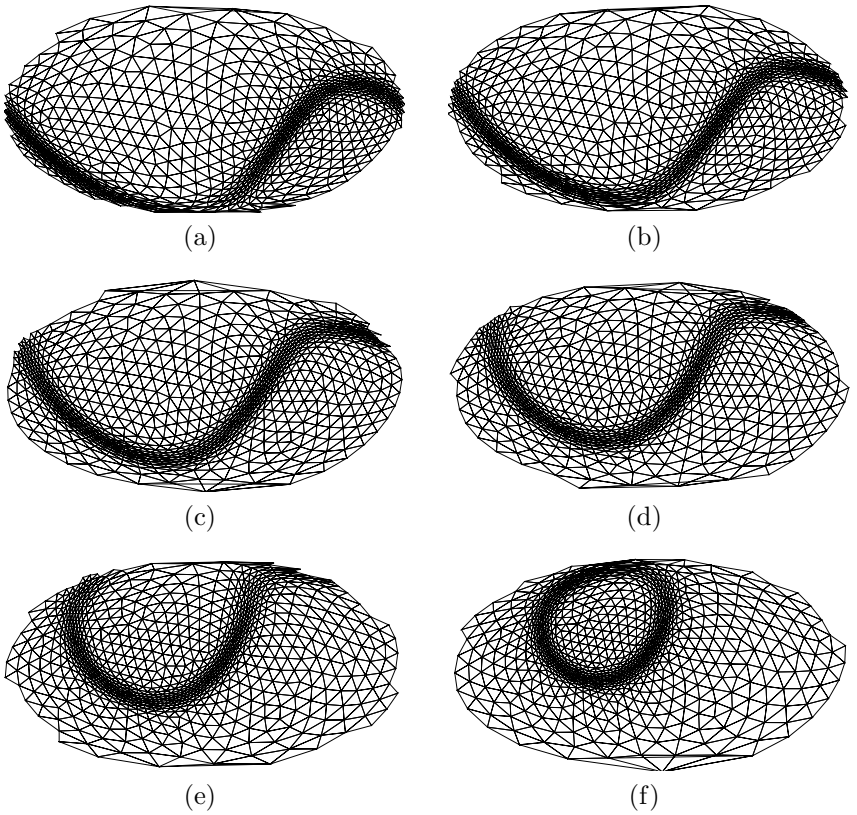


FIG. 3. *Example* 4.1. *The adaptive meshes on planisphere using* 800 *nodes from* $t = 0$ *to* $t = 1.5$.

state: the initial front is at $x+y+z+1 = 0$, where a stage can be seen. The solution is 0 on the top left side of the front and 1 on the bottom right side. It can be seen that the front moves along the direction $(1, 1, 1)^T$ on the sphere. In Figures 2(b)–(f), the area with solution value 1 becomes bigger, and the area of value 0 becomes smaller as time increases. At $t = 1.5$, the solution is mostly 1 on the sphere and is 0 only on a small concave area. In order to display the numerical solution more clearly, the figures are plotted from various visual angles at different time levels. Moreover, in order to have a better idea of the mesh distribution the meshes are projected to a planisphere and are plotted in Figure 3. It is seen that our moving mesh scheme adapts the mesh extremely well in the regions with large solution gradients.

Figure 4 shows the meshes and solutions obtained by the proposed moving mesh algorithm with 800 nodes (left) and by using a uniform mesh approach with 3200 nodes (right) at $t = 0.6$. It is observed from the figure that the moving mesh method with fewer nodes resolves the sharp front much more accurately than the corresponding uniform mesh approach with more nodes. Furthermore, the numerical oscillation in the uniform mesh solution is not seen in the moving mesh solution. In fact, it is found in our numerical computations that it requires more than 7000 nodes in a uniform mesh in order to obtain the same resolution of the moving mesh solution with 800 nodes.

4.2 (solution of the Fokker–Planck equation). The second example is

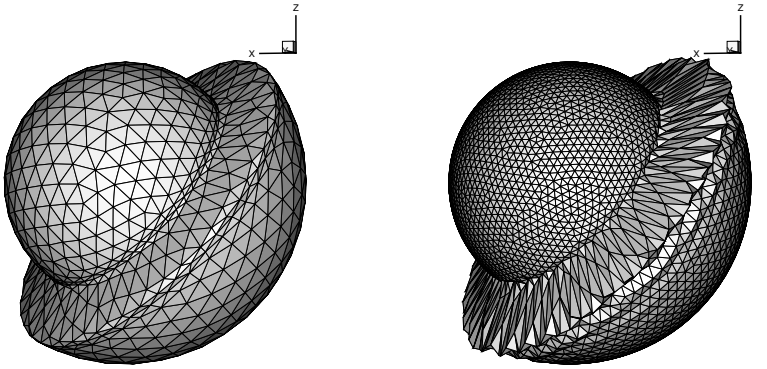


FIG. 4. *Example* 4.1. *The adaptive mesh on sphere with* 800 *nodes (left) and uniform mesh with* 3200 *nodes (right) at* $t = 0.6$.

concerned with the Doi model [9, 16], which is commonly used in studying rod-like polymer fluids. In this model, a homogeneous population of equal rods is described by an orientational distribution function $\Psi(\vec{x}, t)$. The evolution of $\Psi(\vec{x}, t)$ is modeled with the Smoluchowski equation or the Fokker–Planck equation [10]:

$$\frac{\partial \Psi}{\partial t} = \frac{1}{De}\mathcal{R} \cdot (\mathcal{R}\Psi + \Psi \mathcal{R} V_{ev}) \quad \text{on } \Omega = \mathcal{S}^2, \tag{4.6}$$

where De is the Deborah number, $\mathcal{R}$ is the rotational operator $\mathcal{R} \equiv \vec{x} \times \partial/\partial\vec{x}$, and V_{ev} is an effective excluded-volume potential which takes the Maier–Saupe form:

$$V_{ev}(\vec{x}) = \frac{3}{2} U \int_{\Omega} |\vec{x} \times \vec{x'}|^2 \Psi(\vec{x'}) dx'. \tag{4.7}$$

In (4.7), U is the nondimensional potential intensity which is proportional to the concentration of the polymers. The initial distribution function is taken as

$$\Psi(\vec{x}, 0) = \frac{1}{4\pi} + \frac{1}{16}\sqrt{\frac{5}{\pi}}\Big(3(\vec{x}_0 \cdot \vec{x})^2 - 1\Big),$$

where $\vec{x}_0$ is a constant unit vector.

As the potential intensity increases, the orientational distribution function becomes sharply peaked and behaves like a δ-function. To resolve the solutions in this case, it is necessary to use some adaptive grid methods. Here we will use the framework introduced in the last section, with a few additional details described below.

The weak formulation of (4.6) reads as follows: Find $\Psi \in H^1(\Omega)$, such that

$$\int_{\Omega} \frac{\partial \Psi}{\partial t} v dx + \frac{1}{De}\int_{\Omega} \mathcal{R}\Psi \cdot \mathcal{R} v dx = -\frac{1}{De}\int_{\Omega} \Psi \mathcal{R} V(\Psi) \cdot \mathcal{R} v dx \quad \forall v \in H^1(\Omega). \tag{4.8}$$

In the finite element space H_h, the above weak form can be approximated as follows: Find $\Psi_h \in H_h$, such that

$$\int_{\Omega} \frac{\partial \Psi_h}{\partial t} v_h dx + \frac{1}{De}\int_{\Omega} \mathcal{R}\Psi_h \cdot \mathcal{R} v_h dx = -\frac{1}{De}\int_{\Omega} \Psi_h \mathcal{R} V(\Psi_h) \cdot \mathcal{R} v_h dx \quad \forall v_h \in H_h(\Omega). \tag{4.9}$$

For the temporal approximation, similar to [7] we use a semi-implicit three-step Runge–Kutta scheme to solve (4.9): Find $\Psi_h^{n+1} \in H_h$, such that the following hold.

• , , 1.

$$\int_\Omega \frac{\Psi_h^{(1)}-\Psi_h^n}{\triangle t/3}v_h dx+\frac{1}{De}\int_\Omega \mathcal{R}\Psi_h^{(1)}\cdot\mathcal{R}v_h dx = -\frac{1}{De}\int_\Omega \Psi_h^n\mathcal{R}V(\Psi_h^n)\cdot\mathcal{R}v_h dx \quad \forall v_h\in H_h\,. \tag{4.10}$$

• , , 2.

$$\int_\Omega \frac{\Psi_h^{(2)}-\Psi_h^n}{\triangle t/2}v_h dx+\frac{1}{De}\int_\Omega \mathcal{R}\Psi_h^{(2)}\cdot\mathcal{R}v_h dx = -\frac{1}{De}\int_\Omega \Psi_h^{(1)}\mathcal{R}V(\Psi_h^{(1)})\cdot\mathcal{R}v_h dx \quad \forall v_h\in H_h\,. \tag{4.11}$$

• , , 3.

$$\int_\Omega \frac{\Psi_h^{n+1}-\Psi_h^n}{\triangle t}v_h dx+\frac{1}{De}\int_\Omega \mathcal{R}\Psi_h^{n+1}\cdot\mathcal{R}v_h dx = -\frac{1}{De}\int_\Omega \Psi_h^{(2)}\mathcal{R}V(\Psi_h^{(2)})\cdot\mathcal{R}v_h dx \quad \forall v_h\in H_h\,. \tag{4.12}$$

Note that in each step above the diffusion term is treated implicitly while the nonlinear term is treated explicitly. With this treatment, a standard stability criterion of CFL type applies; and in our computations a uniform time step is used. By doing this, it is well known that the moving mesh method does not gain much in time-stepping, since the time steps are controlled by the smallest size of the finite elements. One possible way of further speed-up in temporal discretization is to use the local time-stepping technique; see, e.g., [28].

For a piecewise linear approximation ϕ_h to a function ϕ, the following formula is adopted to approximate the computational error:

$$|\phi-\phi_h|_{1,\Omega}\sim\sqrt{\sum_{l\,:\,\text{interior edge}}\int_l\left[\mathcal{R}\phi_h\cdot\vec{t}_l\right]^2 dl}, \tag{4.13}$$

where $\vec{t}_l$ is the unit tangential vector of the edge l and $[\cdot]$ is the jump along the edge l; see, e.g., [7, 19]. With this a posteriori interpolant error estimator, a monitor function is proposed as

$$m(\Psi_h)|_K=\frac{1}{|K|}\sum_{l\subset\partial K}|l|\int_l\left[\mathcal{R}\Psi_h\cdot\vec{t}_l\right]^2 dl \quad \forall K\in\mathcal{T}_h\,. \tag{4.14}$$

The instantaneous average molecular orientation is quantified by the order parameter tensor

$$\mathbf{S}=\langle\mathbf{xx}\rangle-\frac{1}{3}\mathbf{I}. \tag{4.15}$$

The scalar order parameter $S=\frac{3}{2}\lambda$ represents the degree of molecular alignment, where λ is the eigenvalue with the largest absolute value for $\mathbf{S}$. It lies in the range $0\sim 1$ with $S=0$ for an isotropic phase and $S=1$ for rods that are all aligned in the same direction. Figure 5 shows the evolution of the scalar order parameter S

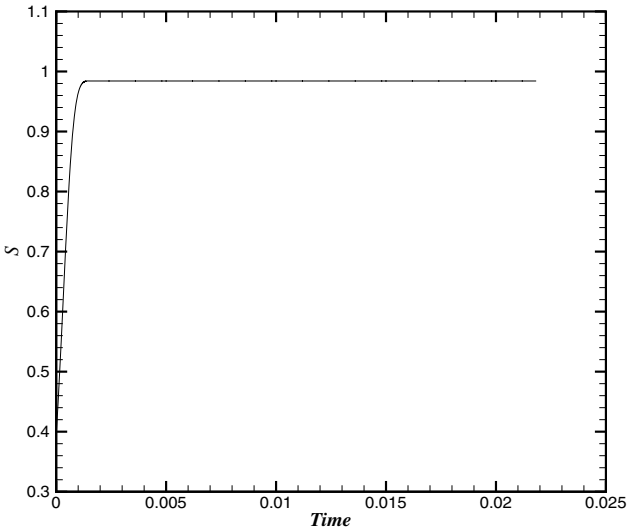


FIG. 5. *Example* 4.2. *Evolution of the scalar order parameter* S *with* $U = 1000$.

with $U = 100$. In this case, the stationary scalar order parameter S is very close to 1, indicating that all the molecular directions converge to one direction and the distribution function is very singular.

Figure 6 shows the meshes and the corresponding orientational distribution functions with $U = 100$ and 1000. To better demonstrate the singular behaviors of the solutions, the sphere is projected into a planisphere. The peak value of the distribution function with $U = 1000$ reaches $\Psi \sim 200$. Since the integral of the distribution function equals to 1, the area of nonzero Ψ is about $\frac{1}{200}$, where the solution has large gradients. Moreover, since the area of the unit sphere is equal to 4π, the compact support occupies only $\frac{1}{200}/(4\pi) \sim \frac{1}{2000}$ portion of the sphere. In order to put 25 $(= 5 \times 5)$ mesh points in such a small area, a uniform mesh requires a total of 50000 mesh points, while our moving mesh approach needs only 1600 points. By a close-up look of the mesh structure in Figure 7, it is seen that with the moving mesh approach more grid points are pushed into the (small) blow-up region.

More comparisons are made for Example 4.2 with $U = 1000$. In Figure 8, we compare the numerical solutions obtained by using the uniform mesh (nodes 12568, 25542, and 50266, respectively) and by using the adaptive moving mesh (1600 nodes). It is seen that the moving mesh solution with 1600 nodes is more accurate than the uniform mesh solution with 25542 nodes and is comparable to the uniform mesh solution with 50266 nodes. This is quite consistent with our estimate above.

Finally, it is necessary to compare the total CPU time usage for these computations. Table 1 gives the relevant CPU time usages with various numbers of nodes. The time step used is $\Delta t = 10^{-6}$ and the total time is $T = 2 \times 10^{-3}$. For this particular computation, the memory savings with the moving mesh method is about 1:30 and the speed-up is over 15.

5. Conclusions. In this paper, an effective moving mesh method is proposed to resolve steep solutions generated by PDEs on a sphere. The method exploits harmonic maps and invokes a regularization procedure to overcome the lack of convexity in the setup. Some special techniques for handling local coordinates on a sphere are also

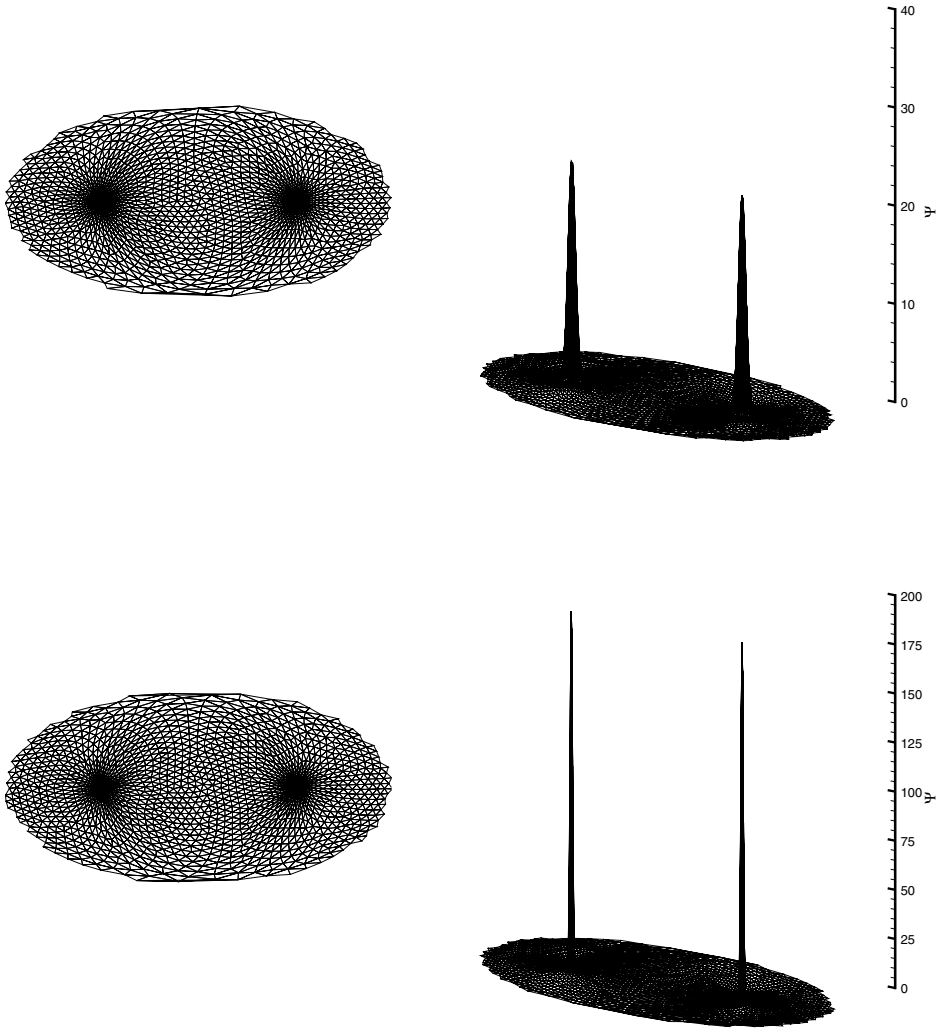


FIG. 6. *Example* 4.2. *The orientational distribution function obtained by using* 1600 *nodes and* 3196 *elements for* $U = 100$ *(top) and* $U = 1000$ *(bottom). The mesh on the sphere is projected to a disk (left) and the corresponding solution is shown on the right.*

designed. Efforts have been made to ensure that the mesh movement occurs only on the sphere and all the grid points remain on the sphere during the entire process of mesh redistribution. Two examples are given, one for a convection-dominated problem and the other for the Fokker–Planck equation on the sphere.

It is demonstrated that the moving mesh methods can not only enhance the solution , but can also substantially reduce the computational . The importance of increasing efficiency is obvious, but the latter part (of reducing storage) is also important for many practical problems, in particular higher-dimensional prob-

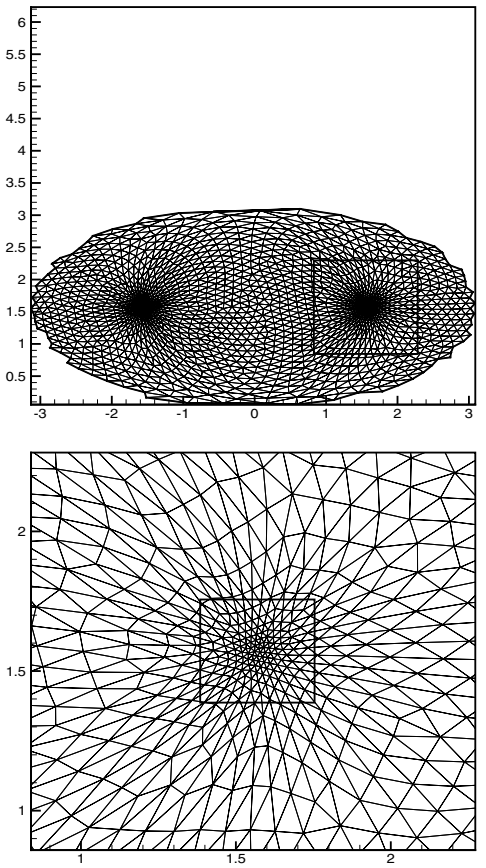


FIG. 7. *Example* 4.2. *Close-up of the adaptive mesh for* $U = 1000$ *(bottom figure of Figure* 6*).*

lems. For example, a large class of problems in complex fluid simulations are posed on spheres, for which Example 4.2 is a simple case. For the microscopic model of the polymer fluid, a well-known model is the rigid dumbbell model of Doi [9, 16]. In the Doi model, the orientation of a rod is determined by a pseudovector $\mathbf{u}$ on the unit sphere. The existing numerical schemes include a spectral-type method with spherical harmonic functions as basis functions (spherical harmonic expansion method); see, e.g., [14]; and a standard finite element approach; see, e.g., [20]. Since the underlying equations for the complex fluid models (such as the Fokker–Planck type) involve several dimensions in both physical and phase spaces, the required storage for the classical methods may be unrealistically large. It is seen that the moving mesh approach can use much less storage to obtain the required resolution. The numerical technique developed in this work is currently being used to study the rheological behavior of rod-like polymers governed by the Doi model; see, e.g., [8].

Finally, we point out that many important practical problems have been posed in spherical geometries, such as climate modeling and ocean modeling. One of our future projects is to extend the numerical techniques developed in this work to solve some practical problems defined on spheres with solution singularity.

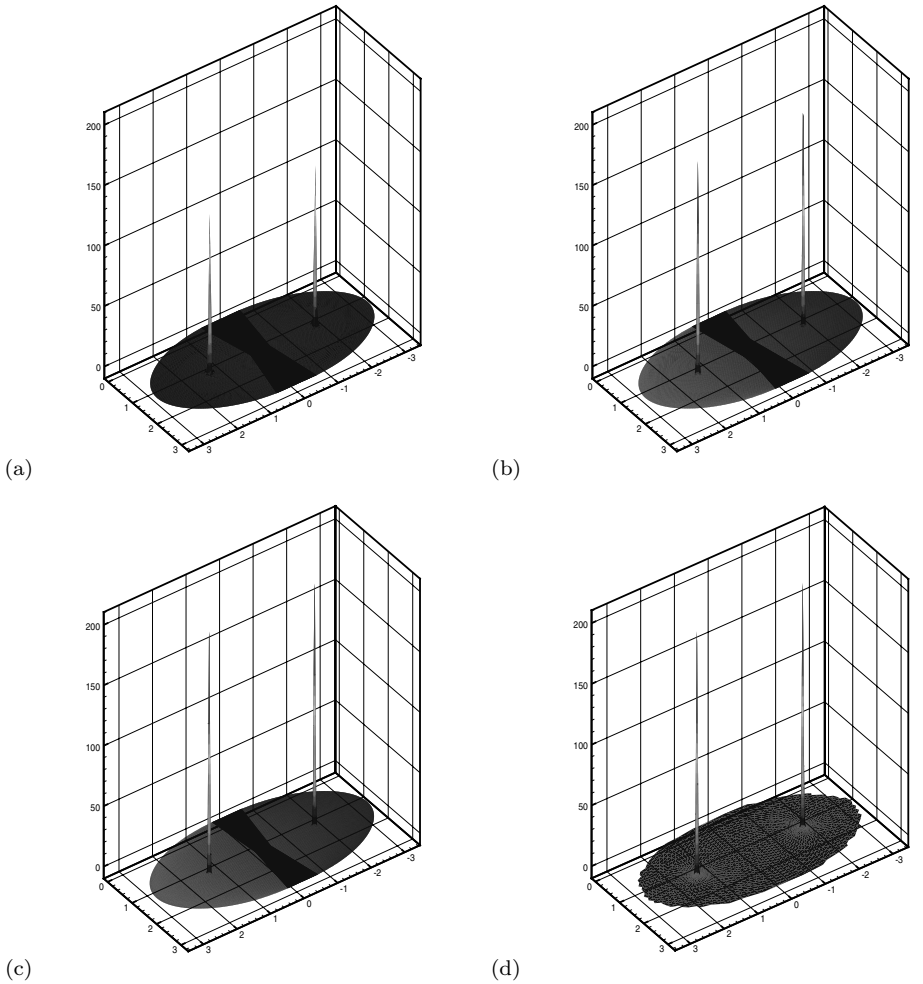


FIG. 8. *Example* 4.2. *The orientational distribution function obtained by using a uniform mesh with* (a) 12568 *nodes,* (b) 25542 *nodes,* (c) 50266 *nodes; and* (d) *by using a moving mesh* 1600 *nodes.* $U = 1000$.

TABLE 1
Example 4.2. *CPU time (seconds) used for uniform and moving mesh computations.*

	# of nodes	CPU time used
Uniform mesh	12568	11977
Uniform mesh	25542	24324
Uniform mesh	50266	48473
Moving mesh	1600	3044
Moving mesh	3200	5981

REFERENCES

[1] A. ANDERSON, X. ZHENG, AND V. CRISTINI, *Adaptive unstructured volume remeshing.* I. *The method*, J. Comput. Phys., 208 (2005), pp. 616–625.

[2] B. N. AZARENOK AND T. TANG, *Second-order Godunov-type scheme for reactive flow calculations on moving meshes*, J. Comput. Phys., 106 (2005), pp. 48–80.

[3] M. J. Baines, M. E. Hubbard, and P. K. Jimack, *A moving mesh finite element algorithm for the adaptive solution of time-dependent partial differential equations with moving boundaries*, Appl. Numer. Math., 54 (2005), pp. 450–469.
[4] G. Beckett, J. A. Mackenzie, and M. L. Robertson, *A moving mesh finite element method for the solution of two-dimensional Stephan problems*, J. Comput. Phys., 168 (2001), pp. 500–518.
[5] W. M. Cao, W. Z. Huang, and R. D. Russell, *An r-adaptive finite element method based upon moving mesh PDEs*, J. Comput. Phys., 149 (1999), pp. 221–244.
[6] W. M. Cao, W. Z. Huang, and R. D. Russell *A study of monitor functions for two-dimensional adaptive mesh generation*, SIAM J. Sci. Comput., 20 (1999), pp. 1978–1994.
[7] Y. Di, R. Li, T. Tang, and P. W. Zhang, *Moving mesh finite element methods for the incompressible Navier–Stokes equations*, SIAM J. Sci. Comput., 26 (2005), pp. 1036–1056.
[8] Y. Di and P. W. Zhang, *Moving mesh kinetic simulation for sheared rodlike polymers with high potential intensities*, Commun. Comput. Phys., 1 (2006), pp. 859–873.
[9] M. Doi, *Molecular dynamics and rheolgical properties of concentrated solutions of rodlike polymers in isotropic and liquid crystalline phases*, J. Polym. Sci., 19 (1981), pp. 229–243.
[10] M. Doi and S. F. Edwards, *The Theory of Polymer Dynamics*, Oxford University Press, New York, 1986.
[11] Q. Du and M. Gunzburger, *Grid generation and optimization based on centroidal Voronoi tessellations*, Appl. Math. Comput., 113 (2002), pp. 591–607.
[12] A. S. Dvinsky, *Adaptive grid generation from harmonic maps on Riemannian manifolds*, J. Comput. Phys., 95 (1991), pp. 450–476.
[13] J. Eell and J. H. Sampson, *Harmonic mapping of Riemannian manifolds*, Amer. J. Math., 86 (1964), pp. 109–160.
[14] M. G. Forest, Q. Wang, and R. Zhou, *The flow-phase diagram of Doi-Hess theory for sheared nematic polymers* II: *Finite shear rates*, Rheological Acta, 44 (2004), pp. 80–95.
[15] R. Hamilton, *Harmonic Maps of Manifolds with Boundary*, Lecture Notes in Math. 471, Springer-Verlag, New York, 1975.
[16] S. Z. Hess, *Fokker-Planck-equation approach to flow alignment in liquid crystals*, Z. Naturforsch. A, 31 (1976), pp. 1034–1037.
[17] W. Huang and W. Sun, *Variational mesh adaptation* II: *Error estimates and monitor functions*, J. Comput. Phys., 184 (2003), pp. 619–648.
[18] W. Z. Huang, *Mathematical principles of anisotropic mesh adaptation*, Commun. Comput. Phys., 1 (2006), pp. 276–310.
[19] R. Li, W.-B. Liu, H.-P. Ma, and T. Tang, *Adaptive finite element approximation for distributed elliptic optimal control problems*, SIAM J. Control Optim., 41 (2002), pp. 1321–1349.
[20] R. Li, C. Luo, and P.-W. Zhang, *Numerical simulation of Doi model of polymeric fluids*, in Advances in Scientific Computing and Applications, Y. Lu, W. Sun, and T. Tang, eds., Science Press, Beijing, 2004, pp. 258–273.
[21] R. Li, T. Tang, and P.-W. Zhang, *Moving mesh methods in multiple dimensions based on harmonic maps*, J. Comput. Phys., 170 (2001), pp. 562–588.
[22] R. Li, T. Tang, and P.-W. Zhang, *A moving mesh finite element algorithm for singular problems in two and three space dimensions*, J. Comput. Phys., 177 (2002), pp. 365–393.
[23] G. Liao, F. Liu, G. D. Pena, D. Peng, and S. Osher, *Level-set-based deformation methods for adaptive grids*, J. Comput. Phys., 159 (2000), pp. 103–122.
[24] K. Lipnikov and M. Shashkov, *The error-minimization-based strategy for moving mesh methods*, Commun. Comput. Phys., 1 (2006), pp. 53–80.
[25] R. S. Palais, *Foundations of Global Nonlinear Analysis*, Benjamin, New York, 1968.
[26] J. Sacks and K. Uhlenbeck, *The existence of minimal immersions of* 2*-spheres*, Ann. of Math. (2), 113 (1981), pp. 1–24.
[27] R. Schoen and S.-T. Yau, *On univalent harmonic maps between surfaces*, Invent. Math., 44 (1978), pp. 265–278.
[28] Z.-J. Tan, Z.-R. Zhang, Y.-Q. Huang, and T. Tang, *Moving mesh methods with locally varying time steps*, J. Comput. Phys., 200 (2004), pp. 347–367.
[29] H.-Z. Tang, *A moving mesh method for the Euler flow calculations using a directional monitor function*, Commun. Comput. Phys., 1 (2006), pp. 656–676.
[30] H. Z. Tang and T. Tang, *Adaptive mesh methods for one- and two-dimensional hyperbolic conservation laws*, SIAM J. Numer. Anal., 41 (2003), pp. 487–515.
[31] M. A. Walkley, P. H. Jimack, P. K. Kelmanson, and J. L. Summers, *Finite element simulation of three-dimensional free-surface flow problems*, J. Sci. Comput., 24 (2005), pp. 147–162.

[32] D. S. Wang and X. P. Wang, *A three-dimensional adaptive method based on the iterative grid redistribution*, J. Comput. Phys., 199 (2004), pp. 423–426.
[33] A. Winslow, *Numerical solution of the quasi-linear Poisson equation*, J. Comput. Phys., 1 (1967), pp. 149–172.
[34] P. A. Zegeling, *On resistive mhd models with adaptive moving meshes*, J. Sci. Comput., 24 (2005), pp. 263–284.

SIAM J. SCI. COMPUT.
Vol. 28, No. 4, pp. 1509–1532
© 2006 Society for Industrial and Applied Mathematics

EVALUATION OF THE BIOT–SAVART INTEGRAL FOR DEFORMABLE ELLIPTICAL GAUSSIAN VORTEX ELEMENTS*

LOUIS F. ROSSI†

Abstract. This paper introduces two techniques for approximating the Biot–Savart integral for deforming elliptical Gaussian functions. The primary motivation is to develop a high spatial accuracy vortex method. The first technique is a regular perturbation of the streamfunction in the small parameter $\epsilon = \frac{a-1}{a+1}$, where a^2 is the aspect ratio of the basis function. This perturbative technique is suitable for direct interactions. In the far field, this paper studies the applicability of the fast multipole method for deforming elliptical Gaussians since the multipole series are divergent. The noncompact basis functions introduce a new computational length scale that limits the efficiency of the multipole algorithm; however, by imposing a lower bound on the finest mesh size, one can approximate the far-field streamfunction to any specified tolerance.

Key words. computational fluid dynamics, vortex methods, deforming elements, elliptical Gaussians, fast summation

AMS subject classifications. 35Q30, 41A25, 41A30, 65D99, 65M12, 65M50, 65M60, 76D05

DOI. 10.1137/050637789

1. Introduction.

In this paper, we explore the calculation of Biot–Savart integrals for elliptical Gaussian basis functions,

$$\psi = -\frac{1}{4\pi} \iint_{-\infty}^{\infty} \log(|\vec{x} - \vec{s}|^2)\phi(\vec{s})d\vec{s}, \tag{1.1a}$$

$$\phi(\vec{s};\sigma,a) = \frac{1}{4\pi\sigma^2} \exp\left[\frac{-(x^2/a^2 + y^2a^2)}{4\sigma^2}\right], \tag{1.1b}$$

to determine the flow field and its derivatives induced by an elliptical Gaussian distribution of vorticity, or blob, in a two-dimensional incompressible fluid. The core size σ^2 and the aspect ratio a^2 are included as parameters. Translations and rotations of ϕ can be introduced through the symmetries in (1.1a), but we exclude them now for simplicity. There are two reasons for creating a robust and accurate approximation for the Biot–Savart integral of an elliptical Gaussian. The first reason, which is the impetus for this research, is the development of high spatial order vortex methods for two-dimensional, incompressible, viscous flows. For a vortex method, it is necessary to have a consistent description for the velocity field for it to be convergent. Recent results demonstrating that deforming elliptical Gaussian blobs can be used to achieve fourth order spatial accuracy for the convection and diffusion of passive scalars opened the door to high order viscous vortex methods [30, 31]. While not emphasized in this paper, the second reason for developing methods of approximating the Biot–Savart integral is that elliptical Gaussian vortices may be useful as modeling elements. That is, one could use a small number of elliptical Gaussians to model a more complex geophysical or laboratory system. To assess the interactions within the reduced system, one would need a means of evaluating the Biot–Savart integral of

*Received by the editors August 8, 2005; accepted for publication (in revised form) March 21, 2006; published electronically September 15, 2006. Portions of this work were supported by NSF grant DMS-9971800. Computations were supported by NSF SCREMS DMS-0322583.
http://www.siam.org/journals/sisc/28-4/63778.html
†Department of Mathematical Sciences, University of Delaware, Newark, DE 19716 (rossi@math.udel.edu).

each element. These reduced models yield helpful insights without resorting to specialized, large-scale simulations. For example, Knio, Collorec, and Juvé studied sound emissions from regular and chaotic systems of discrete vortices using three Gaussian vortices [18]. To understand the vortex core deformations, they refined their computation by using a large number of Gaussian elements to approximate the initial larger Gaussian. Deforming elliptical Gaussians offer more internal degrees of freedom for reduced models like these. Another use for exact solutions are as base states for stability studies. For instance, Sipp et al. recently studied the stability of a Gaussian vortex in an inhomogeneous fluid [37]. Similarly, a number of investigations have been conducted using interacting elliptical patches of constant vorticity (Kida vortices) to understand vortex mergers [26, 34, 45]. Elliptical Gaussian vortices in linear flow fields provides another family of base states for investigators to explore.

The primary purpose of this work is to the development of high accuracy viscous vortex methods. Vortex methods are a category of Lagrangian schemes long associated with successful computation of unsteady flow fields. For example, Rosenhead used eight-point vortices to compute the dynamics of an inviscid mixing layer in 1931 [27]. The common ingredient in all vortex methods is that the vorticity field of the flow is approximated by a linear combination of localized basis functions. These basis functions move with the local flow field, which must determined via a Biot–Savart integral over the basis functions. Overviews of a variety of vortex method techniques and applications can be found in [7, 20, 33] although new methods and techniques emerge regularly. Some more recent developments include large-scale engineering simulations of high Reynolds number flow past bluff bodies by Ploumhans et al. [25] and the use of radial basis functions for remeshing by Barba, Leonard, and Allen [4]. This paper focuses on deforming elliptical Gaussian basis functions because they provide a means of achieving high spatial accuracy for fully viscous flow. The use of deforming blobs for the two-dimensional, incompressible Navier–Stokes equations

$$\omega_t + \vec{u} \cdot \nabla\omega = \frac{1}{\mathrm{Re}} \nabla^2 \omega, \tag{1.2a}$$

$$\nabla \cdot \vec{u} = 0 \tag{1.2b}$$

(where $\vec{u}$ is the fluid velocity, $\omega = (\nabla \times \vec{u}) \cdot \widehat{z}$ is the fluid vorticity, and Re is the Reynolds number) is a natural extension of existing work both on core spreading techniques for Navier–Stokes and on the use of anisotropic elements to facilitate greater adaptivity and accuracy for Euler equations.

Core spreading has considerable appeal because it is deterministic, grid-free, and exact for uniform flow fields without operator splitting and the associated splitting errors. Furthermore, the uncorrected algorithm is easy to implement and computationally efficient. However, the spreading terms have an adverse effect on schemes using cores that satisfy the moment condition. For instance, Beale and Majda constructed a fourth order method for the Euler equation by superposing two Gaussian basis functions [5], but it drops to second order accuracy if one incorporates diffusion. To the knowledge of the author, core spreading was first proposed by Leonard [20], but concerns quickly arose over the consistency of this approach [13] because the simplest implementation of the scheme imposes a lower bound on the core size of the computational elements. The method was corrected in [28] with a consistent splitting algorithm and augmented with a merging algorithm [29]. Others have built upon these topics [16, 23, 35, 36]. Also, there have been engineering computations with core spreading methods (for example, [17, 41, 42, 43]), but the accuracy

of such methods is questionable in lieu of the finite core size issue as well as the lack of consistency in the core spreading formulation. Barba explores the numerical issues involved in [3]. While there is cause for concern about a lack of communication across disciplines, corrected core spreading is a strong and robust method for unsteady flows.

Still, core spreading is one of a small number of families of schemes featuring deforming blobs. There have been numerous efforts to use anisotropic elements for Euler and Navier–Stokes equations as a means of improving the efficiency of computations in large aspect ratio flows, a common situation in flows near solid boundaries. Teng used rigid and deforming elliptical patches to resolve boundary layers and vortex sheets more efficiently [38, 39, 40]. He establishes a theoretical $O(l^2 \log l)$ rate of convergence for his method where l is the core width of the computational elements. Marshall and Grant designed highly anisotropic elements to satisfy the no-slip, no normal flow boundary conditions [22]. Ojima and Kamemoto developed a scheme for large-scale simulations using deforming vortex elements that stretch with local flow deviations, but the resulting element is replaced with an isotropic element of equal volume at the end of each timestep to avoid having to calculate the Biot–Savart integral for anisotropic elements [24]. Rigorous convergence studies have not been reported for this method. Leonard used deforming elliptical Gaussian basis functions to simulate the convection of passive scalar quantities, (1.2a), but where $\vec{u}$ is specified, motivated by the fact that they remain self-similar under the linearized convection-diffusion equations [21]. Also, Moeleker and Leonard proposed using anisotropic elements with velocity field corrections based on subgrid scale for when computing Gaussian filtered scalar quantities, though they report modest gains in accuracy at a high computational cost [23].

This paper builds upon findings that fourth order spatial accuracy can be achieved with deforming elements when the basis functions advect with a velocity field given by the physical velocity field plus a velocity curvature correction [30]. These studies with an uncorrected method are limited to convection-diffusion equations. However, the analysis suggests that the same performance should be achieved for Navier–Stokes as long as the velocity field computed via the Biot–Savart integral and the exact field are close to one another because the errors are introduced through the nonlinear term at second order. In section 4, we shall see that this is the case. For convection-diffusion equations, the advantages of using more accurate but more computationally complex basis functions over less accurate simpler radial basis functions are realized at moderate problem sizes [31]. In short, deforming basis functions are better than axisymmetric basis functions in the sense that the former can compute a more accurate solution in the same amount of CPU time. This paper is a step in the same direction, building a better vortex method for Navier–Stokes. The vortex method described in this paper is an elliptical corrected core spreading vortex method, and for convenience it shall be termed ECCSVM. It is beyond the scope of this paper to explore corrections (the first "C"), but techniques such as remeshing, splitting, and merging are applicable to deformable blobs as well as axisymmetric blobs.

For a vortex method to be viable for scientific computation, it is not enough to evaluate the Biot–Savart integral directly to find the velocity field at each basis function location. If this were the case, the computational complexity of determining the velocity field at N blob positions would be $O(N^2)$. A number of fast summation algorithms have been developed over the past two decades that are capable of reducing certain particle computations from $O(N^2)$ to $O(N \log N)$ or even $O(N)$. These methods all hinge upon a separation of length scales. For instance, the Greengard–Rohklin algorithm uses Laurent expansions of complex Coulomb potentials to collect far-field

influences. The effects of distant elements are approximated as an aggregate, while velocity contributions from nearby elements are computed directly. The Greengard–Rohklin fast multipole method (FMM) has demonstrably reduced the computational complexity from $O(N^2)$ to $O(N)$ for potential kernels [14]. The multipole coefficients for the Laurent series can be collected on a hierarchy of meshes over the domain, so that the separation of near and far fields can be nested. There have been many extensions and augmentations to the FMM (for instance, [1, 2, 6, 15, 44]). The original FMM concepts have also been extended for fast polynomial interpolation and differentiation [11] and for a much broader class of velocity kernels [12]. An alternative to this algorithm based on Taylor series expansions has been analyzed and demonstrated by Draghicescu and Draghicescu [8, 9, 10]. This technique requires that one be able to Taylor expand the core shape function, and it achieves $O(N \log N)$ complexity. While the Draghicescu algorithm offers more flexibility in the types of core shapes one could use, the method requires that all elements have the same shape, so the Draghicescu method would not be suitable for the evolving anisotropic elements that are the focus of this paper. One of several difficulties in using the FMM under these circumstances is that the Biot–Savart integral (i.e., the velocity kernel) cannot be written in terms of elementary functions. The approach developed in this paper is to expand the Coulomb potential inside the integral (1.1a) as a Laurent series, and then perform the integration term by term. We shall see later that the resulting series is divergent but useful as an approximation and will yield an accurate $O(N)$ algorithm for finding the velocity field induced by a distant collection of elliptical Gaussian basis functions.

This paper has a simple organizational structure. This section reviewed different perspectives and contributions in the use of anisotropic elements for particle computations. Section 2 derives and demonstrates how one can approximate the streamfunction and its derivatives for an elliptical Gaussian basis function for direct pairwise interactions. In section 3, the fast multipole method is modified for use with anisotropic elliptical Gaussians. In that section, we rely heavily on the work of Greengard and Rohklin [14]. Finally in section 4, we certify these methods by approximating solutions to the Navier–Stokes for the Lamb–Oseen vortex and compare the numerical solutions with an exact solution.

2. Asymptotic evaluations for direct interactions in the near field. The evolution equations for the elliptical Gaussian basis functions (1.1b) depend upon $\vec{u}$ and $D\vec{u}$,[1] but unlike basis functions commonly used for vortex methods, there is no known elementary formula for these induced fields. A standard approach to problems of this type is to apply a quadrature method. Quadratures are accurate for this problem, but not effective relative to commonly used basis functions. The quadrature is computationally nontrivial for two reasons. First, there are at least two and as many as three relevant length scales (1, σa, σ/a) involved. (If a is close to unity, the last two collapse to one.) Second, numerical integration over two dimensions does not look promising in comparison with the single function evaluation required when the velocity kernel can be expressed in terms of elementary functions. In this section, we will see that the dimensionality can be reduced by one using Lamb's solution for a elliptical patch of constant vorticity. Rather than applying a quadrature to the resulting one-dimensional integral, we will use a regular perturbation so that the resulting one-dimensional integral can be evaluated analytically. This yields a series

[1] See [30] for the dynamics, and fourth order accuracy also requires knowing the second partial derivatives of the velocity components.

approximation written in terms of elementary functions, so that we are trading a two-dimensional quadrature for a few elementary function evaluations.

The induced velocity and velocity derivatives can be written as a regular perturbation in the small parameter

$$\epsilon = \frac{a-1}{a+1}. \tag{2.1}$$

For simplicity, we let $R^2 = x^2/a^2 + y^2a^2$, so that (1.1b) becomes

$$\phi(R;\sigma,a) = \frac{1}{4\pi\sigma^2} e^{-R^2/4\sigma^2}. \tag{2.2}$$

The variable R is a continuous index of level sets of the elliptical Gaussian element. To avoid an expensive two-dimensional quadrature of the Biot–Savart integral (1.1b), one can use the fact that the streamfunction (and therefore all derivatives) of an elliptical patch of vorticity with unit density and semimajor and minor axes of l_1 and l_2 can be determined using elliptical coordinates [19]:

$$\psi = \begin{cases} \frac{1}{2\pi(l_1+l_2)} \left(\frac{x^2}{l_1} + \frac{y^2}{l_2} \right), & (x,y) \in \mathcal{E}(l_1,l_2), \\ \frac{1}{2\pi} \left[\ln\left(\frac{\alpha+\beta}{l_1+l_2} \right) + \frac{\frac{x^2}{\alpha} + \frac{y^2}{\beta}}{\alpha+\beta} \right], & (x,y) \notin \mathcal{E}(l_1,l_2), \end{cases} \tag{2.3a}$$

$$\alpha = \sqrt{l_1^2 + \xi}, \tag{2.3b}$$

$$\beta = \sqrt{l_2^2 + \xi}, \tag{2.3c}$$

$$1 = \left(\frac{x^2}{l_1^2 + \xi} \right) + \left(\frac{y^2}{l_2^2 + \xi} \right), \tag{2.3d}$$

where $\mathcal{E}(l_1,l_2)$ is the support of the ellipse and

$$\vec{u} = \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} -\frac{\partial\psi}{\partial y} \\ \frac{\partial\psi}{\partial x} \end{bmatrix}.$$

The streamfunction induced by an elliptical blob can be expressed as an infinite sum of uniform elliptical patches. For each value of R in (2.2), the constituent patch would have density $\pi R^2(-\partial_R\phi)dR$ and axes $l_1 = Ra$ and $l_2 = R/a$. Adding all these patches together to form an elliptical Gaussian blob, one can calculate the value of the streamfunction at a point (x_*, y_*):

$$\psi(x_*,y_*) = -\int_0^{R_*} \psi_1 \partial_R\phi R^2 dR - \int_{R_*}^{\infty} \psi_2 \partial_R\phi dR, \tag{2.4a}$$

$$\psi_1 = \frac{1}{2} \left[\frac{\left(\frac{{x_*}^2}{\alpha} + \frac{{y_*}^2}{\beta} \right)}{\alpha+\beta} + \ln\left(\frac{\alpha+\beta}{Ra + R/a} \right) \right], \tag{2.4b}$$

$$\psi_2 = \frac{1}{2} \frac{\frac{{x_*}^2}{a} + {y_*}^2 a}{a + 1/a}, \tag{2.4c}$$

$$\alpha = \sqrt{R^2a^2 + \xi}, \tag{2.4d}$$

$$\beta = \sqrt{\frac{R^2}{a^2} + \xi}. \tag{2.4e}$$

Rather than using the implicit relation in (2.3d), we can use the quadratic solution

(2.5)
$$\xi = \frac{1}{2}\left\{ {\rho_*}^2 - R^2\left(a^2 + \frac{1}{a^2}\right) + \sqrt{\left[R^2\left(a^2 + \frac{1}{a^2}\right) - {\rho_*}^2\right]^2 + 4R^2({R_*}^2 - R^2)}\right\},$$

where ${\rho_*}^2 = {x_*}^2 + {y_*}^2$ and ${R_*}^2 = \frac{{x_*}^2}{a^2} + {y_*}^2 a^2$. Important limits to be used later are $\xi \to {\rho_*}^2$ as $R \to 0$ and $\xi \to 0$ as $R \to R_*$.

The second term on the right side of (2.4a) is elementary to integrate because ψ_2 does not vary with R. ψ_1 is not readily expressed in terms of the variable R, making the first term problematic. However, one can expand ψ_1 in powers of the small parameter ϵ. Expanding in ϵ rather than R has the advantage that, unlike R/σ, ϵ is a small number that is less than 1. The variable R is not guaranteed to be small relative to σ since one may wish to use the streamfunction anywhere in the domain. Thus, expanding in ϵ will yield a uniformly valid approximation of ψ_1 over the entire domain. The elliptical radius, R, is the natural variable of integration. At the same time, R_* is a parameter representing one possible combination of x_* and y_*. One possible complement is T_*, where ${T_*}^2 = {x_*}^2/a^2 - {y_*}^2 a^2$. Combined with the fact that the streamfunction must retain symmetries about the major and minor axes, the streamfunction can be expressed as a function of R_* and T_*. One could use radially symmetric coordinates such as ${x_*}^2 + {y_*}^2$ and ${x_*}^2 - {y_*}^2$, but it was found that these parameters yield a less accurate streamfunction approximation.

To perform an expansion of ψ_1, we make the substitutions

$$ {x_*}^2 = \frac{a^2}{2}\left({R_*}^2 + {T_*}^2\right), \tag{2.6a}$$

$$ {y_*}^2 = \frac{1}{2a^2}\left({R_*}^2 - {T_*}^2\right), \tag{2.6b}$$

$$ a = \frac{1+\epsilon}{1-\epsilon}, \tag{2.6c}$$

the latter arising from (2.1). Thus, it follows that

$$ {\rho_*}^2 = {R_*}^2 + 4\epsilon {T_*}^2 + 8\epsilon^2 {R_*}^2 + 12\epsilon^3 {T_*}^2 + 16\epsilon^4 {R_*}^4 + O(\epsilon^5). \tag{2.7}$$

Substituting this expression into the streamfunctions (2.4e) and (2.5), one can obtain an asymptotic expression for the streamfunction of the elliptical patch. If one expands these expressions out to a power of ϵ^n, then one obtains a polynomial in R of order $2n$. For example, if we expand in powers of ϵ for $n = 4$, we capture the following approximation for the streamfunction:

$$\begin{aligned}
\psi_1 = {} & \frac{1}{4}\left[1 + \ln\left(\frac{{R_*}^2}{R^2}\right)\right] + \epsilon\left(\frac{{T_*}^2}{{R_*}^2} - \frac{1}{2}\frac{{T_*}^2 R^2}{{R_*}^4}\right) \\
& + \epsilon^2\left[1 - 2\frac{{T_*}^4}{{R_*}^4} + \left(-2 + 4\frac{{T_*}^4}{{R_*}^4}\right)\frac{R^2}{{R_*}^2} + \left(1 - 2\frac{{T_*}^4}{{R_*}^4}\right)\frac{R^4}{{R_*}^4}\right] \\
& + \epsilon^3\left[\frac{16}{3}\frac{{T_*}^6}{{R_*}^6} - 5\frac{{T_*}^2}{{R_*}^2} + \left(\frac{37}{2}\frac{{T_*}^2}{{R_*}^2} - 24\frac{{T_*}^6}{{R_*}^6}\right)\frac{R^2}{{R_*}^2} + \left(-24\frac{{T_*}^2}{{R_*}^2} + 32\frac{{T_*}^6}{{R_*}^6}\right)\frac{R^4}{{R_*}^4}\right. \\
& \left. + \left(10\frac{{T_*}^2}{{R_*}^2} - \frac{40}{3}\frac{{T_*}^6}{{R_*}^6}\right)\frac{R^6}{{R_*}^6}\right]
\end{aligned}$$

$$
\begin{aligned}
&+\epsilon^4\left[-4+20\frac{{T_*}^4}{{R_*}^4}-16\frac{{T_*}^8}{{R_*}^8}+\left(20-136\frac{{T_*}^4}{{R_*}^4}+128\frac{{T_*}^8}{{R_*}^8}\right)\frac{R^2}{{R_*}^2}\right.\\
&+\left(-42+324\frac{{T_*}^4}{{R_*}^4}-320\frac{{T_*}^8}{{R_*}^8}\right)\frac{R^4}{{R_*}^4}+\left(40-320\frac{{T_*}^4}{{R_*}^4}+320\frac{{T_*}^8}{{R_*}^8}\right)\frac{R^6}{{R_*}^6}\\
&\left.+\left(-14+112\frac{{T_*}^4}{{R_*}^4}-112\frac{{T_*}^8}{{R_*}^8}\right)\frac{R^8}{{R_*}^8}\right]+O(\epsilon^5),
\end{aligned}
\tag{2.8}
$$

$$
\psi_2=\frac{1}{2}\left[\frac{1}{2}{R_*}^2+\left(\epsilon-\epsilon^3\right){T_*}^2\right]+O(\epsilon^5).
\tag{2.9}
$$

Using this expansion, the calculation of an approximate streamfunction together with its derivatives reduces to computing moments of elliptical Gaussians. Since ψ_2 does not vary with R, the second term of (2.4a) can be evaluated with the exact form of ψ_2 from (2.4e) or with the approximate form from (2.9). The advantage to using the approximation is that it makes the integrand and one derivative of the integrand continuous across R_*, so that one need not use Leibniz's rule when calculating velocities, but this is a minor consideration.

Next, we would like to calculate the derivatives of ψ. To be more precise, knowing the nth derivatives of ψ, we would like to be able to quickly compute the $(n+1)$st derivatives. While these operations can be performed using finite differences, the cost is high because ψ would have to be evaluated many times over a stencil near the origin. Instead, we shall determine the derivatives exactly from the asymptotic expression.

The regular perturbation in ϵ approximates ψ_1 as a polynomial in R whose coefficients depend upon ratios of powers T_* and R_*. Using the polynomial approximation, finding ψ in (2.4a) reduces to summing moments of the Gaussian. Differentiation of ψ reduces to a transformation of the polynomial coefficients in (2.8) followed by the summation of all the resulting moments. For instance, we can rewrite (2.8) as

$$
\begin{aligned}
\psi_1=&\frac{1}{4}\left[1+\ln\left(\frac{{R_*}^2}{R^2}\right)\right]\\
&+\left(\epsilon^2-4\epsilon^4\right)+\left(\epsilon-5\epsilon^3\right)\frac{{T_*}^2}{{R_*}^2}+\left(-2\epsilon^2+20\epsilon^4\right)\frac{{T_*}^4}{{R_*}^4}+\frac{16}{3}\epsilon^3\frac{{T_*}^6}{{R_*}^6}-16\epsilon^4\frac{{T_*}^8}{{R_*}^8}\\
&+\left[\left(-2\epsilon^2+20\epsilon^4\right)+\left(-\frac{1}{2}\epsilon+\frac{37}{2}\epsilon^3\right)\frac{{T_*}^2}{{R_*}^2}\right.\\
&\left.+\left(4\epsilon^2-136\epsilon^4\right)\frac{{T_*}^4}{{R_*}^4}-24\epsilon^3\frac{{T_*}^6}{{R_*}^6}+128\epsilon^4\frac{{T_*}^8}{{R_*}^8}\right]\frac{R^2}{{R_*}^2}\\
&+\left[\left(\epsilon^2-42\epsilon^4\right)-24\epsilon^3\frac{{T_*}^2}{{R_*}^2}+\left(-2\epsilon^2+324\epsilon^4\right)\frac{{T_*}^4}{{R_*}^4}+32\epsilon^3\frac{{T_*}^6}{{R_*}^6}-320\epsilon^4\frac{{T_*}^8}{{R_*}^8}\right]\frac{R^4}{{R_*}^4}\\
&+\left(40\epsilon^4+10\epsilon^3\frac{{T_*}^2}{{R_*}^2}-320\epsilon^4\frac{{T_*}^4}{{R_*}^4}-\frac{40}{3}\epsilon^3\frac{{T_*}^6}{{R_*}^6}+320\epsilon^4\frac{{T_*}^8}{{R_*}^8}\right)\frac{R^6}{{R_*}^6}\\
&+\left(-14\epsilon^4+112\epsilon^4\frac{{T_*}^4}{{R_*}^4}-112\epsilon^4\frac{{T_*}^8}{{R_*}^8}\right)\frac{R^8}{{R_*}^8}+O(\epsilon^5).
\end{aligned}
\tag{2.10}
$$

If we terminate the expansion of ψ_1 after finding the coefficients for ϵ^n, the result will be a polynomial of order $2n$ in R.

Differentiating the approximate streamfunction with respect to position x_* or y_* reduces to differentiating ratios of the form $\frac{{T_*}^{2m}}{{R_*}^{2n}}$, where $m \le n$. For instance, if

$m > 0$,

$$\frac{\partial}{\partial x_*}\left(\frac{{T_*}^{2m}}{{R_*}^{2n}}\right) = \left(m\frac{{T_*}^{2(m-1)}}{{R_*}^{2n}} - n\frac{{T_*}^{2m}}{{R_*}^{2(n+1)}}\right)\frac{2x_*}{a^2}, \tag{2.11a}$$

$$\frac{\partial}{\partial y_*}\left(\frac{{T_*}^{2m}}{{R_*}^{2n}}\right) = -\left(m\frac{{T_*}^{2(m-1)}}{{R_*}^{2n}} + n\frac{{T_*}^{2m}}{{R_*}^{2(n+1)}}\right)2y_*a^2. \tag{2.11b}$$

Similarly, when $m > 0$, we find

$$\begin{aligned}\frac{\partial^2}{\partial {x_*}^2}\left(\frac{{T_*}^{2m}}{{R_*}^{2n}}\right) &= \left(m\frac{{T_*}^{2(m-1)}}{{R_*}^{2n}} - n\frac{{T_*}^{2m}}{{R_*}^{2(n+1)}}\right)\frac{2}{a^2} \\ &\quad + \left[m(m-1)\frac{{T_*}^{2(m-2)}}{{R_*}^{2n}} - 2mn\frac{{T_*}^{2(m-1)}}{{R_*}^{2(n+1)}} + n(n+1)\frac{{T_*}^{2m}}{{R_*}^{2(n+2)}}\right]\frac{4{x_*}^2}{a^4}.\end{aligned} \tag{2.12}$$

However, if we make the substitution $4{x_*}^2/a^4 = 2({R_*}^2 + {T_*}^2)/a^2$ and perform similar substitutions on terms involving ${y_*}^2 = ({R_*}^2 - {T_*}^2)/(2a^2)$, we determine the following relations:

$$\begin{aligned}\frac{\partial^2}{\partial {x_*}^2}\left(\frac{{T_*}^{2m}}{{R_*}^{2n}}\right) &= \frac{2}{a^2}\left[m(m-1)\frac{{T_*}^{2(m-2)}}{{R_*}^{2(n+1)}} + m(m-2n)\frac{{T_*}^{2(m-1)}}{{R_*}^{2n}}\right. \\ &\quad \left. + n(n-2m)\frac{{T_*}^{2m}}{{R_*}^{2(n+1)}} + n(n+1)\frac{{T_*}^{2(m+1)}}{{R_*}^{2(n+2)}}\right],\end{aligned} \tag{2.13a}$$

$$\begin{aligned}\frac{\partial^2}{\partial {y_*}^2}\left(\frac{{T_*}^{2m}}{{R_*}^{2n}}\right) &= 2a^2\left[m(m-1)\frac{{T_*}^{2(m-2)}}{{R_*}^{2(n+1)}} + m(2n-m)\frac{{T_*}^{2(m-1)}}{{R_*}^{2n}}\right. \\ &\quad \left. + n(n-2m)\frac{{T_*}^{2m}}{{R_*}^{2(n+1)}} - n(n+1)\frac{{T_*}^{2(m+1)}}{{R_*}^{2(n+2)}}\right],\end{aligned} \tag{2.13b}$$

$$\frac{\partial^2}{\partial x_*\partial y_*}\left(\frac{{T_*}^{2m}}{{R_*}^{2n}}\right) = 4x_*y_*\left[m(1-m)\frac{{T_*}^{2(m-2)}}{{R_*}^{2n}} + n(n+1)\frac{{T_*}^{2m}}{{R_*}^{2(n+2)}}\right]. \tag{2.13c}$$

Relations (2.11) and (2.13) create an opportunity to calculate the partial derivatives of ψ. If we represent the streamfunction as a vector of coefficients with indices m and p because $n = m+p$ for terms of the form $\frac{{T_*}^m}{{R_*}^n}R^{2p}$ as shown in (2.10), the coefficients of the partial derivatives of ψ_1 are linear combinations of the coefficients for ψ_1. Specific algorithms arising from these expressions are provided in the appendix.

Now we can approximate the Biot–Savart integral by calculating moments of the elliptical Gaussian. For instance, if $n = 4$, we calculate the elementary integral

$$\begin{aligned}I &= -\int_0^{R_*} R^2(C_0 + C_1R^2 + C_2R^4 + C_3R^6 + C_4R^8)\partial_R\phi dR - C_5\phi(R_*) \\ &= \frac{\gamma}{2\pi}\Bigg\{2C_0 + 16C_1\sigma^2 + 192C_2\sigma^4 + 3072C_3\sigma^6 + 61440C_4\sigma^8 \\ &\quad - \left[C_0\left(\frac{{R_*}^2}{2\sigma^2} + 2\right) + C_1\left(\frac{{R_*}^4}{2\sigma^2} + 4{R_*}^2 + 16\sigma^2\right)\right.\end{aligned}$$

$$
\begin{aligned}
&+ C_2 \left(\frac{{R_*}^6}{2\sigma^2} + 6{R_*}^4 + 48{R_*}^2\sigma^2 + 192\sigma^4 \right) \\
&+ C_3 \left(\frac{{R_*}^8}{2\sigma^2} + 8{R_*}^6 + 96\sigma^2{R_*}^4 + 768\sigma^4{R_*}^2 + 3072\sigma^6 \right) \\
&+ C_4 \left(\frac{{R_*}^{10}}{2\sigma^2} + 10{R_*}^8 + 160\sigma^2{R_*}^6 + 1920\sigma^4{R_*}^4 + 15360\sigma^6{R_*}^2 + 61440\sigma^8 \right) \\
&- \frac{C_5}{2\sigma^2} \Bigg] \exp\left(-\frac{{R_*}^2}{4\sigma^2} \right) \Bigg\}, \qquad (2.14)
\end{aligned}
$$

where I could approximate ψ_x, ψ_y, ψ_{xx}, etc., and the C_i are polynomials in $\frac{{T_*}^m}{{R_*}^n}$.

To assess the accuracy of these approximations, the errors associated with the approximate solutions are compared to a reference solution using finite differences on a precise double-exponential quadrature of (2.4e) with fourteen digits of precision so that the third order finite differences could be adequately resolved. The results of a typical calculation is shown in Figure 1. The displayed relative error e is

$$e = \frac{\|f(x,y) - f_{\text{ref}}(x,y)\|_{\sup}}{\|f_{\text{ref}}\|_{\sup}}, \qquad (2.15)$$

where $f(x,y)$ is the approximation, $f_{\text{ref}}(x,y)$ is the reference solution, and the supremum is taken over the displayed domain.

As computational elements grow more eccentric and ϵ grows closer to one, the approximation becomes less accurate. On the other hand, the regular perturbation can be expanded for any n. Thus, there are competing trends. Figure 2 is an exploration of variations in ϵ and n, and we see the predicted rates of convergences are realized in practice.

3. Far-field calculations using fast multipoles. Using fast summation, far-field calculations of anisotropic elements can be performed more efficiently than direct calculations. Fast summation algorithms all hinge upon a separation of length scales, either artificial or physical. For instance, the Greengard–Rohklin algorithm uses Laurent expansions for complex potentials to collect far-field influences where the series is guaranteed to converge. At first glance, the Greengard–Rokhlin scheme would appear to be unsuitable for this method because the basis functions have infinite support and the kernel is singular, so the Laurent series will not converge at any point in the domain. We shall see in this section that the lack of compact support does not sink the idea completely because the partial sums contain useful information. By necessity, some details of Greengard–Rokhlin summation will be presented, but other aspects not related to anisotropic elements will be omitted in the interest of brevity. For a complete discussion of this method, the reader is referred to Greengard and Rokhlin's original article [14] or its successors, some of which are cited in section 1.

To begin, we will express positions in the domain (x,y) as a complex variable $z = x+iy$. Then the complex potential for a point vortex at the origin with circulation γ will be

$$\varphi(z) = -\frac{i\gamma}{2\pi} \ln z. \qquad (3.1)$$

If the velocity field has horizontal and vertical components u and v, respectively, then $\varphi' = u - iv$. Velocity derivatives in the far field can be computed by repeated

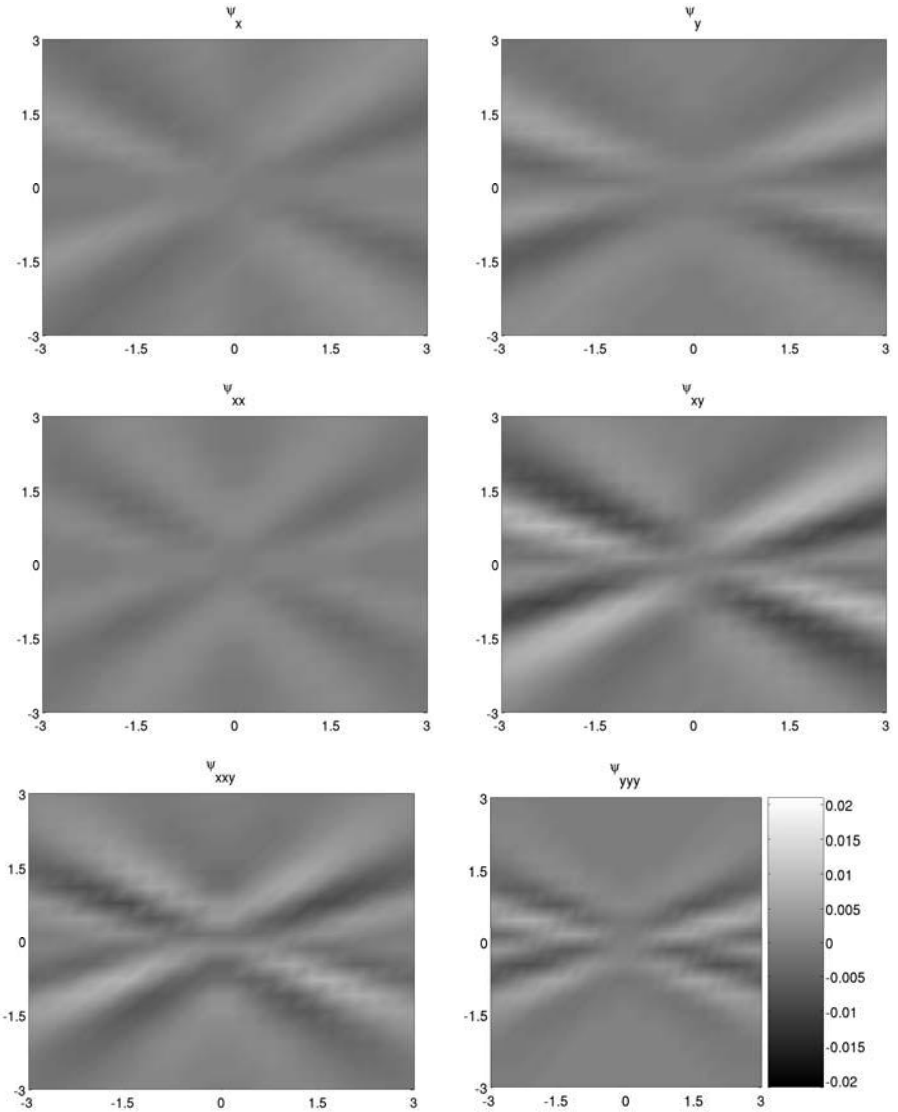


FIG. 1. *Relative field errors for the fourth order approximation of ψ for $\epsilon = 0.20$ (aspect ratio 2.25) and $\sigma = 1/4$. The relative error is calculated using (2.15) where the reference solution is calculated using a high precision quadrature. The scale as shown in the grayscale gradation bar on the ψ_{yyy} plot spanning an relative error of $\pm 2\%$ is used for all the plots.*

differentiation of φ. If the point vortex were at a point z_*, the complex potential could be evaluated relative to the origin using a power series expansion:

$$\begin{aligned}\varphi(z) &= -\frac{i\gamma}{2\pi}\ln(z - z_*) \\ &= -\frac{i\gamma}{2\pi}\left[\ln z + \ln\left(1 - \frac{z_*}{z}\right)\right] \\ &= -\frac{i\gamma}{2\pi}\left[\ln z - \sum_{j=1}^{\infty}\frac{1}{j}\left(\frac{z_*}{z}\right)^j\right].\end{aligned} \tag{3.2}$$

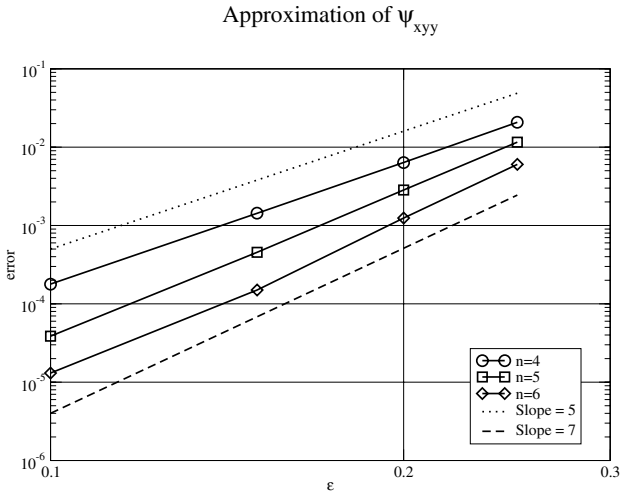


FIG. 2. *Behavior of asymptotic approximation for the streamfunction and its derivatives as one varies ϵ and n. The graph presents the absolute error versus ϵ on logarithmic axes along with reference slopes to show that the predicted convergence rates are realized.*

Thus, a singular potential displaced from the origin can be represented through a multipole expansion relative to the origin. The power series for $\ln(1-z)$ has a radius of convergence of 1, so that as long as the point of evaluation is further from the origin than the point vortex, the series will converge. In fact, one could design a simple algorithm where one uses direct velocity calculations if, say, $|z_*/z| > 1/2$ and use the series approximation otherwise. In the latter case, one could compute enough terms of the series to satisfy any desired error tolerance. The precise ratio $1/2$ is not so important, but it is crucial that one define the far field. For the remainder of this paper, we shall fix a parameter $\delta_1 < 1$ and define what is far from the origin as $|z_*/z| \leq \delta_1$. Contributions to the velocity field from elements that are not deemed far from the point of measurement will be computed directly as described in section 2, but for elements that are far from the point of measurement, we shall take advantage of the series solutions described in this section. When the point of measurement z is far from the point vortex z_*, series (3.2) converges.

One can [illegible] φ when $|z_*/z| \leq \delta_1$ by truncating the convergent power series,

$$\varphi(z) = -\frac{i\gamma}{2\pi}\left[\ln z - \sum_{j=1}^{p}\frac{1}{j}\left(\frac{z_*}{z}\right)^j - \frac{1}{p+1}\left(\frac{w}{z}\right)^{p+1}\right], \tag{3.3}$$

where w is a point on the line segment connecting 0 and z_* in the complex plane. If we consider p to be a controllable parameter in a calculation, we can choose p large enough to guarantee that the multipole approximation is as close as one would like to the exact potential.

Next, we can think about many elements with circulation γ_k and position z_k clustered near the origin. Following the same procedure as (3.2), we can find the

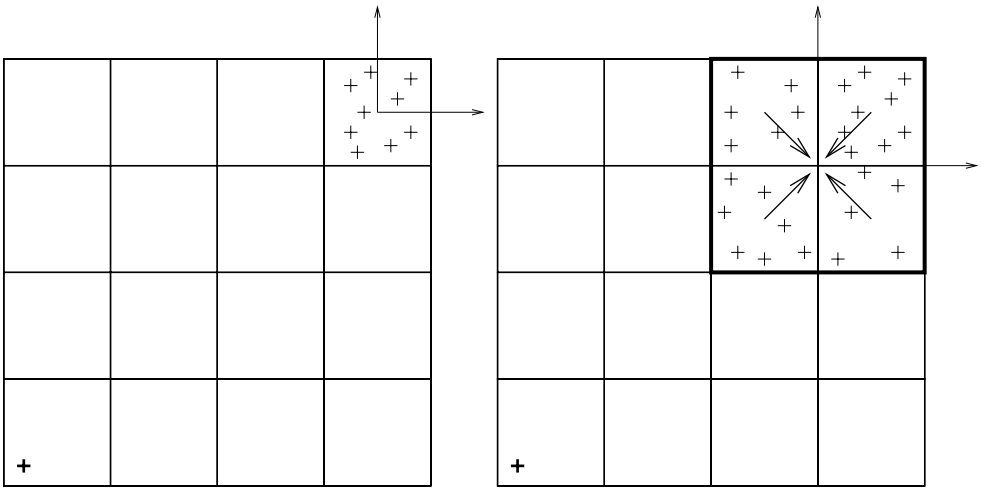


FIG. 3. *Overview of fast multipole summation. At left, we can see how a simple cluster can speed a calculation. If we seek to calculate the streamfunction and its derivatives for the computational element (shown in bold) at the bottom left box, we can treat distant elements in the upper right box as an aggregate rather than through individual interactions. Once one calculates the multipole coefficients for all the elements in the upper right box relative to the origin labeled with axes in the figure,* (3.4) *can be used anywhere in the far field, including the position of the basis function in the bottom left box. At right, we can see how one can save more computational effort by shifting multipole coefficients from the four small boxes in the upper right to the larger superbox by performing multipole expansion about the new origin labeled in the figure. The shifts are indicated by the bold arrows.*

complex potential at some point z relative to the origin:

$$\begin{aligned}\varphi &= -\sum_k \frac{i\gamma_k}{2\pi} \ln(z - z_k) \\ &= -\sum_k \frac{i\gamma_k}{2\pi} \left[\ln z - \sum_{j=1}^{p} \frac{1}{j} \left(\frac{z_k}{z}\right)^j - \frac{1}{p+1} \left(\frac{w_k}{z}\right)^{p+1} \right] \\ &= a_0 \ln z - \sum_{j=1}^{p} \frac{a_j}{j z^j} + R_{p+1}, \qquad \text{(3.4a)}\end{aligned}$$

where w_k is on the line segment connecting 0 and z_k in the complex plane and

$$a_0 = -\sum_k \frac{i\gamma_k}{2\pi}, \qquad a_j = -\sum_k \frac{i\gamma_k}{2\pi} z_k^j, \qquad |R_{p+1}| \le \frac{\delta_1^{p+1}}{2\pi(p+1)} \sum_k |\gamma_k|. \qquad \text{(3.4b)}$$

Thus, one can calculate the multipole coefficients a_j once for a cluster of elements grouped near the origin, and use (3.4) many times in the far field, as shown in Figure 3.

Substantial economy is added to the technique if one shifts multipole expansions relative to one another. That is, one can expand (3.2) or (3.4a) about another point (not the origin) and combine it with other clusters. In Figure 3, we see an example where four localized clusters can be combined into a single larger cluster when calculating distant interactions. The original Greengard–Rokhlin algorithm nests clusters in this arrangement to an arbitrarily fine level. The finest level can be defined as a

mesh where no box has more than k elements in it, so there are no more than Nk direct interactions.

Applying multipole summation to elliptical Gaussian basis functions poses a challenge because the basis functions have infinite support, so the Laurent series will not be convergent anywhere. However, even divergent series can provide useful information. In fact, one can use the leading partial sums to compute a useful approximation for the streamfunction and its derivatives.

We begin by calculating a multipole expansion for an elliptical Gaussian basis function that is displaced from the origin to some point (x_0, y_0), with circulation γ, aspect ratio a^2, width σ, and orientation θ. The complex potential would be the superposition

$$\varphi(z) = -\frac{i}{2\pi} \iint_{-\infty}^{\infty} \ln(z - z_*) \frac{\gamma}{4\pi\sigma^2} \exp\left[-\frac{\tilde{x}^2/a^2 + \tilde{y}^2 a^2}{4\sigma^2}\right] dx_* dy_*, \tag{3.5}$$

where $z = x + iy$, $z_* = x_* + iy_*$, and

$$\tilde{x} = \cos(\theta)(x_* - x_0) + \sin(\theta)(y_* - y_0), \quad \tilde{y} = -\sin(\theta)(x_* - x_0) + \cos(\theta)(y_* - y_0). \tag{3.6}$$

Writing the truncated multipole expansion inside the integral, interchanging the summation and integration, and changing variables, we obtain

$$\begin{aligned}
\varphi(z) &= -\frac{i\gamma}{2\pi} \iint_{-\infty}^{\infty} \left[\ln(z) - \sum_{j=1}^{p} \frac{1}{j} \left(\frac{z_*}{z}\right)^j - \frac{1}{p+1} \left(\frac{w_*}{z}\right)^{p+1}\right] \\
&\quad \times \frac{1}{4\pi\sigma^2} \exp\left[-\frac{\tilde{x}^2/a^2 + \tilde{y}^2 a^2}{4\sigma^2}\right] dx_* dy_* \\
&= -\frac{i\gamma}{2\pi} \left\{ \ln z - \sum_{j=1}^{p} \frac{1}{jz^j} \iint_{-\infty}^{\infty} (z_*)^j \frac{1}{4\pi\sigma^2} \exp\left[-\frac{\tilde{x}^2/a^2 + \tilde{y}^2 a^2}{4\sigma^2}\right] dx_* dy_* + R_{p+1} \right\} \\
&= -\frac{i\gamma}{2\pi} \left\{ \ln z - \sum_{j=1}^{p} \frac{1}{jz^j} \iint_{-\infty}^{\infty} \eta^j \frac{1}{4\pi\sigma^2} \exp\left[-\frac{\tilde{x}^2/a^2 + \tilde{y}^2 a^2}{4\sigma^2}\right] d\tilde{x} d\tilde{y} + R_{p+1} \right\},
\end{aligned} \tag{3.7}$$

where

$$\eta = (x_0 + iy_0) + (\cos\theta + i\sin\theta)\tilde{x} + (-\sin\theta + i\cos\theta)\tilde{y} \tag{3.8}$$

and

$$R_{p+1} = -\frac{1}{4\pi\sigma^2} \iint_{-\infty}^{\infty} \frac{1}{p+1} \left(\frac{w_*}{z}\right)^{p+1} \exp\left[-\frac{\tilde{x}^2/a^2 + \tilde{y}^2 a^2}{4\sigma^2}\right] dx_* dy_*. \tag{3.9}$$

The R_{p+1} determines the accuracy of the truncated series, so we will compute an upper bound on this quantity. Without loss of generality, we assume that $a > 1$ and that p is odd. Then, from (3.8) and (3.9), we can apply the triangle inequality and

change to polar coordinates to find that

$$
\begin{aligned}
|R_{p+1}| &\leq \frac{1}{p+1}\frac{1}{4\pi\sigma^2}\iint_{-\infty}^{\infty}\frac{|z_0|^{p+1}+|\tilde{z}|^{p+1}}{|z|^{p+1}}\exp\left[-\frac{\tilde{x}^2/a^2+\tilde{y}^2a^2}{4\sigma^2}\right]d\tilde{x}d\tilde{y} \\
&\leq \frac{1}{p+1}\left[\left(\frac{|z_0|}{|z|}\right)^{p+1}+\frac{2\pi}{4\pi\sigma^2}\int_0^{\infty}\frac{r^{p+2}}{|z|^{p+1}}\exp\left(-\frac{r^2}{4a^2\sigma^2}\right)dr\right] \\
&\leq \frac{1}{p+1}\left[\left(\frac{|z_0|}{|z|}\right)^{p+1}+\frac{2\pi}{4\pi\sigma^2}\int_0^{\infty}\frac{r^{p+2}}{|z|^{p+1}}\exp\left(-\frac{r^2}{4a^2\sigma^2}\right)dr\right]. \qquad (3.10)
\end{aligned}
$$

If we let

$$
C_{p+2} = \frac{2\pi}{4\pi\sigma^2}\int_0^{\infty}\frac{r^{p+2}}{|z|^{p+1}}\exp\left(-\frac{r^2}{4a^2\sigma^2}\right)dr, \qquad (3.11)
$$

then integrating by parts, we find that

$$
C_{p+2} = (p+1)\frac{2\sigma^2a^2}{|z|^2}C_p. \qquad (3.12)
$$

Since $C_1 = a^2$,

$$
C_{p+2} = a^2((p+1)/2)!\left(\frac{4\sigma^2a^2}{|z|^2}\right)^{(p+1)/2}. \qquad (3.13)
$$

Therefore,

$$
|R_{p+1}| \leq \frac{1}{p+1}\left[\left(\frac{|z_0|}{|z|}\right)^{p+1}+a^2\left(\frac{p+1}{2}\right)!\left(\frac{4\sigma^2a^2}{|z|^2}\right)^{(p+1)/2}\right]. \qquad (3.14)
$$

From this expression, we see two dominant mechanisms controlling the error of the multipole series. The first term inside the brackets is the traditional ratio discussed earlier requiring that $\frac{|z_0|}{|z|} < 1$. In other words, the singular potential has no natural length scale. In the second term, we see that that core size of the anisotropic element imposes a characteristic length scale $2\sigma a$ based on the computational resolution of the basis functions. If $\frac{|z|^2}{4\sigma^2a^2} < 1$, the upper bound on R_p in the multipole series grows without bound. However, if $\frac{|z|^2}{4\sigma^2a^2} > 1$, the upper bound on R_p decreases initially and then grows. For fast multipole summation with point vortices, one can fix δ_1 and compute a p that makes R_{p+1} smaller than any desired tolerance, and one can set the finest mesh level for direct interactions based on memory and computational performance criteria. In other words, with point vortices, the investigator can control the number of computational elements per box at the finest mesh level and then carry out the multipole expansion as far as needed to achieve a specified tolerance. However, for elements with finite cores, this is not the case because the number of terms in the multipole series that are guaranteed to be accurate depends upon the distance to the point of measurement $|z|$ relative to the numerical length scale $2\sigma a$. So, we must define a new parameter δ_2 that characterizes the far field as $\frac{|z|^2}{4\sigma^2a^2} > \delta_2$. One can estimate the number of useful terms by finding the value of p for which the ratio of consecutive upper bounds (3.14) is approximately one in the special case when $z_0 = 0$. This turnaround point is when

$$
\left(\frac{p-1}{2}\right)\left(\frac{4\sigma^2a^2}{|z|^2}\right) \approx 1,
$$

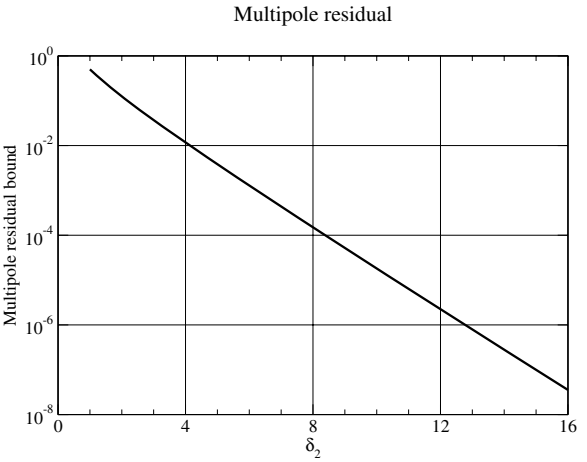


FIG. 4. *Exponential decay of multipole residual. This curve represents* (3.16) *for* $a^2 = 1$ *and* $\delta_1 = 0$. *Other values of* a^2 *will simply translate the curve vertically by* a^2. *The number of multipole terms required varies linearly with* δ_2 *as in* (3.15). *For instance, at* $\delta_2 = 8$ *we can expect* 17 *useful terms in the multipole series.*

$$p \approx \frac{|z|^2}{2\sigma^2 a^2} + 1. \tag{3.15}$$

This suggests a practical value for p would be $\bar{p} = 2\delta_2 + 1$. If we substitute (3.15) into (3.14) and apply the bounds δ_1 and δ_2 judiciously, we can obtain a useful estimate for the upper bound on $|R_{\bar{p}+1}|$ in (3.14),

$$|R_{\bar{p}+1}| \leq \frac{1}{2(\delta_2+1)} \left[\delta_1^{2(\delta_2+1)} + a^2 \Gamma(\delta_2+2)\, \delta_2^{-(\delta_2+1)} \right], \tag{3.16}$$

which is plotted in Figure 4. In practice, p is a discrete variable, but we have extended it as a continuum value and replaced the factorial with a gamma function. By inspection, the contributions from δ_2 dominate this expression.

While the convergence rate is exponential, accuracy requirements place a lower bound on the size of the finest mesh level via the parameter δ_2. When $\frac{|z|^2}{2\sigma^2 a^2}$ is too small, one must calculate the potential and its derivatives directly using the methods described in section 2.

Fixing $\bar{p}$ to achieve a desire accuracy, we calculate the multipole coefficients,

$$a_n = \frac{1}{4\pi\sigma^2} \iint_{-\infty}^{\infty} \eta^n \exp\left[-\frac{\tilde{x}^2/a^2 + \tilde{y}^2 a^2}{4\sigma^2} \right] d\tilde{x} d\tilde{y}, \tag{3.17}$$

where from (3.7),

$$\varphi(z) = -\frac{i\gamma}{2\pi} \left(\ln z - \sum_{j=1}^{p} \frac{a_j}{j z^j} + R \right). \tag{3.18}$$

Integrating directly, we see that $a_1 = x_0 + iy_0 = z_0$. If $n > 1$, we develop a recurrence

relation for the a_n's:

$$\begin{aligned}
a_n &= \frac{1}{4\pi\sigma^2}\iint_{-\infty}^{\infty}\eta^n \exp\left[-\frac{\tilde{x}^2/a^2+\tilde{y}^2a^2}{4\sigma^2}\right]d\tilde{x}d\tilde{y}\\
&= \frac{1}{4\pi\sigma^2}\iint_{-\infty}^{\infty}\left[x_0+iy_0+(\cos\theta+i\sin\theta)\tilde{x}+(-\sin\theta+i\cos\theta)\tilde{y}\right]\eta^{n-1}\\
&\quad\times\exp\left[-\frac{\tilde{x}^2/a^2+\tilde{y}^2a^2}{4\sigma^2}\right]d\tilde{x}d\tilde{y}\\
&= z_0a_{n-1}\\
&\quad+\frac{1}{4\pi\sigma^2}\iint_{-\infty}^{\infty}\eta^{n-1}\big[(\cos\theta+i\sin\theta)\tilde{x}\\
&\qquad\qquad+(-\sin\theta+i\cos\theta)\tilde{y}\big]\exp\left[-\frac{\tilde{x}^2/a^2+\tilde{y}^2a^2}{4\sigma^2}\right]d\tilde{x}d\tilde{y}.
\end{aligned}$$

Integrating by parts once, we obtain the relationship

$$\begin{aligned}
a_n &= z_0a_{n-1}\\
&\quad+\frac{1}{4\pi\sigma^2}(n-1)2\sigma^2\left[a^2(\cos\theta+i\sin\theta)^2+(-\sin\theta+i\cos\theta)^2/a^2\right]\\
&\quad\times\iint_{-\infty}^{\infty}\eta^{n-2}\exp\left[-\frac{\tilde{x}^2/a^2+\tilde{y}^2a^2}{4\sigma^2}\right]d\tilde{x}d\tilde{y}\\
&= z_0a_{n-1}+2(n-1)\sigma^2\left[a^2(\cos\theta+i\sin\theta)^2+(-\sin\theta+i\cos\theta)^2/a^2\right]a_{n-2}\\
&= z_0a_{n-1}+2(n-1)\sigma^2e^{2i\theta}\left(a^2-a^{-2}\right)a_{n-2}. \qquad (3.19)
\end{aligned}$$

If one is differentiating to find the velocity field, one finds that $a_0 = -1$ from the logarithmic term. With (3.19), one can quickly generate the multipole coefficients for an elliptical Gaussian.

To summarize, the essential difference between the classic FMM and fast summation of anisotropic Gaussians is that the latter requires that one impose a numerical length scale on the finest mesh level to achieve a specified accuracy via the partial summation of a divergent series. The accuracy limitations discussed in section 2 apply only to direct interactions amongst elements at the finest mesh level, but the far-field calculation enjoys exponential accuracy. After finding the finest mesh level, one applies the classic FMM using (3.19) as coefficients. The computational expense of determining or manipulating the multipole coefficients for an elliptical Gaussian is no different than it would be for any other basis function.

4. Sample calculations. To verify that the method actually achieves the predicted fourth order convergence rate, we apply the full vortex method to a standard problem with a known analytic solution. The initial vorticity distribution is

$$\omega(\vec{x},0) = 4\exp\left(4|\vec{x}|^2\right). \qquad (4.1)$$

The exact solution is

$$\omega(\vec{x},t) = \frac{1}{\frac{1}{4}+4t/\mathrm{Re}}\exp\left(\frac{|\vec{x}|^2}{\frac{1}{4}+4t/\mathrm{Re}}\right), \qquad (4.2)$$

where Re is the Reynolds number of the flow.

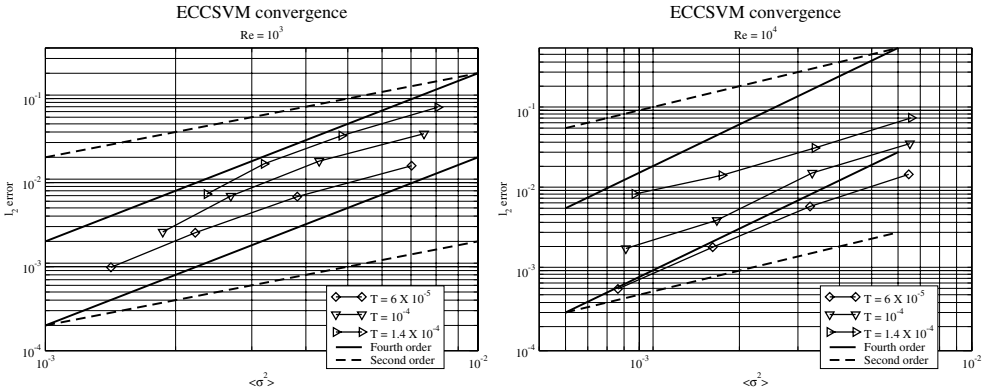


FIG. 5. *Convergence of ECCSVM at different Reynolds numbers with order* 4 *streamfunctions. As time grows, computational elements become more elongated and the Biot–Savart approximation becomes less accurate. Eventually, the velocity corrections break down, and the method reverts to second order spatial accuracy. The effective core* $\langle\sigma^2\rangle$ *is a circulation-weighted average over all elements.*

To generate the initial conditions, computational elements are laid down in a regular grid within a disk of radius 5/4 to approximate (4.1). This initial distribution can be constructed using initially isotropic or anisotropic basis functions using exact deregularization as explained in [32]. In these examples, the calculations always begin with isotropic basis functions. For all sample calculations, N/σ_0^2 remains constant where σ_0 is the initial core width and N is the number of computational elements used in the calculation. In other words, the initial overlap between computational elements is constant, $\frac{\sigma}{h} = 2$, where h is the initial grid width, as we refine the method. The initial core sizes were selected to be $\sigma_0^2 = 6.4 \times 10^{-3}$, 3.2×10^{-3}, 1.6×10^{-3}, and 8.0×10^{-4}. The vortex method uses the dynamic equations provided in [30] including curvature corrections to describe the movement and deformation of the elliptical Gaussian basis functions. The velocity field and its derivatives are computed using the methods described in this paper.

The results of these calculations at different Reynolds numbers are shown in Figure 5. For larger times in the simulation, the aspect ratios grow large in the domain, and the streamfunction approximation degrades to a point where the method falls back to second order spatial accuracy. The circulation weighted mean aspect ratio at $T = 1.4\times10^{-4}$ is approximately 2.8 (varying slightly from 2.7 to 2.9 depending upon σ_0^2 and the order of the streamfunction) with some aspect ratios in excess of 4.0 so that computations involve highly anisotropic elements in later phases of each calculation. The impact of using higher accuracy streamfunction calculations is shown in Figure 6. In this case, high order spatial accuracy is maintained throughout the interval shown even when basis functions are substantially elongated. Calculations were performed with and without the FMM discussed in this paper using $\delta_2 = 16$, but there is no noteworthy difference save that the former is substantially faster. For a single velocity evaluation on a single AMD Opteron CPU with a small problem size of $N = 1741$ ($\sigma = 6.4 \times 10^{-3}$), direct evaluation required 151 CPU seconds, while the multipole computation required only 55 CPU seconds. At a problem size of $N = 13{,}909$ ($\sigma_0^2 = 8.0 \times 10^{-4}$), direct evaluation required 9720 seconds, while the multipole computation required 496 CPU seconds.

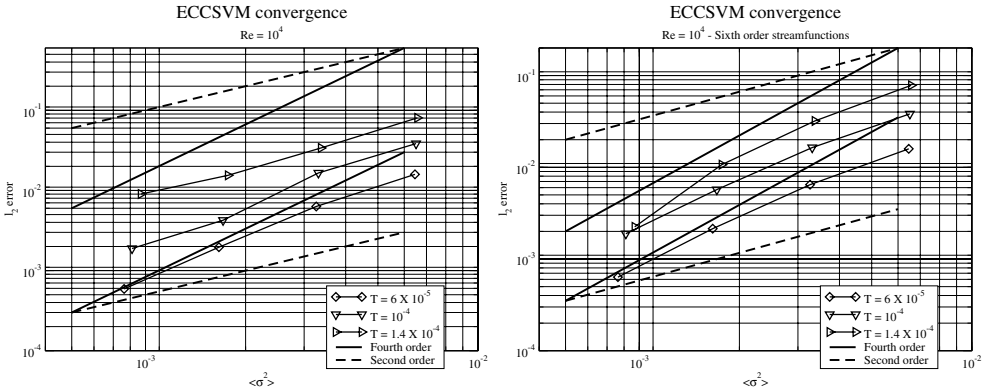


FIG. 6. *Convergence of ECCSVM with order* 4 *and* 6 *streamfunctions. The sixth order streamfunction permits more accurate velocity and velocity curvature computations than the forth order one as the aspect ratios grow. Thus, we see the anticipated fourth order spatial accuracy is sustained longer for the calculation using the sixth order calculation. The lower order behavior for small time is a result of interpolation error when setting the initial conditions for the computation.*

5. Conclusions. In this paper, we have explored a direct and far-field approximation to the Biot–Savart integral for elliptical Gaussian basis functions with the purpose of developing a high spatial accuracy vortex method. The direct method uses regular perturbations and yields a consistent means of producing uniformly valid estimates of the streamfunction and its derivatives for any order of accuracy. We have shown that Greengard and Rohklin's FMM algorithm is applicable to deforming Gaussians as long as care is taken when choosing the finest mesh size, but this paper also presents a means for calculating a fine mesh size that will meet specified accuracy requirements while reducing the operation count to $O(N)$. The result is a vortex method for the two-dimensional incompressible Navier–Stokes equations with fourth order spatial accuracy.

As discussed in the introduction, this paper has not addressed corrections, the first "C" in ECCSVM, since it is beyond the scope of the central problem. For large-scale computations, there is a need for remeshing or splitting/merging techniques. Furthermore, splitting, merging, and remeshing may serve the additional need to replace large aspect ratio elements with lower aspect ratio elements to retain accuracy in the Biot–Savart computation. Deformable elements introduce more internal degrees of freedom and so may offer some advantages inaccessible to rigid or axisymmetric elements. This remains fertile ground for future investigation.

Appendix. Algorithms for calculation of velocities and derivatives. We represent the approximation to the kernel of the streamfunction as a combination of terms:

$$\psi_1 = \frac{1}{4}\ln\left(\frac{{R_*}^2}{R^2}\right) + \sum_{p=0}^{N}\sum_{m=0}^{N} c_{m,p}^{\psi_1}\frac{{T_*}^{2m}}{{R_*}^{2(m+p)}}R^{2p}. \tag{A.1}$$

Knowing ϵ, all the c^{ψ_1}'s can be calculated.

To calculate the vector, $c^{\partial_{x_*}\psi_1}$, we represent the function as

$$\frac{\partial\psi_1}{\partial x_*} = \sum_{p=0}^{N}\sum_{m=0}^{N+1} c_{m,p}^{\partial_{x_*}\psi_1} x_* \frac{{T_*}^{2m}}{{R_*}^{2(m+p+1)}}R^{2p}, \tag{A.2}$$

and apply the following algorithm to determine $c^{\partial_{x_*}\psi_1}$ from the c^{ψ_1}.

Algorithm for $\frac{\partial}{\partial x_*}$.

1. Initialize the vector for $c^{\partial_{x_*}\psi_1}$ to zero.
2. $c_{0,0}^{\partial_{x_*}\psi_1} = \frac{1}{2a^2}$ from the logarithmic term in ψ_1.
3. $p \leftarrow 0$ to N.
4. $m \leftarrow 0$ to $N+1$.
5. If $m > 0$, then
$$c_{m,p}^{\partial_{x_*}\psi_1} = c_{m,p}^{\partial_{x_*}\psi_1} + \frac{2}{a^2} m c_{m,p}^{\psi_1}.$$
6.
$$c_{m,p}^{\partial_{x_*}\psi_1} = c_{m,p}^{\partial_{x_*}\psi_1} - \frac{2}{a^2}(m+p) c_{m,p}^{\psi_1}.$$
7. Next m.
8. Next p.

Both the first and second derivatives increase the number of $\frac{{T_*}^{2m}}{{R_*}^{2n}}$ terms in each power of R as seen in (2.11) and (2.13), so the upper limit on the m loop is $N+1$.

Similarly, to calculate the vector, $c^{\partial_{y_*}\psi_1}$, we represent the function as

$$\frac{\partial \psi_1}{\partial y} = \sum_{p=0}^{N} \sum_{m=0}^{N+1} c_{m,p}^{\partial_{x_*}\psi_1} y_* \frac{{T_*}^{2m}}{{R_*}^{2(m+p+1)}} R^{2p} \tag{A.3}$$

and apply the following algorithm to determine $c^{\partial_{y_*}\psi_1}$ from the c^{ψ_1}.

Algorithm for $\frac{\partial}{\partial y_*}$.

1. Initialize the vector for $c^{\partial_{y_*}\psi_1}$ to zero.
2. $c_{0,0}^{\partial_{y_*}\psi_1} = \frac{a^2}{2}$ from the logarithmic term in ψ_1.
3. $p \leftarrow 0$ to N.
4. $m \leftarrow 0$ to $N+1$.
5. If $m > 0$, then
$$c_{m,p}^{\partial_{y_*}\psi_1} = c_{m,p}^{\partial_{y_*}\psi_1} - 2a^2 m c_{m,p}^{\psi_1}.$$
6.
$$c_{m,p}^{\partial_{y_*}\psi_1} = c_{m,p}^{\partial_{y_*}\psi_1} - 2a^2(m+p) c_{m,p}^{\psi_1}.$$
7. Next m.
8. Next p.

Approximations for second derivatives can be computed directly from ψ_1 rather than using the first derivatives as a bootstrap.

We can calculate the vector $c^{\partial_{x_*x_*}\psi_1}$ for the representation

$$\frac{\partial^2\psi_1}{\partial {x_*}^2} = \sum_{p=0}^{N}\sum_{m=0}^{N+1} c_{m,p}^{\partial_{x_*x_*}\psi_1} \frac{{T_*}^{2m}}{{R_*}^{2(m+p+1)}} R^{2p} \tag{A.4}$$

using the following algorithm to determine $c^{\partial_{x_*x_*}\psi_1}$ from the c^{ψ_1}.

Algorithm for $\frac{\partial^2}{\partial {x_*}^2}$.

1. Initialize the vector for $c^{\partial_{x_*x_*}\psi_1}$ to zero.
2. $c_{1,0}^{\partial_{x_*x_*}\psi_1} = -\frac{1}{2a^2}$ from the logarithmic term in ψ_1. (This step would not be included for general polynomials in $\frac{{T_*}^{2m}}{{R_*}^{2n}}$.)
3. $p \leftarrow 0$ to N.
4. $m \leftarrow 0$ to $N+1$.
5. If $m > 0$,
$$c_{m-1,p}^{\partial_{x_*x_*}\psi_1} = c_{m-1,p}^{\partial_{x_*x_*}\psi_1} + \frac{2}{a^2}(m+p)(2p-m)c_{m,p}^{\psi_1}.$$
6. If $m > 1$,
$$c_{m-2,p}^{\partial_{x_*x_*}\psi_1} = c_{m-2,p}^{\partial_{x_*x_*}\psi_1} + \frac{2}{a^2}m(m-1)c_{m,p}^{\psi_1}.$$
7. $$c_{m,p}^{\partial_{x_*x_*}\psi_1} = c_{m,p}^{\partial_{x_*x_*}\psi_1} + \frac{2}{a^2}(p+m)(p-m)c_{m,p}^{\psi_1}.$$
8. If $m < N-1$,
$$c_{m+1,p}^{\partial_{x_*x_*}\psi_1} = c_{m+1,p}^{\partial_{x_*x_*}\psi_1} + \frac{2}{a^2}(p+m)c_{m,p}^{\psi_1}.$$
9. Next m.
10. Next p.

We can calculate the vector $c^{\partial_{y_*y_*}\psi_1}$ for the representation

$$\frac{\partial^2\psi_1}{\partial {y_*}^2} = \sum_{p=0}^{N}\sum_{m=0}^{N+1} c_{m,p}^{\partial_{y_*y_*}\psi_1} \frac{{T_*}^{2m}}{{R_*}^{2(m+p+1)}} R^{2p} \tag{A.5}$$

using the following algorithm to determine $c^{\partial_{y_*y_*}\psi_1}$ from the c^{ψ_1}.

Algorithm for $\frac{\partial^2}{\partial y_*^2}$.

1. Initialize the vector for $c^{\partial_{y_*y_*}\psi_1}$ to zero.
2. $c_{1,0}^{\partial_{y_*y_*}\psi_1} = \frac{1}{2}a^2$ from the logarithmic term in ψ_1. (This step would not be included for general polynomials in $\frac{{T_*}^{2m}}{{R_*}^{2n}}$.)
3. $p \leftarrow 0$ to N.
4. $m \leftarrow 0$ to $N+1$.
5. If $m > 0$,
$$c_{m-1,p}^{\partial_{y_*y_*}\psi_1} = c_{m-1,p}^{\partial_{y_*y_*}\psi_1} + 2a^2(m+p)(2p-m)c_{m,p}^{\psi_1}.$$
6. If $m > 1$,
$$c_{m-2,p}^{\partial_{y_*y_*}\psi_1} = c_{m-2,p}^{\partial_{y_*y_*}\psi_1} + 2a^2 m(m-1)c_{m,p}^{\psi_1}.$$
7. $$c_{m,p}^{\partial_{y_*y_*}\psi_1} = c_{m,p}^{\partial_{y_*y_*}\psi_1} + 2a^2(p+m)(p-m)c_{m,p}^{\psi_1}.$$
8. If $m < N-1$,
$$c_{m+1,p}^{\partial_{y_*y_*}\psi_1} = c_{m+1,p}^{\partial_{y_*y_*}\psi_1} - 2a^2(p+m)c_{m,p}^{\psi_1}.$$
9. Next m.
10. Next p.

We can calculate the vector $c^{\partial_{x_*y_*}\psi_1}$ for the representation

$$\frac{\partial^2\psi_1}{\partial x_*\partial y_*} = \sum_{p=0}^{N}\sum_{m=0}^{N+1} c_{m,p}^{\partial_{x_*y_*}\psi_1} x_* y_* \frac{{T_*}^{2m}}{{R_*}^{2(m+p+2)}} R^{2p} \tag{A.6}$$

using the following algorithm to determine $c^{\partial_{y_*y_*}\psi_1}$ from the c^{ψ_1}.

Algorithm for $\frac{\partial^2}{\partial x_*\partial y_*}$.

1. Initialize the vector for $c^{\partial_{y_*y_*}\psi_1}$ to zero.
2. $c_{0,0}^{\partial_{x_*y_*}\psi_1} = -1$ from the logarithmic term in ψ_1. (This step would not be included for general polynomials in $\frac{{T_*}^{2m}}{{R_*}^{2n}}$.)
3. $p \leftarrow 0$ to N.
4. $m \leftarrow 0$ to $N+1$.
5. If $m > 1$,
$$c_{m-2,p}^{\partial_{x_*y_*}\psi_1} = c_{m-2,p}^{\partial_{x_*y_*}\psi_1} - 4m(m-1)c_{m,p}^{\psi_1}.$$
6. $$c_{m,p}^{\partial_{x_*y_*}\psi_1} = c_{m,p}^{\partial_{x_*y_*}\psi_1} + 4(m+p)(m+p+1)c_{m,p}^{\psi_1}.$$
7. Next m.
8. Next p.

Higher order derivatives are calculated using first and second derivatives as building blocks and using these building blocks in the proper order. For instance, to calculate $\frac{\partial^3 \psi_1}{\partial {x_*}^3}$, we would begin with ψ_1 and calculate $\frac{\partial^2 \psi_1}{\partial {x_*}^2}$, which has the form given in (A.4). Then, we can apply the same $\frac{\partial}{\partial x_*}$ operation given before but without step 2 because there is no logarithmic term and the index on the $\frac{{T_*}^{2m}}{{R_*}^{2n}}$ terms is shifted by one. Therefore, we develop the representation

$$\frac{\partial \psi_1}{\partial {x_*}^3} = \sum_{p=0}^{N} \sum_{m=0}^{N+1} c_{m,p}^{\partial_{x_* x_* x_* x_*} \psi_1} x_* \frac{{T_*}^{2m}}{{R_*}^{2(m+p+2)}} R^{2p} \tag{A.7}$$

using the algorithm below.

Algorithm for $\frac{\partial}{\partial x_*}$.

1. Initialize the vector for $c^{\partial_{x_* x_* x_* x_*} \psi_1}$ to zero.
2. $p \leftarrow 0$ to N.
3. $m \leftarrow 0$ to $N+1$.
4. If $m > 0$, then
$$c_{m,p}^{\partial_{x_* x_* x_* x_*} \psi_1} = c_{m,p}^{\partial_{x_* x_* x_* x_*} \psi_1} + \frac{2}{a^2} m c_{m,p}^{\partial_{x_* x_* x_*} \psi_1}.$$
5.
$$c_{m,p}^{\partial_{x_* x_* x_* x_*} \psi_1} = c_{m,p}^{\partial_{x_* x_* x_* x_*} \psi_1} - \frac{2}{a^2} (m+p+1) c_{m,p}^{\partial_{x_* x_* x_*} \psi_1}.$$
6. Next m.
7. Next p.

REFERENCES

[1] C. Anderson, *A method of local corrections for computing the velocity field due to a distribution of vortex blobs*, J. Comput. Phys., 62 (1986), pp. 111–123.

[2] C. R. Anderson, *An implementation of the fast multipole method without multipoles*, SIAM J. Sci. Statist. Comput., 13 (1992), pp. 923–947.

[3] L. Barba, *Discussion: "Three-dimensional vortex method for gas-particle two-phase compound round jet," T. Uchiyama and A. Fukase, ASME J. Fluids Engrg.,* 127 (2005), *pp.* 32–40, J. Fluids Engrg., 128 (2006), pp. 643–645.

[4] L. A. Barba, A. Leonard, and C. B. Allen, *Advances in viscous vortex methods—meshless spatial adaption based on radial basis functions*, Internat. J. Numer. Methods Fluids, 47 (2005), pp. 387–421.

[5] J. T. Beale and A. Majda, *High order accurate vortex methods with explicit kernels*, J. Comput. Phys., 58 (1985), pp. 188–205.

[6] H. Cheng, L. Greengard, and V. Rokhlin, *A fast adaptive multipole algorithm in three dimensions*, J. Comput. Phys., 155 (1999), pp. 468–498.

[7] G.-H. Cottet and P. D. Koumoutsakos, *Vortex Methods: Theory and Practice*, Cambridge University Press, Cambridge, New York, 2000.

[8] C. I. Draghicescu, *An efficient implementation of particle methods for the incompressible Euler equations*, SIAM J. Numer. Anal., 31 (1994), pp. 1090–1108.

[9] C. I. Draghicescu, *A fast algorithm for vortex blob interactions*, J. Comput. Phys., 116 (1995), pp. 69–78.

[10] C. I. Draghicescu and M. Draghicescu, *A fast algorithm for vortex blob interactions*, J. Comput. Phys., 116 (1995), pp. 69–78.

[11] A. Dutt, M. Gu, and V. Rokhlin, *Fast algorithms for polynomial interpolation, integration, and differentiation*, SIAM J. Numer. Anal., 33 (1996), pp. 1689–1711.

[12] Z. Gimbutas and V. Rokhlin, *A generalized fast multipole method for nonoscillatory kernels*, SIAM J. Sci. Comput., 24 (2002), pp. 796–817.
[13] C. Greengard, *The core spreading vortex method approximates the wrong equation*, J. Comput. Phys., 61 (1985), pp. 345–348.
[14] L. Greengard and V. Rokhlin, *A fast algorithm for particle simulations*, J. Comput. Phys., 73 (1987), pp. 325–348.
[15] T. Hrycak and V. Rokhlin, *An improved fast multipole algorithm for potential fields*, SIAM J. Sci. Comput., 19 (1998), pp. 1804–1826.
[16] N. J. Huang, *Diffusion via splitting and remeshing via merging in vortex methods*, Internat. J. Numer. Methods Fluids, 48 (2005), pp. 521–539.
[17] T. Kida, T. Nakajima, and H. Suemitsu, *Second order core spreading vortex method in two-dimensional viscous flows*, JSME Internat. J. Ser. B Fluids Therm. Engrg., 41 (1998), pp. 441–446.
[18] O. M. Knio, L. Collorec, and D. Juvé, *Numerical study of sound emission by 2d regular and chaotic vortex configurations*, J. Comput. Phys., 116 (1995), pp. 226–246.
[19] H. Lamb, *Hydrodynamics*, Cambridge University Press, Cambridge, UK, 1993, Chaps. IV, VII, pp. 84–86, 232–233.
[20] A. Leonard, *Vortex methods for flow simulation*, J. Comput. Phys., 37 (1980), pp. 289–335.
[21] A. Leonard, *AIAA* 97-0204: *Large-eddy simulation of chaotic convection and beyond*, in Proceedings of the 35th Aerospace Sciences Meeting & Exhibit, American Institute of Aeronautics and Astronautics, 1997, pp. 1–8.
[22] J. S. Marshall and J. R. Grant, *A method for determining the velocity induced by highly anisotropic vorticity blobs*, J. Comput. Phys., 126 (1996), pp. 286–298.
[23] P. Moeleker and A. Leonard, *Lagrangian methods for the tensor-diffusivity subgrid model*, J. Comput. Phys., 167 (2001), pp. 1–21.
[24] A. Ojima and K. Kamemoto, *Numerical simulation of unsteady flow around three dimensional bluff bodies by an advanced vortex method*, JSME Internat. J. Ser. B Fluids Therm. Mixing, 43 (2000), pp. 127–135.
[25] P. Ploumhans, G. S. Winckelmans, J. K. Salmon, A. Leonard, and M. S. Warren, *Vortex methods for direct numerical simulation of bluff body flows: Application to the sphere at* $Re = 300, 500, 1000$, J. Comput. Phys., 178 (2002), pp. 427–463.
[26] G. Riccardi and R. Piva, *Motion of an elliptical vortex under rotating strain: Conditions for asymmetric merging*, Fluid Dynam. Res., 23 (1998), pp. 63–88.
[27] L. Rosenhead, *The point vortex approximation of a vortex sheet*, Proc. Roy. Soc. London Ser. A, 134 (1931), pp. 170–192.
[28] L. F. Rossi, *Resurrecting core spreading methods: A new scheme that is both deterministic and convergent*, SIAM J. Sci. Comput., 17 (1996), pp. 370–397.
[29] L. F. Rossi, *Merging computational elements in vortex simulations*, SIAM J. Sci. Comput., 18 (1997), pp. 1014–1027.
[30] L. F. Rossi, *Achieving high-order convergence rates with deforming basis functions*, SIAM J. Sci. Comput., 26 (2005), pp. 885–906.
[31] L. F. Rossi, *A comparative study of Lagrangian methods using axisymmetric and deforming blobs*, SIAM J. Sci. Comput., 27 (2006), pp. 1168–1180.
[32] L. F. Rossi, J. F. Lingevitch, and A. J. Bernoff, *Quasi-steady monopole and tripole attractors for relaxing vortices*, Phys. Fluids, 9 (1997), pp. 2329–2339.
[33] J. A. Sethian, *A brief overview of vortex methods*, in Vortex Methods and Vortex Motion, Frontiers Appl. Math., K. Gustafson and J. A. Sethian, eds., SIAM, Philadelphia, 1991, pp. 1–32.
[34] B. N. Shashikanth and P. K. Newton, *Geometric phases for corotating elliptical vortex patches*, J. Math. Phys., 41 (2000), pp. 8148–8162.
[35] D. Shiels, *Simulation of Controlled Bluff Body Flow with a Viscous Vortex Method*, Ph.D. thesis, California Institute of Technology, Pasadena, CA, 1998.
[36] D. Shiels and A. Leonard, *Investigation of a drag reduction on a circular cylinder in rotary oscillation*, J. Fluid Mech., 431 (2001), pp. 297–322.
[37] D. Sipp, D. Fabre, S. Michelin, and L. Jacquin, *Stability of a vortex with a heavy core*, J. Fluid Mech., 526 (2005), pp. 67–76.
[38] Z.-H. Teng, *Elliptic-vortex method for incompressible flow at high Reynolds number*, J. Comput. Phys., 46 (1982), pp. 54–68.
[39] Z.-H. Teng, *Variable-elliptical-vortex method for incompressible flow simulation*, J. Comput. Math., 4 (1986), pp. 255–262.
[40] Z.-H. Teng, L.-A. Ying, and P. Zhang, *Convergence of the variable-elliptic-vortex method for Euler equations*, SIAM J. Numer. Anal., 32 (1995), pp. 754–774.

[41] T. Uchiyama and M. Naruse, *Numerical simulation of gas-particle two-phase mixing layer by vortex method*, Powder Technology, 125 (2002), pp. 111–121.
[42] T. Uchiyama and M. Naruse, *Vortex simulation of slit nozzle gas-particle two-phase jet*, Powder Technology, 131 (2003), pp. 156–165.
[43] T. Uchiyama and T. Okita, *Numerical prediction of a plume diffusion field around a circular cylinder by the particle method*, Adv. Environmental Res., 7 (2003), pp. 573–581.
[44] L. van Dommelen and E. A. Rundensteiner, *Fast, adaptive summation of point forces in the two-dimensional Poisson equation*, J. Comput. Phys., 83 (1989), pp. 126–147.
[45] I. Yasuda and G. R. Flierl, *Two-dimensional asymmetric vortex merger: Merger dynamics and critical merger distance*, Dynam. Atmos. Oceans, 26 (1997), pp. 159–181.

SIAM J. SCI. COMPUT.
Vol. 28, No. 4, pp. 1533–1562
© 2006 Society for Industrial and Applied Mathematics

AN ALGORITHM FOR TWO-DIMENSIONAL MESH GENERATION BASED ON THE PINWHEEL TILING*

PRITAM GANGULY†, STEPHEN A. VAVASIS‡, AND KATERINA D. PAPOULIA§

Abstract. We propose a new two-dimensional meshing algorithm called PINW able to generate meshes that accurately approximate the distance between any two domain points by paths composed only of cell edges. This technique is based on an extension of pinwheel tilings proposed by Radin and Conway (see [C. Radin, *Ann. of Math.* (2), 139 (1994), pp. 661–702]). We prove that the algorithm produces triangles of bounded aspect ratio. This kind of mesh would be useful in cohesive interface finite element modeling when the crack propagation path is an outcome of a simulation process.

Key words. pinwheel tiling, mesh generation, isoperimetric, cohesive interface finite element

AMS subject classifications. 52C23, 68U05

DOI. 10.1137/040611343

1. Introduction.

One of the most widely used techniques to simulate fracture is cohesive interface finite element modeling. In this kind of model, the area or volume under consideration is subdivided into bulk elements, which are typically triangles or quadrilaterals in two dimensions and tetrahedra or hexahedra in three dimensions. Next, interfacial elements, which are edge elements in two dimensions or surface elements in three dimensions, are placed between some or all pairs of adjacent bulk elements. The cohesive model prescribes a relationship relating traction to relative displacement on the interfacial elements. There is an abundance of literature that deals with the nature of this relationship; e.g., see [5] and the references therein. A widely accepted modeling assumption is that the total energy needed to create the crack is proportional to its surface area (or length in two dimensions). In fact, the critical energy release rate G_c per unit surface area or length of crack is often a parameter of the cohesive model.

In a finite element model, the energy release rate is associated with surface area or length of interfacial elements composing the crack being modeled. If the discrepancy between the "true" crack path (i.e., the path the crack would follow if it were not for the finite element constraint that the crack path must lie on predetermined interfacial elements) and the path of the simulated crack is large for certain paths, then nonphysical preferred crack directions can exist. In other words, the results of the simulation would depend upon how well the boundaries of the mesh cells are aligned along the true crack path. In this paper, we propose a meshing technique that approximates the true path with the path along mesh boundaries with high accuracy even though the true path is unknown to the mesh generation algorithm. In particular, meshes generated by this technique have the property that for any crack path, the simulated

*Received by the editors July 9, 2004; accepted for publication (in revised form) February 13, 2006; published electronically September 15, 2006. This research was supported in part by NSF grants CMS-0239068 and CCF-0085969.

http://www.siam.org/journals/sisc/28-4/61134.html

†Department of Theoretical and Applied Mechanics, Thurston Hall, Cornell University, Ithaca, NY 14853 (pg45@cornell.edu).

‡Department of Computer Science, Upson Hall, Cornell University, Ithaca, NY 14853 (vavasis@cs.cornell.edu).

§School of Civil and Environmental Engineering, Hollister Hall, Cornell University, Ithaca, NY 14853 (kp58@cornell.edu).

and true crack path lengths converge to each other upon refining the mesh, which is a property not possessed by other simpler families of meshes. We call this algorithm the PINW mesh generator because it is based on an extension of the 1:2 pinwheel tiling described in the next section.

In section 3 we define "deviation ratio" and consider a simple experiment to test the properties of the 1:2 pinwheel mesh. The 1:2 pinwheel tiling seems to be too restricted to be useful for a general-purpose algorithm, so we explain how to generalize it to arbitrary triangles in section 4. This generalization is the basis for our meshing algorithm PINW. In section 5 we prove that our generalization still has the isoperimetric property. Then in section 6 we describe the algorithm. The main new ingredient introduced in that section is a procedure to convert a tiling to a mesh. The aspect ratio of the resulting mesh is analyzed in section 7.

The aspect ratio of the mesh is important for the cohesive fracture application because the bulk elements (e.g., triangles in two dimensions) are used to model a continuum mechanical theory such as linear elasticity. It is well known (see, e.g., Theorem 4.4.4 of [2], in which aspect ratio is called "chunkiness") that poorly shaped elements can lead to substantial errors in the elasticity solution.

2. Pinwheel tilings. In this section, we provide a brief introduction to the properties of pinwheel tilings. Tilings are a covering of Euclidean 2-space, $\mathcal{E}^2$, starting with a finite number of shapes called [illegible]. The tilings are constructed by translated and rotated copies of the prototiles that intersect each other only along the boundaries. The tilings were proposed to model crystallographic structures in the physics community.

The pinwheel tilings [6] are classified as [illegible]. In $\mathcal{E}^2$ this is equivalent to saying that no translation of the tiling leaves it invariant. The basic pinwheel tiling as developed by Radin and Conway (see [6]) has a hierarchical structure and is constructed by successive operations of subdivisions and expansions.

Consider a right triangle P with legs of length 1 and 2 referred to as the [illegible] and [illegible] sides. The hypotenuse is thus of length $\sqrt{5}$ and will be called the [illegible] side. The vertices will be named similarly; that is, the [illegible], and [illegible] vertices are opposite the corresponding sides. For brevity, we will call a right triangle with the ratio of its short to medium edge equal to 1/2 a "1:2 right triangle." This single tile is subdivided into five triangles that are all congruent to each other as shown in Figure 2.1. We call this subdivision into five triangles the "1:2 pentasection."

If one dilates the configuration in Figure 2.1 by a factor of $\sqrt{5}$ and then rotates and translates the resulting figure so that the dilated copy of C ends up coincident with the original tile P, then a larger subset of $\mathcal{E}^2$ is now tiled. Then 1:2 pentasection is applied to each of the five copies of P, and then another dilation followed by rotation and translation is carried out. Continuing this process infinitely would lead to a tiling of the plane. Thus, in the case of the standard pinwheel tilings, P and P_R (where P_R denotes the reflection of P about the x-axis) form the set of fixed prototiles, and the tiling uses translations and rotations of this set.

For our purposes, however, we will concentrate just on the repeated application of the 1:2 pentasection and omit the dilation, translation, and rotation steps, leading to the "subdivision pinwheel tiling" in which the cell diameter tends to zero and the area of the plane covered by the mesh does not expand from step to step. This is because we are interested in generating a mesh with varying amounts of refinement for a fixed region rather than a mesh that ultimately covers $\mathcal{E}^2$. In the subdivision pinwheel tiling, one starts with a fixed 1:2 triangle and then repeatedly subdivides

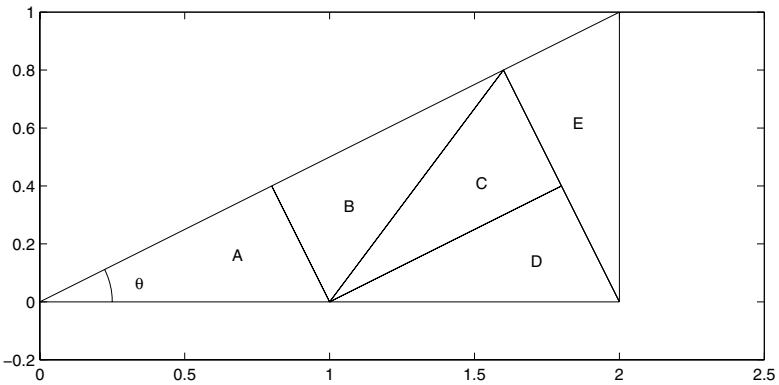


FIG. 2.1. *Basic pinwheel subdivision proposed by Radin.*

first the initial triangle and then each subtriangle into five congruent subtriangles using the 1:2 pentasection.

One can enumerate the rotation angles of the child triangles with respect to P and P_R as $R_\theta P_R$, $R_{\pi+\theta}P$, $R_\theta P$, $R_\theta P_R$, $R_{\frac{\pi}{2}+\theta}P$ corresponding to A, B, C, D, E in Figure 2.1, respectively. Here R_θ is rotation by θ in the counterclockwise direction. For a 1:2 right triangle, $\theta = \arctan(1/2)$, and θ/π in this case is irrational. The significance of this is as follows. As the number of subdivisions goes to infinity, so do the distinct orientations of the triangles. For example, suppose we keep track of the orientation of all triangles of type C with respect to the parent triangle in the subdivisions. As can be seen in Figure 2.1, the angle made by a triangle of type C with respect to the parent in the nth subdivision is $n\theta$. Since θ/π is irrational, $n\theta$ will represent a different angle for each n.

This presence of an infinite number of orientations leads to a special property known as the isoperimetric property [7]. For a tiling of $\mathcal{E}^2$, [illegible] means that given an $\epsilon > 0$, there exists a ρ such that for any two points p and q on the boundaries of the triangles with $\|p - q\| > \rho$, the shortest path from p to q that uses only tile edges has length at most $(1 + \epsilon)\|p - q\|$. Here $\|p - q\|$ denotes the Euclidean distance from p to q, which will also be denoted as $|pq|$, i.e., the length of edge pq.

There is an analogous property for the subdivision pinwheel tiling. In this case, let p, q be two points on the boundary of the initial triangle. Then for every $\epsilon > 0$, there exists an n such that after n recursive subdivisions of the initial triangle, the shortest path from p to q using only triangle edges is at most $\|p - q\|(1 + \epsilon)$. This theorem can be generalized so that p and q do not have to be on the boundary of the initial triangle but may be any two distinct boundary or interior points.

The isoperimetric property is the reason that pinwheel tilings are attractive for cohesive interface modeling. Consider a finite region $\Omega \subset \mathcal{E}^2$ tiled with an infinite sequence of pinwheel tilings $\mathcal{M}_1, \mathcal{M}_2, \ldots$ in which the triangles in $\mathcal{M}_i$ all have side lengths η_i, $2\eta_i$, $\sqrt{5}\eta_i$, and $\eta_i \to 0$ as $i \to \infty$. Then for an arbitrary straight segment of length λ connecting $p \in \Omega$ to $q \in \Omega$, and for an arbitrary $\epsilon > 0$, there exists an i such that in each of the tilings $\mathcal{M}_i, \mathcal{M}_{i+1}, \ldots$, there is a path from p to q using only mesh edges (except for initial and ending segments to connect p and q to the boundaries of the triangles that contain them) such that the length of the path is $\lambda(1 + \epsilon)$. We will give a proof of this result in a more general setting in section 5.

Since an arbitrary line segment can be well approximated by mesh edges, so can any piecewise smooth curve or network of such curves. The reason is that a network of piecewise smooth curves can be approximated arbitrarily accurately with a network of line segments. Then each line segment can be approximated arbitrarily accurately with a path from the pinwheel tiling.

Thus, when used for cohesive fracture, the pinwheel tiling has the property that all possible crack paths are approximated as accurately as desired (in terms of their length) by paths that use only mesh edges, as the mesh diameter tends to zero. As we shall see in the next section, more common mesh generation techniques do not have this property.

3. A computational experiment. In this section we carry out some simple experiments to quantify the isoperimetric property of the 1:2 tiling. Since our interest here is in meshes, we first explain how to convert the 1:2 pinwheel tiling to a mesh. It is apparent from Figure 2.1 that the pinwheel tiling is almost a triangulation except for the hanging node bisecting the medium side of triangle E. (Recall our terminology from the previous section that the short, medium, and long sides of a 1:2 right triangle are the edges of length 1, 2, and $\sqrt{5}$, respectively; the short vertex is opposite from the short side; the medium and long vertices are defined analogously.) We define a [illegible] of a planar subdivision into triangles to be a point that is a vertex of one triangle but lies on the strict relative interior of an edge of another triangle.

It is fairly simple to make the pinwheel tiling a mesh [6]: We divide every triangle into two by joining its medium vertex to the midpoint of its medium edge. In fact, it is not necessary to split all the triangles, and in our example we have obtained a mesh by splitting a certain subset of the tiles. This splitting is done only on the finest level of the pinwheel subdivision.

Our computational experiment is as follows. Starting from a 1:2 rectangle, we divide it into two 1:2 triangles and then apply the 1:2 pentasection n times to each of the 1:2 triangles. Thus, the final pinwheel tiling has $2 \cdot 5^n$ triangles. The resulting tiling of the original rectangle is then converted to a mesh using the technique in the previous paragraph.

Given a tiling $\mathcal{T}$ of a domain Ω, let $\mathrm{Skel}(\mathcal{T})$ be the 1-skeleton of $\mathcal{T}$, that is, the union of all edges of all triangles, and let $V(\mathcal{T})$ be the set of all vertices of $\mathcal{T}$. Let ρ be a positive parameter chosen small enough so that Ω contains a disk of diameter ρ. We propose to evaluate isoperimetric quality of the triangulation with the following quantity, which we refer to as the ρ [illegible]:

$$\operatorname*{dev}_{\rho}(\mathcal{T}) = \max\left\{ \frac{\mathrm{dist}_{\mathrm{Skel}(\mathcal{T})}(p,q)}{\|p-q\|_{\Omega}} : p,q \in V(\mathcal{T}) \text{ and } \|p-q\| \geq \rho \right\}.$$

Here, $\mathrm{dist}_{\mathrm{Skel}(\mathcal{T})}(\cdot,\cdot)$ means the shortest distance among paths restricted to $\mathrm{Skel}(\mathcal{T})$. The notation $\|p-q\|_{\Omega}$ means the geodesic distance from p to q, i.e., the shortest path among paths lying in Ω. Thus, this quantity measures the maximum ratio between the paths in the mesh versus geodesic paths. Clearly for any mesh $\mathcal{T}$ of any polygon, $\mathrm{dev}_{\rho}(\mathcal{T}) > 1$. The pinwheel mesh of the 1:2 rectangle has the property that for any $\rho \in (0,1)$, $\mathrm{dev}_{\rho}(\mathcal{PT}_m) \to 1$ as $m \to \infty$, where $\mathcal{PT}_m$ is the pinwheel tiling of the 1:2 rectangle after m levels of refinement.

Our experiment is to evaluate $\mathrm{dev}_1(\mathcal{PT}_m)$ for $\mathcal{PT}_1, \ldots, \mathcal{PT}_5$. The results are depicted in Table 3.1. The worst-case shortest path is shown in Figure 3.1.

In contrast, consider the meshes in Figure 3.2. The deviation ratios of these meshes have lower bounds greater than 1 irrespective of the number of subdivisions.

TABLE 3.1
Direct computation of deviation ratios for the first five levels of pinwheel subdivision.

n	$\mathrm{dev}_1(\mathcal{PT}_n)$
1	1.3416
2	1.1948
3	1.1843
4	1.1264
5	1.0831

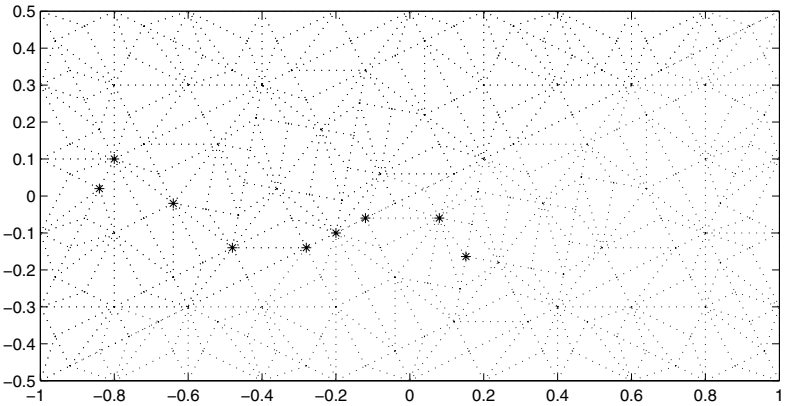


FIG. 3.1. *The path with the largest deviation ratio in $\mathcal{PT}_4$ is marked with asterisks.*

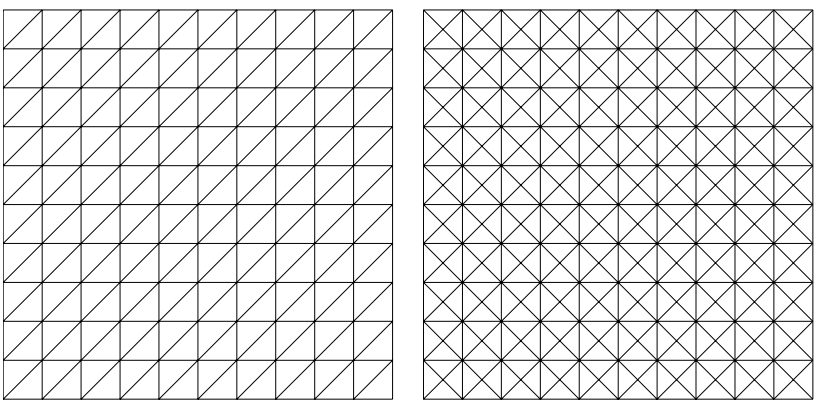


FIG. 3.2. *Two regular meshes of a square.*

In particular, the lower bound is $\sqrt{2} \approx 1.414$ for the mesh in Figure 3.2(a). For the mesh that was used by Xu and Needleman [10] (one of the first papers on cohesive finite element modeling), which is shown in Figure 3.2(b) and is sometimes called a "cross-triangle quadrilateral" mesh, the worst-case deviation ratio can be shown to be approximately equal to 1.082 in the limit as the mesh cell size tends to 0.

4. Generalization of pinwheel tilings. Pinwheel tiling based on 1:2 pentasection was extended to a tiling using an arbitrary right triangle and its reflection

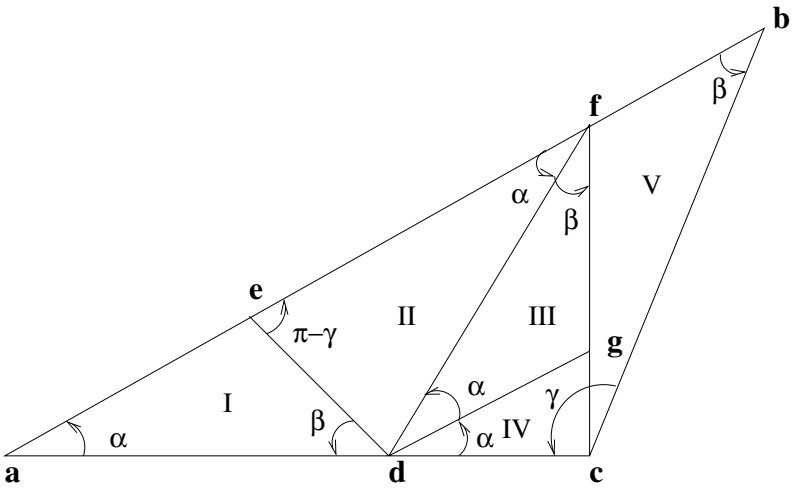


FIG. 4.1. *Our ATP subdivision of a triangle into five subtriangles, which generalizes* 1:2 *pinwheel subdivision.*

as prototiles by Sadun [8]. We now describe our approach to extend the pinwheel subdivision to arbitrary (nonright) triangles.

First, we propose a way of subdividing a general triangle and show that any number of subdivisions would produce triangles similar to a finite set of prototiles. Consider the triangle shown in Figure 4.1. We denote the vertices by a, b, and c and the included angles at these vertices by α, β, and γ, respectively, and assume that $\alpha < \gamma$. All angles are measured in radians unless otherwise noted. For the purpose of the ensuing description of our subdivision scheme, assume that abc are in clockwise order as in Figure 4.1. First, draw the segment cf such that f is a point on ab and $\angle fcb = \alpha$ measured counterclockwise from cb. From f draw fd such that d is on ac and $\angle dfc = \beta$ measured clockwise from fc. From d draw e and g such that e is on ab and g is on cf and $\angle ade = \beta$ clockwise from da and $\angle gdc = \alpha$ counterclockwise from dc. Thus, we have a subdivision of a general triangle $\triangle abc$ into five triangles of which I, III, and V are similar to the parent and the remaining two, II and IV, are similar to each other but not to the parent. Note that we required $\alpha < \gamma$ to make this construction. Let us call this procedure for subdividing a triangle into five subtriangles [illegible] subdivision or ATP subdivision for short. A different subdivision will result if the three nodes are reordered. Therefore, we assume that when ATP subdivision is applied, the nodes are specified in a certain sequence a, b, c. We assume also that the sequence is [illegible] for ATP, meaning that $\alpha < \gamma$.

THEOREM 4.1. [illegible] $\triangle abc$ [illegible] $\angle a, \angle b, \angle c$ [illegible] α, β, γ [illegible] $A_1 = \{\alpha, \beta, \gamma\}$ [illegible] $A_2 = \{\alpha, \gamma - \alpha, \pi - \gamma\}$ [illegible]. This is obvious by simply checking all the angles in Figure 4.1 and using the fact that angles of a triangle sum to π. Observe that triangles I, III, and V in Figure 4.1 are similar to $\triangle abc$ while triangles II and IV have angles from A_2. □

Let us assume that the vertices of triangles I, III, and V are sequenced according to their similarity with $\triangle abc$. Let us assume that triangles II and IV have angles sequenced $\alpha, \gamma - \alpha, \pi - \gamma$. Call these the [illegible] sequences of the vertices of the

subtriangles (inherited from $\triangle abc$). Note that the inherited sequence is feasible: $\alpha < \pi - \gamma$ since $\alpha + \beta + \gamma = \pi$.

THEOREM 4.2. [illegible] A_1 [illegible] A_2 [illegible]. One checks that if we define $\alpha' = \alpha$, $\beta' = \gamma - \alpha$, and $\gamma' = \pi - \gamma$, then $(\alpha', \gamma' - \alpha', \pi - \gamma') = (\alpha, \beta, \gamma)$. □

For the rest of this paper, we say that a triangle with angles α, β, γ is [illegible] to a triangle with angles $\alpha, \gamma - \alpha, \pi - \gamma$. The point of Theorem 4.2 is that conjugacy is a symmetric relationship. We remark that if the original triangle is a right triangle, i.e., $\gamma = \pi/2$, then this triangle is similar to its conjugate. This is the case considered by [8].

Starting from an arbitrary nonequilateral triangle $T_1 = \triangle abc$, let us assume that its nodes are sequenced a, b, c according to $\alpha \le \beta \le \gamma$. One can produce a tiling of T_1 by applying ATP to T_1 and then recursively applying ATP to subtriangles according to some scheme. The previous theorem implies that all the resulting subtriangles are congruent either to T_1 or its conjugate. We call the resulting subdivision of T_1 a [illegible], GPT for short, of T_1.

This procedure runs into a difficulty when $\gamma \approx \alpha$ (i.e., the initial triangle is close to equilateral) because in this case the conjugate triangle will have poor aspect ratio. We define the [illegible] of a triangle to be the square of the longest side length of the triangle divided by its area. An angle close to 0 automatically implies poor (large) aspect ratio as proved later in Lemma 7.8.

If an initial triangle is close to equilateral and therefore has $\gamma \approx \alpha$, we first subdivide it into three about its in-center; that is, we join the in-center to the vertices of the original triangle to form three subtriangles. We use a cutoff K_s in our algorithm: If $\gamma - \alpha < K_s$, then the preliminary tripartition is carried out. Parameter K_s is chosen to optimize the smallest angle. In other words, a parent is divided about the in-center if the smallest angle in $A_1 \cup A_2$ without preliminary tripartition is smaller than the smallest angle among the three sets $A_1 \cup A_2$ (one A_1 and one A_2 for each subtriangle) after tripartition. It can be shown that with the best choice of cutoff K_s, it is guaranteed that $\gamma - \alpha > 0.4$.

5. Isoperimetric property. This section is devoted to showing the result that the GPT introduced in the previous section obeys an isoperimetric inequality. The analysis and proof technique in this section closely follow the proof from [7]. The following is the key lemma in the proof of isoperimetry.

LEMMA 5.1. [illegible] $T = \triangle abc$ [illegible] T [illegible] α, β, γ [illegible] a, b, c, [illegible] $\alpha < \gamma$ [illegible] α/π [illegible] $\theta \in [0, \pi)$ [illegible] $\epsilon > 0$ [illegible] T [illegible] E [illegible] E [illegible] x [illegible] $(\theta - \epsilon, \theta + \epsilon)$ [illegible] θ [illegible] $\theta + \pi$ [illegible] x [illegible] E [illegible] $\zeta(\alpha, \beta, \gamma, \epsilon)\lambda$ [illegible] ϵ [illegible] $\zeta()$ [illegible] λ [illegible] T [illegible]. Observe that Triangle III in ATP subdivision is similar to the initial triangle T but is rotated by α. Call this triangle T'. If this triangle is subdivided by ATP, there will be another smaller copy of T, say T'', rotated by 2α, etc. The infinite sequence $\alpha, 2\alpha, 3\alpha, \ldots$ taken mod π is dense in the interval $[0, \pi)$ by the assumption that α/π is irrational. Therefore, for some sufficiently fine tiling, there is an edge E of triangle $T^{(k)}$ in the interval $(\theta - \epsilon, \theta + \epsilon)$.

For the second part of the lemma, observe that for any $\epsilon > 0$ there is an $n \equiv n(\epsilon, \alpha)$ such that every point in $[0, \pi]$ is distance (mod π) at most ϵ from at least one point in the set $\{\alpha, 2\alpha, 3\alpha, \ldots, n\alpha\}$. Therefore, one of $T, T', \ldots, T^{(n)}$ described in the last paragraph will have the desired edge E. The longest side length of T is λ; the longest side length of T' is $q(\alpha, \beta, \gamma)\lambda$, where q is some universal function (not depending on anything other than α, β, γ) derived from our construction. By similarity, the longest edge of $T^{(2)}$ has length $q(\alpha, \beta, \gamma)^2\lambda$. Thus, if we define $\zeta(\alpha, \beta, \gamma, \epsilon) \equiv \phi(\alpha, \beta, \gamma)q(\alpha, \beta, \gamma)^{n(\epsilon,\alpha)}$, where $\phi(\alpha, \beta, \gamma)$ is the ratio of the shortest to longest side length of T, then the length of E is at least $\zeta(\alpha, \beta, \gamma, \epsilon)\lambda$. This proves the second part of the lemma. □

For the first main theorem of this section, we need more definitions. We say that a GPT $\mathcal{T}'$ of a triangle T [illegible] another GPT $\mathcal{T}$ of T provided that for each tile U of $\mathcal{T}$, either U appears in $\mathcal{T}'$ or a subdivision of U appears in $\mathcal{T}'$. This definition implies that $V(\mathcal{T}) \subset V(\mathcal{T}')$ and $\mathrm{Skel}(\mathcal{T}) \subset \mathrm{Skel}(\mathcal{T}')$. If T is a triangle, we use the words [illegible] and [illegible] of T synonymously. The [illegible] of T is the length of its longest side, and the [illegible] of a tiling is the maximum over all tiles of triangle side lengths.

The first main theorem for this section is as follows.

THEOREM 5.2. [illegible] $T = \triangle abc$ [illegible] α, β, γ [illegible] a, b, c, [illegible] $\alpha < \gamma$ [illegible] α/π [illegible] $\mathcal{T}^0, \mathcal{T}^1, \ldots$ [illegible] T [illegible] i [illegible] η_i [illegible] $\mathcal{T}^i$ [illegible] (a) $\mathcal{T}^{i+1}$ [illegible] $\mathcal{T}^i$ [illegible] (b) $\eta_i \to 0$ [illegible] $i \to \infty$ [illegible] p, q [illegible] T [illegible]

$$\lim_{i\to\infty} \operatorname*{dist}_{\mathrm{Skel}(\mathcal{T}^i)}(p, q) = |pq|.$$

In other words, every straight-line path connecting two points (p, q) on the boundary of T is approximated with arbitrary accuracy by a path of edges of the tiling.

[illegible]. In order to prove this theorem, we require a simultaneous analysis of tilings of a conjugate triangle. Therefore, let us change notation so that the original triangle is T_1, its conjugate is T_2, and there are two sequences of generalized pinwheel tilings: $\mathcal{T}_1^0, \mathcal{T}_1^1, \mathcal{T}_1^2, \ldots$, which are tilings of T_1, and $\mathcal{T}_2^0, \mathcal{T}_2^1, \mathcal{T}_2^2, \ldots$, which are tilings of T_2.

Without loss of generality, let us further assume that $\mathcal{T}_\nu^0$ for $\nu \in \{1, 2\}$ is simply $\{T_\nu\}$, and that each subsequent $\mathcal{T}_\nu^i$ is obtained from $\mathcal{T}_\nu^{i-1}$ by applying ATP to exactly one tile (so that $\mathcal{T}_\nu^i$ has exactly $4i + 1$ tiles). This assumption is without loss of generality because we can take the original given sequence $\mathcal{T}_\nu^1, \mathcal{T}_\nu^2$, etc., and insert all intermediate tilings (i.e., if the original $\mathcal{T}_\nu^i$ was obtained from T_ν^{i-1} via l ATP operations, then we can insert $l - 1$ intermediate tilings in the sequence). If we prove that the limiting property holds for the augmented sequence, then it certainly also holds for the original sequence.

We make the following preliminary observation about GPTs. If $\mathcal{S}$ is any tiling of T_ν for $\nu = 1$ or 2 obtained by repeated ATP subdivision, then there exists an i such that $\mathcal{T}_\nu^i$ refines $\mathcal{S}$.

We use the following additional definitions and notation to prove the theorem.

- Let ∂T_ν denote the boundary of T_ν, $\nu = 1, 2$. Thus ∂T_1 and ∂T_2 are each unions of three segments. As noted above, these segments are called either [illegible] or [illegible] of T_ν.
- Define $X_\nu = \partial T_\nu \times \partial T_\nu$ for $\nu = 1, 2$, that is, the set of ordered pairs of boundary points.

- For any $(p,q) \in X_\nu$, let $f_\nu((p,q),n)$ denote $\mathrm{dist}_{\mathrm{Skel}(\mathcal{T}_\nu^n)}(p,q)$.
- For $(p,q) \in X_\nu$, let $g_\nu((p,q),n) = f_\nu((p,q),n)/\|p-q\|$, $\nu = 1,2$. If we took the maximum of this quantity over choices of (p,q), we would arrive at a quantity analogous to the "deviation ratio" introduced in section 3. Clearly $g_\nu((p,q),n) \geq 1$ for all p,q,n. If $p = q$, then define $g_\nu((p,p),n)$ to be 1.
- Let $f_\nu^*(p,q) = \inf_{n\geq 0} f_\nu((p,q),n)$. Note that $f_\nu((p,q),n)$ is a nonincreasing function of n (because every edge of $\mathcal{T}_\nu^n$ is covered by edges of $\mathcal{T}_\nu^{n+1}$ for all n), so this inf is also the limit of the sequence.
- Let $g_\nu^*(p,q) = \inf_{n\geq 0} g_\nu((p,q),n)$. Clearly $g_\nu^*(p,q) \geq 1$ for all p,q. For the same reason as above, g^* is also the limit of the sequence.

  The main theorem now reduces to showing that $g_1^*(p,q) = 1$ for all $(p,q) \in X_1$. Our proof technique requires us to claim more strongly that $g_\nu^*(p,q) = 1$ for all $(p,q) \in X_\nu$ for $\nu = 1$ and $\nu = 2$.
- Let η_ν be the length of the shortest edge of T_ν for $\nu = 1,2$. Let χ_ν be the length of the shortest edge among the five triangles that result from one application of ATP to T_ν, and let $\rho_\nu = \eta_\nu/\chi_\nu$.
- Let $X_\nu' \subset X_\nu$ denote ordered pairs (p,q) such that $p,q \in \partial T_\nu$ and such that $\mathrm{dist}_{\partial T_\nu}(p,q) \geq \chi_\nu$. Note that X_ν' is a compact set under usual product topology.

The reason for introducing X_ν' is that g_ν^* is continuous on X_ν' as we now proceed to show. (Once the main theorem is proved, we will have shown that g_ν^* is identically 1 and hence obviously continuous on all of X_ν, but continuity is not so easy to prove at this stage of the argument.) Observe that for $(p,q) \in X_\nu'$, $g_\nu^*(p,q) = f_\nu^*(p,q)/\|p-q\|$. We start with the following lemma.

LEMMA 5.3. [illegible] f_ν^* [illegible] X_ν

[illegible]. Let μ be the scalar such that $\mathrm{dist}_{\partial T_\nu}(p,q) \leq \mu\|p-q\|$ for all $(p,q) \in X_\nu$. This scalar depends on the sharpest angle of T_ν. Given two pairs $(p,q),(p',q') \in X_\nu$, and given $n > 0$, observe that $f_\nu((p,q),n) \leq f_\nu((p',q'),n) + \mu\|p-p'\| + \mu\|q-q'\|$. This follows because a possible path from p to q in $\mathrm{Skel}(\mathcal{T}_\nu^n)$ would follow p to p' along the outer boundary of T_ν (a distance of at most $\mu\|p-p'\|$), then p' to q' along the shortest path in $\mathrm{Skel}(\mathcal{T}_\nu^n)$ (a distance of $f_\nu((p',q'),n)$), followed by a path from q' to q along the outer boundary of T_ν. This argument is symmetric in (p,q) and (p',q'), so therefore we can combine the directions to claim

$$|f_\nu((p,q),n) - f_\nu((p',q'),n)| \leq \mu\|p-p'\| + \mu\|q-q'\|.$$

This inequality must hold in the limit of $n \to \infty$, thus proving that f_ν^* is continuous. □

This shows that the numerator in the formula $g_\nu^*(p,q) = f_\nu^*(p,q)/\|p-q\|$ is continuous. The denominator $\|p-q\|$ is also continuous and bounded away from 0 on X_ν'; hence g_ν^* is continuous on this set.

The following lemma shows that it suffices to analyze X_ν' rather than all of X_ν.

LEMMA 5.4. [illegible] $\nu = 1,2$

$$\sup\{g_\nu^*(p,q) : (p,q) \in X_\nu\} \leq \max(\sup\{g_1^*(p,q) : (p,q) \in X_1'\}, \sup\{g_2^*(p,q) : (p,q) \in X_2'\}).$$

[illegible]. Choose an arbitrary $(p,q) \in X_\nu$. This proof will show that

$$g_\nu^*(p,q) \leq \max(\sup\{g_1^*(p,q) : (p,q) \in X_1'\}, \sup\{g_2^*(p,q) : (p,q) \in X_2'\}).$$

Taking the supremum on the left will prove the result. If $(p,q) \in X_\nu'$, then the result is immediate since that value appears in one of the two terms in the right-hand side.

So assume for the rest of the proof that $(p,q) \in X_\nu - X'_\nu$. If p,q lie on the same side of T_ν, then the left-hand side is 1 because $f_\nu((p,q),n) = \|p-q\|$ for all n in this case since the boundaries of T_ν are covered by the edges of $\mathcal{T}_\nu^n$ for each n. Since the right-hand side is greater than or equal to 1, the result follows immediately.

The last case is that p,q are on distinct sides (and in particular, are not vertices of T_ν). In this case they must be less than distance χ_ν of the same vertex by definition of X'_ν. For the rest of the proof of this lemma, consider only the $\nu = 1$ case since the $\nu = 2$ case is similar.

First, suppose that p,q are both within distance χ_1 of vertex a of T_1. Without loss of generality, p lies on ab and q lies on ac. Consider the sequence of tiles $H_0 = T_1, H_1, H_2, \ldots$ such that H_i is the tile from $\mathcal{T}_\nu^i$ that contains vertex a. Each of these tiles is similar to T_1. The diameter of the H_i's tends to 0 as $i \to \infty$ by assumption. Thus, there is a k such that H_k contains both p and q but H_{k+1} fails to contain one or both of p or q. Let τ be the length of the shortest side of H_k. We claim that either $\|a-p\| \geq \tau/\rho_1$ or $\|a-q\| \geq \tau/\rho_1$. The reason is that if both $\|a-p\| < \tau/\rho_1$ and $\|a-q\| < \tau/\rho_1$, then p,q would both lie on the boundaries of H_{k+1} since the side lengths of H_{k+1} are all at least τ/ρ_1 by definition of ρ_1. This would contradict the choice of k.

As mentioned above, H_k is similar to T_1, and the constant of proportionality is τ/η_1. Note that H_k could be either a dilation of T_1 with no reflection or a dilation of T_1 with a reflection. Assume the former case since the latter is similar. There exist $\bar{p}$ and $\bar{q}$ lying on sides ab, ac of T_1 whose positions with respect to ab, ac are proportional to the positions of p,q with respect to the two sides of H_k. Refer to Figure 5.1. Since either $\|a-p\| \geq \tau/\rho_1$ or $\|a-q\| \geq \tau/\rho_1$ and the scaling factor between H_k and T_1 is τ/η_1, this means that at least one of $\bar{p}$, $\bar{q}$ is distance from a greater than or equal to $(\tau/\rho_1)/(\tau/\eta_1) = \chi_1$. Hence $(\bar{p},\bar{q}) \in X'_1$. For an arbitrary $n > 0$, consider the tiling $\mathcal{T}_*^n$ of H_k that is obtained by shrinking $\mathcal{T}_1^n$ by a factor of τ/η_1 and translating it so that it lies on top of H_k. The shortest path between p and q in this tiling is $f_1((\bar{p},\bar{q}),n)\tau/\eta_1$ by scaling. Also, there is an n' such that the portion of $\mathcal{T}_1^{n'}$ lying in H_k is strictly a refinement of $\mathcal{T}_*^n$ by the observation made at the beginning of the proof. The distance between p and q in this tiling is $f_1((p,q),n')$, and since $\mathcal{T}_1^{n'}$ refines $\mathcal{T}_*^n$, $f_1((p,q),n') \leq f_1((\bar{p},\bar{q}),n)\tau/\eta_1$. Note that $\tau/\eta_1 = \|p-q\|/\|\bar{p}-\bar{q}\|$ by similarity, so the previous inequality implies $g_1((p,q),n') \leq g_1((\bar{p},\bar{q}),n)$. Take the infimum over all n of both sides to conclude that $g_1^*(p,q) \leq g_1^*(\bar{p},\bar{q})$, thus establishing the lemma in this case.

In case that p,q are both within distance χ_1 from vertex b, the lemma follows by the same argument since the triangles containing b in all subdivisions of T_1 are similar to T_1.

The last case is that p,q are both within distance χ_1 of vertex c. Say, e.g., that p lies on ac and q on bc. In this case, the argument is slightly more complicated since there are two triangles containing c in the next level of subdivision. Let cf be the segment that is the common boundary to the two triangles of the next level of subdivision that meet vertex c. (Refer to Figure 4.1.) Let r be the point where segment pq crosses edge cf. Then the argument above shows that the infimum over n of the distance between p and r using edges from $\mathcal{T}_1^n$ is less than or equal to $|pr| \cdot g_2^*(\bar{p},\bar{r})$, where $(\bar{p},\bar{r}) \in X'_2$. Similarly, the infimum over n of the distance between r and q using edges from $\mathcal{T}_1^n$ is less than or equal to $|rq| \cdot g_1^*(\bar{r},\bar{q})$, where $(\bar{r},\bar{q}) \in X'_1$. Therefore, $f_1^*(p,q) \leq |pq| \cdot \max(\sup\{g_1^*(p,q) : (p,q) \in X'_1\}, \sup\{g_2^*(p,q) : (p,q) \in X'_2\})$ and thus the result follows. ∎

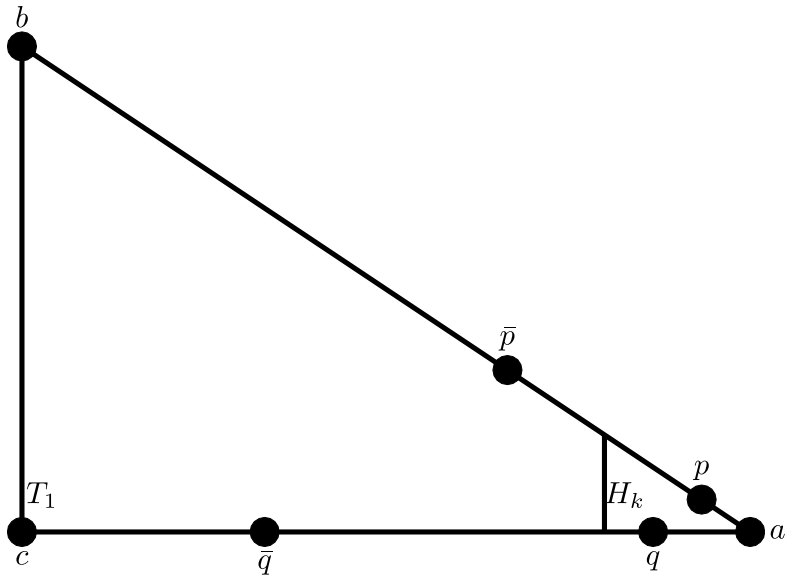


FIG. 5.1. *Construction in the proof of Lemma* 5.4. *The idea of the proof is that* $g_1^*(p,q)$ *cannot exceed* $g_1^*(\bar{p},\bar{q})$, *where* $\bar{p}$ *and* $\bar{q}$ *have positions in* T_1 *that are proportional to the positions of* p, q *in* H_k, *since any* (p,q) *path of tile edges in* H_k *for any particular tiling will eventually also occur (scaled up) as a* $(\bar{p},\bar{q})$ *path of tile edges in* T_1.

Finally, we conclude the proof of the main theorem by showing that $\sup\{g_\nu^*(p,q) : (p,q) \in X_\nu'\} = 1$ for $\nu = 1, 2$. To this end, choose the ν (either 1 or 2) that maximizes $\sup\{g_\nu^*(p,q) : (p,q) \in X_\nu'\}$. Without loss of generality, say $\nu = 1$ is chosen.

Since X_1' is a compact set and g_1^* is continuous on this set, there exists a (p^*, q^*) in X_1' that maximizes $g_1^*(p,q)$. If p^*, q^* lie on the same side of T_1, then $g_1^*(p^*, q^*) = 1$; thus the proof is finished. Else assume (for a contradiction) that $g_1^*(p^*, q^*) = \sigma > 1$. Let H^* be the line segment joining p^* and q^*.

Choose n_1 large enough so that there exists a tile T' in $\mathcal{T}_1^{n_1}$ such that $T' \cap H^*$ has positive length and is contained in the middle third H^*. Let the longest edge of T' be ω. For reasons to be explained below, we also choose n_1 large enough so that

$$\omega < \frac{(\sqrt{\sigma} - 1)\zeta(\alpha, \beta, \gamma, \arccos(1/\sqrt{\sigma}))}{6\sigma} |p^* q^*|, \tag{5.1}$$

where $\zeta()$ is the function defined by Lemma 5.1. Continue splitting until we reach split number $n_2 \geq n_1$ so that within $\mathcal{T}_1^{n_2}$, there exists a tile T'' in $\mathcal{T}_1^{n_2}$ lying inside in T' that has an edge making an angle θ with H^*, with $\cos\theta > 1/\sqrt{\sigma}$. This is possible by Lemma 5.1. See Figure 5.2. Let κ_2 be the length of this edge. By the second part of the lemma, we may assume $\kappa_2 \geq \zeta(a, b, c, \iota)\omega$, where $\iota = \arccos(1/\sqrt{\sigma})$.

Let v_1, v_2 be the endpoints of this edge with v_1 being the vertex near p^* and v_2 near q^*. Observe that $p^* v_1$, $v_1 v_2$, and $v_2 q^*$ make up a three-segment path from p^* to q^*. The length of this path is $\kappa_1 + \kappa_2 + \kappa_3$, where $\kappa_1 = |p^* v_1|$ and $\kappa_3 = |v_2 q^*|$. (We have already defined $\kappa_2 = |v_1 v_2|$.) Let λ_1, λ_2, and λ_3 be the lengths of the projections of $p^* v_1$, $v_1 v_2$, $v_2 q^*$, respectively, onto H^*. Because v_1 lies within T' while H^* crosses through T', the distance from v_1 to H^* is at most ω; hence

$$\kappa_1 \leq \sqrt{\lambda_1^2 + \omega^2} = \lambda_1 \sqrt{1 + \omega^2/\lambda_1^2}$$

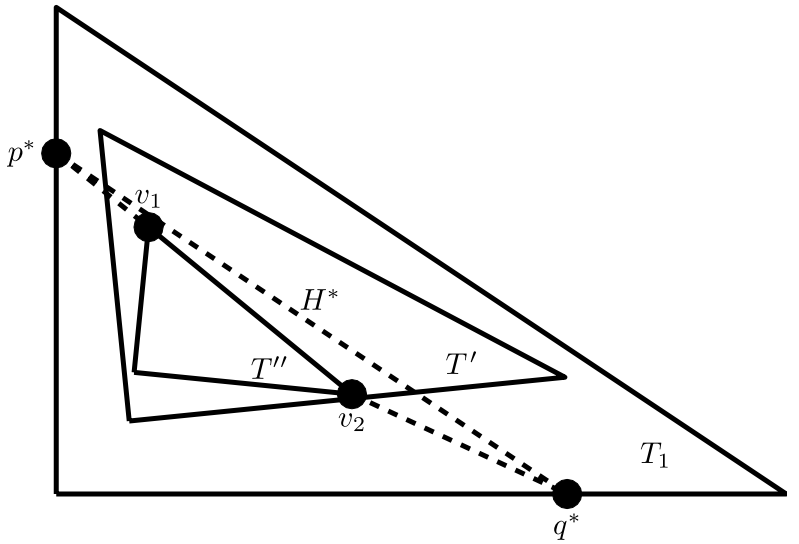


FIG. 5.2. *Construction for the proof of Theorem* 5.2. *In this figure,* $(p^*, q^*) \in X_1'$ *is the maximizer of* g_1^*. *Points* p^*, q^* *are connected by line segment* H^*. *Edge* v_1v_2 *of triangle* T'' *is very closely aligned with (i.e., makes a small angle with respect to)* H^*. *The assumption that* $g_1^*(p^*, q^*) > 1$ *leads to a contradiction because a path whose length relative to* H^* *is shorter than* $g_1^*(p^*, q^*)$ *is found by using tile edges to connect* p^* *to* v_1, *then the* v_1v_2 *line segment is followed, and then tile edges are used to connect* v_2 *to* q^*.

$$\begin{aligned} &\leq \lambda_1 \left(1 + \omega^2/\lambda_1^2\right) \\ &= \lambda_1 + \frac{\omega^2}{\lambda_1} \end{aligned}$$

and hence

$$\kappa_1 - \lambda_1 \leq \frac{\omega^2}{\lambda_1} \leq \frac{3\omega^2}{|p^*q^*|}. \tag{5.2}$$

(The factor of 3 arises because $\|p^* - v_1\| \geq \|p^* - q^*\|/3$ as assumed earlier.) Similarly,

$$\kappa_3 - \lambda_3 \leq \frac{3\omega^2}{|p^*q^*|}. \tag{5.3}$$

Next, because $\cos\theta > \frac{1}{\sqrt{\sigma}}$, where θ is the angle between H^* and v_1v_2, we have $\sqrt{\sigma}\lambda_2 > \kappa_2$. Therefore,

$$\sigma\lambda_2 - \kappa_2 > (\sqrt{\sigma} - 1)\kappa_2. \tag{5.4}$$

Next, note that

$$f_1^*(p^*, q^*) \leq \sigma\kappa_1 + \kappa_2 + \sigma\kappa_3, \tag{5.5}$$

thanks to the existence of the three-edge path $p^*v_1v_2q^*$. The reasoning is as follows. From p^* to v_1 there is a straight-line path of length κ_1. This path cuts through a finite list of triangles, say l triangles, within the tiling $\mathcal{T}_1^{n_2}$, since p^* and v_1 both lie on triangle edges of this tiling. Let the individual segments within these triangles be

of length $\mu_1, \ldots, \mu_l$. By construction, these quantities sum to κ_1. Then by further refinement, we can find paths within the tiling with lengths less than or arbitrarily close to $\mu_1\sigma$, $\mu_2\sigma$, etc., since σ is the factor that is the maximum amount longer that an edge path in refinements of either T_1 or T_2 can be versus the straight-line path. So the infimum of the lengths of these paths added up is at most $\sigma\kappa_1$. The same reasoning accounts for the term $\sigma\kappa_3$ in (5.5). Finally, the edge v_1v_2 is length κ_2 and is already in the tiling.

Use (5.2), (5.3), (5.4) and the equality $|p^*q^*| = \kappa_1 + \kappa_2 + \kappa_3$ to bound the right-hand side of (5.5):

$$\begin{aligned} f_1^*(p^*, q^*) &\leq \sigma\kappa_1 + \sigma\kappa_3 + \kappa_2 \\ &= \sigma(\kappa_1 - \lambda_1) + \sigma(\kappa_3 - \lambda_3) - (\sigma\lambda_2 - \kappa_2) + \sigma(\lambda_1 + \lambda_2 + \lambda_3) \\ &< 6\omega^2 \frac{\sigma}{|p^*q^*|} - (\sqrt{\sigma} - 1)\kappa_2 + \sigma|p^*q^*|. \end{aligned} \tag{5.6}$$

Multiply (5.1) by ω on both sides and use the fact that $\kappa_2 \geq \zeta(\alpha, \beta, \gamma, \iota)\omega$ to obtain

$$\omega^2 < \frac{(\sqrt{\sigma} - 1)\kappa_2}{6\sigma}|p^*q^*|. \tag{5.7}$$

Substituting (5.7) in (5.6) shows that $f_1^*(p^*, q^*) < \sigma|p^*q^*|$. But this is a contradiction, because the hypothesis of this analysis was that $g_1^*(p^*, q^*) = \sigma$; i.e., $f_1^*(p^*, q^*) = \sigma|p^*q^*|$. □

The preceding theorem has the drawback that it pertains only to paths starting and ending on the boundary of the root triangle. For isoperimetry, we would like to generalize the result to paths with arbitrary interior p and q. Since the nodes of the pinwheel tiling are dense in the interior (in the limit as the mesh size is refined), the following theorem provides a suitable generalization and will be taken as our definition of the isoperimetric property.

THEOREM 5.5. $\mathcal{T}^0, \mathcal{T}^1, \ldots$ $T = \triangle abc$ $\alpha < \gamma$ α/π $\mathcal{T}^{i+1}$ $\mathcal{T}^i$ $i = 0, 1, 2, \ldots$. p, q $\mathrm{Skel}(\mathcal{T}^n)$ n

$$\lim_{\substack{m \to \infty \\ m \geq n}} \operatorname{dist}_{\mathrm{Skel}(\mathcal{T}^m)}(p, q) = \|p - q\|.$$

. Consider the segment pq lying in T. Let $\epsilon > 0$ be given. Make a list $U_1, \ldots, U_l$ of tiles in $\mathcal{T}^n$ traversed by this segment. Since pq crosses U_i, define p_iq_i to be $U_i \cap pq$. Observe that p_i, q_i both lie on the boundary of U_i. By the preceding theorem, after a sufficient number of further subdivisions (say s), there exists a path in $\mathrm{Skel}(\mathcal{T}^{n+s})$ between p_i and q_i of length $|p_iq_i|(1 + \epsilon)$. This choice of s depends on i, so take the maximum such value of s (maximum over all $i = 1, \ldots, l$). Then there is a path in $\mathrm{Skel}(\mathcal{T}^{n+s})$ from p to q of length at most

$$|p_1q_1|(1 + \epsilon) + |p_2q_2|(1 + \epsilon) + \cdots + |p_rq_r|(1 + \epsilon),$$

i.e., at most $|pq|(1 + \epsilon)$. □

Algorithm PINW 1.0—Tiling Phase

1. Generate a mesh for Ω with bounded aspect ratio using Triangle.
2. Split triangles too close to equilateral at their in-centers.
3. Split triangles whose smallest angle is a rational multiple of π at a point near the in-center.
4. Let the set of triangles obtained after steps 1–3 be called $\mathcal{T}_0$.
5. Initialize a heap containing triangles that need splitting. The triangles are ordered so that the one whose minimum altitude is maximum is at the top of the heap. Initially the heap contains all triangles from $\mathcal{T}_0$.
6. Repeatedly remove a triangle from the heap and split it with ATP until the size of the top element of the heap is sufficiently small according to the user's specification.

FIG. 6.1. *Summary of first phase of Algorithm PINW in which a tiling of the input region is generated.*

Algorithm PINW 1.0—Simplicial Complex Phase

7. Let $\mathcal{T}_*$ be the set of tiles including those in $\mathcal{T}_0$ and all their descendants obtained by subdivision. Let $\mathcal{T}_f \subset \mathcal{T}_*$ be the set of leaf tiles.
8. Loop over all tiles in $\mathcal{T}_*$ starting from the coarsest to determine the value of big(E) for each edge E of any tile.
9. For each big edge (i.e., each edge in the image of the "big" operator), select one side as [illegible] and the other as [illegible]. Sort the list of nodes lying on the staying side of each such edge.
10. Loop over tiles in $\mathcal{T}_* - \mathcal{T}_f$ starting from the coarsest. For each such tile T and for each of its vertices d, e, f as labeled in Figure 4.1, let E be the big edge containing the particular vertex. If this vertex d, e, or f is on the moving side of E and is very close to a vertex v' on the staying side, then displace it to coincide with v' and apply the induced affine transformation to subtriangles of T.
11. Apply Delaunay triangulation to each leaf tile. The leaf tiles have been distorted by the affine transformations in the previous step. Their edges are subdivided because of the presence of hanging nodes. The collection of triangles output from this step is a simplicial mesh of Ω.

FIG. 6.2. *Summary of second phase of Algorithm PINW in which the tiling from the first phase is converted to a simplicial complex; i.e., hanging nodes are eliminated.*

6. Meshing an arbitrary region. In this section we present our algorithm PINW to mesh a region Ω with arbitrary polygonal boundary. The algorithm has two phases: generation of a tiling, which is summarized in Figure 6.1, and conversion of the tiling to a simplicial mesh, which is summarized in Figure 6.2. The steps in these summaries are described in more detail in the remainder of this section. The current version of PINW is 1.0 and has been coded in MATLAB. An example output from this algorithm is shown in Figure 6.3.

We first start with a coarse triangulation of the domain. We use the [illegible] package [9] developed by Shewchuk, which uses Delaunay triangulation. The triangles

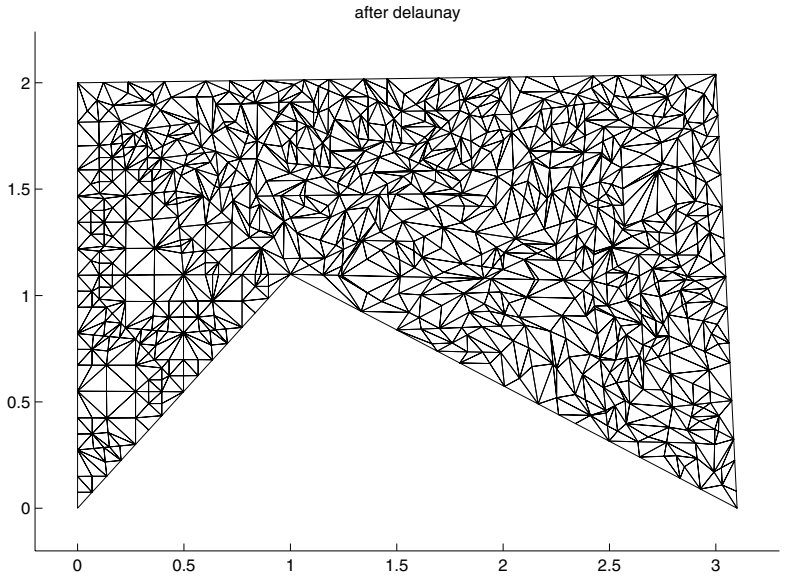


FIG. 6.3. *The coarse mesh for this figure had three triangles. The final mesh after pinwheel tiling, collapse-node operations, and Delaunay triangulation is shown.*

produced have bounded aspect ratio. The second preliminary step, as mentioned in section 4, locates triangles too close to equilateral and splits them at their in-centers.

A third preliminary step is to identify and split triangles whose smallest angle α is a rational multiple of π. (As noted above, the proof of isoperimetry requires that α/π be irrational.) In principle, this test could be conducted exactly using number-theoretic methods since the coordinates of the vertices of each triangle, being floating-point numbers, are rational numbers and can be treated with integer algorithms by clearing common denominators. Modifying a triangle in which α is a rational multiple of π is trivial in principle because any small random perturbation of a node of such a triangle will lead to an angle that is not a rational multiple of π with probability 1.

In practice, this exact test and solution are both undesirable. For practical use of the algorithm, we would like to avoid the case when α is close to a rational multiple of π of the form $m\pi/n$, where n is a small integer. The reason is that the presence of such a triangle in which α/π is irrational but is close to m/n implies that, although the isoperimetric property is asymptotically valid, the available angles will be badly distributed (clustered around multiples of π/n) for modest levels of refinement.

Therefore, a more practical heuristic is to check each smallest angle against a finite list of the form $m\pi/n$, where m, n range over a preselected set of small integers. If a triangle's smallest angle comes too close to a member of this list, then the triangle is either split into three using a point near its in-center or is perturbed. (The exact in-center obviously should not be used since this would replace each angle by half its previous value, and hence will still be close to a rational multiple of π.) This step has not been implemented in the current version of our code PINW 1.0 because we are still seeking the best practical heuristic. (Indeed, in Figure 6.3, one coarse triangle is close to a 45° right triangle, and hence one part of the subdivision exhibits a shortage of possible directions.)

Let $\mathcal{T}_0$ be the list of triangles that are produced by these preliminary steps. Thus, the triangles in $\mathcal{T}_0$ form a simplicial triangulation of the input set Ω. We call these

triangles the [illegible]. The generalized pinwheel subdivision is then performed on the root tiles to obtain a refined tiling. The procedure to refine the mesh used in PINW 1.0 is based on a simple heap [1]. The heap is initialized with all triangles in $\mathcal{T}_0$, which are ordered in the heap according to length of the minimum altitude. The main loop for the subdivision is to remove the top member of the heap (i.e., the unsubdivided tile with the largest value of minimum altitude), apply ATP to it, replace it with its five children. The procedure terminates when the top triangle in the heap is smaller than the user-specified mesh size requirement.

From this description, it is apparent that PINW 1.0 supports a single global user-specified mesh size requirement. For many applications of mesh generation, it is useful to have a finer mesh in one part of the domain versus another. This can also be implemented in the framework of generalized pinwheel subdivision but is not available in PINW 1.0. In addition, several aspects of our analysis that follows below would have to be generalized to cover graded meshes.

Once the subdivision procedure is complete, the resulting tiling must be converted to a simplicial mesh. For the 1:2 pinwheel triangulation, this step is quite straightforward as mentioned in section 3. In the generalized case, however, it is much more complicated and involves several steps that we shall now describe.

Let $\mathcal{T}_*$ be the list of all tiles in the hierarchy: It includes the tiles in $\mathcal{T}_0$ and all their descendants from the subdivision procedure. The tiles in $\mathcal{T}_*$ naturally have a forest structure associated with them in which the forest roots are root tiles. Let [illegible] denote a triangle in $\mathcal{T}_*$ that is not further subdivided during the generalized pinwheel subdivision phase. Let $\mathcal{T}_f$ be the set of leaf tiles.

The first step in converting the tiling to a mesh is to identify for each edge E of each tile $T \in \mathcal{T}_*$ the edge that we denote big(E). This is defined as follows. For the edges of a root tile, big() is the identity operator; i.e., for each edge E of a root tile, big(E) = E. Every nonroot tile arises from ATP subdivision, and hence each of its edges can be identified with one of the labeled edges in Figure 4.1. For the interior edges labeled de, df, dg, and cf in the figure, big() is taken to be the identity operator. For the interior edges labeled cg and gf, big(cg) = big(gf) = cf. For exterior edges ad, dc, cb, bf, fe, and ea, big() is inherited from the edge of $\triangle abc$ that contains this edge; i.e., big(ad) and big(dc) are defined to be big(ac), etc.

This definition implies in all cases that big(E) for an edge E of triangle T is the edge E' of an ancestor triangle T' in the subdivision hierarchy (i.e., T is derived from T' via a sequence of zero or more ATP operations) such that $E \subset E'$, and such that E' is maximal with this property. In other words, there is no other ancestor of T with an edge E'' that strictly contains E'.

If T is a nonroot triangle in the hierarchy and its parent's edges have already been labeled with their values of big(), then the edges of T can be labeled in a constant number of operations. Thus, assuming big() is determined starting from the root tiles and working downward, the total number of operations to determine big() for all edges is proportional to $|\mathcal{T}_*|$.

Next, for each "big" edge E (that is, an edge such that big(E) = E), identify a [illegible] and a [illegible] side. This choice can be arbitrary except for two stipulations. The moving side of an edge E that lies on the exterior boundary of Ω should be facing the exterior of Ω, which implies that no tiles lie on its moving side. The moving side of an edge in correspondence with cf in Figure 4.1 should be facing vertex b. PINW now identifies all the nodes on the staying side of E, i.e., nodes that are vertices of tiles adjacent to E and lying on its staying side. These nodes are sorted in order of occurrence on E. This sorted list is saved for the next step of the algorithm.

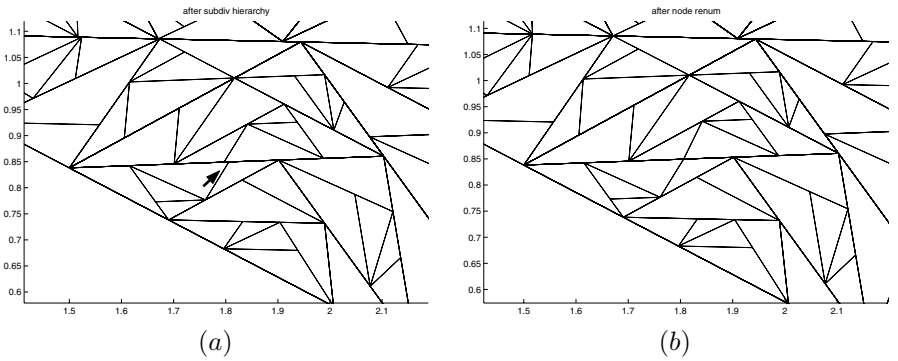


FIG. 6.4. *Example of a collapse-node operation is shown. A node on one side of a "big" edge that lies within the tolerance of a node on the other side is moved and merged with the nearby node on the other side.*

In the next step, PINW loops over triangles in $\mathcal{T}_* - \mathcal{T}_f$ starting from the coarsest and performs collapse-node operations on each. Let T be a tile in $\mathcal{T}_* - \mathcal{T}_f$. Let the four vertices of T introduced when it is subdivided be labeled d, e, f, g as in Figure 4.1. No operation is performed for g since it is on the staying side of edge cf. The big edge containing d is big(ac); call this $B(d)$. The big edge containing e and f is big(ab); call this $B(e)$ and also call it $B(f)$. Let v stand for one of d, e, f. PINW checks whether v (i.e., tile T) is on the moving side of $B(v)$. If it is on the staying side, then no further operation is performed. If it is on the moving side, then PINW finds the vertex v' taken from the staying side of $B(v)$ that is closest to v. This v' can be found efficiently using binary search on the precomputed sorted lists described in the previous paragraph. If $\|v - v'\| \le \delta$, nodes v and v' are collapsed by displacing v to v'. Here δ is a tolerance defined by (7.9) below.

This displacement induces uniquely determined affine transformations on triangles contained in T as follows. If v is the vertex labeled d in Figure 4.1, then there is a unique affine transformation on $\triangle adf$ that leaves a and f fixed and moves d to v'. A second affine transformation of $\triangle cdf$ leaves f and c fixed and moves d to v'. If v is the vertex labeled e, then there are unique affine transformations determined for $\triangle ade$ and $\triangle def$. Finally, if v is the vertex labeled f, then there are transformations for each of $\triangle def$, $\triangle cdf$, and $\triangle bcf$. The algorithm applies all the relevant affine transformations caused by motion of the node. Note that the affine transformations agree on the boundaries between these triangles, so there is no consistency issue regarding which transformation to apply. These transformations move the triangle, including every node at deeper levels of the hierarchy contained in it. This concludes the description of the operation. See Figure 6.4 for an illustration of this operation.

Note that a single tolerance δ is used to determine motion. The theoretical value for δ is given by (7.9) below. We will verify later that this value of δ is sufficiently small so that two important properties hold:

1. If a vertex v' is the target of a collapse-node move, then it should be uniquely determined; i.e., there should not be two vertices v' and v'' on the staying side of $B(v)$ that are both within distance δ of v.
2. No two vertices on the moving side of big(E) for any E should be collapsed to the same node on the staying side.

In a future extension of PINW to handle graded meshes, presumably the value of δ would not be a single global value.

The affine transformations described above have the property that all of the segments illustrated in Figure 4.1 remain straight (collinear) segments after the transformation. It is apparent that each collapse-node operation could cause many nodes to move. We will say that the one node v that is displaced to match v' is [illegible] displaced. The nodes moved by virtue of an affine transformation induced by moving v are said to be [illegible] displaced.

A collapse-node operation, once executed, cannot be undone by future collapse-node operations. The reason is that v is never moved again. It is never moved again directly since it can be moved directly only when the tile $T = \triangle abc$ that caused its creation is processed. It can also never be moved again indirectly since there is no tile in lower levels of the hierarchy that contains it except as a corner vertex, and corner vertices of a triangle T' are not moved when T' is processed. Similarly, v' can never be moved again. The reason is that v' is never moved directly (since it is on the staying side of its big edge). Any transformation that might move v' indirectly takes place at a level of the hierarchy higher than the level of T.

We carry out all available collapse-node operations for all triangles in the order described. Once all collapse-node operations are complete, we are left with the collection of distorted leaf tiles, each of which may have one or more hanging nodes. These hanging nodes are collinear with the endpoints of the edges on which they hang because, as noted above, we do not disturb any collinearity relationships with collapse-node operations. The hanging nodes are all at least δ apart from the corners and from each other.

For each of these distorted tiles, we compute its Delaunay triangulation (including the hanging nodes). The collection of all of these Delaunay triangles forms a simplicial mesh that is the final output of PINW.

The running time of PINW is analyzed as follows. Let n be the number of leaf tiles. Then the total number of tiles is $O(n)$, as is the total number of vertices, edges, and triangles of the final simplicial mesh. The heap insertions and deletions require $O(n \log n)$ total operations. Sorting all the lists associated with big edges requires $O(n \log n)$ operations. Looking up a vertex in a sorted list requires $O(\log n)$ operations for binary search; hence all of the lookups to see if a node should be collapsed require $O(n \log n)$ operations.

The recursive application of affine transformations requires $O(nd)$ operations since each vertex is transformed at most d times, where d is the maximum depth of the forest associated with $\mathcal{T}^*$. We claim $d = O(\log n)$. It follows from Lemmas 7.4 and 7.5 in the next section that the minimum altitude of a triangle at depth k lies between $\mu_0 \xi_1^k$ and $\mu_1 \xi_2^k$, where μ_0, μ_1 are lower and upper bounds on the minimum altitudes among root tiles, ξ_2 is an absolute constant, and ξ_1 is a scalar depending on the worst aspect ratio among root tiles. This means that a leaf tile can be at most a factor $\log \xi_2 / \log \xi_1$ (asymptotically) deeper in the forest than any other leaf tile. Thus, all leaves have depth $O(\log n)$.

Finally, the Delaunay triangulation operations in the last step of the algorithm also require $O(n \log n)$ operations total. Overall, we see that PINW requires $O(n \log n)$ operations.

7. Analysis of aspect ratio. In this section we analyze the aspect ratios of triangles produced by PINW, showing that they are bounded above by a number that depends only on the sharpest angle in the original polygon Ω. The first part

of this section obtains lower and upper bounds on the minimum altitudes of the five subtriangles resulting from ATP with respect to the initial triangle. We then explain how to choose δ, the tolerance parameter of algorithm PINW described in the previous section, which will be proportional to the minimum altitude among leaf tiles. Finally, we analyze the aspect ratio of the triangles produced by PINW. Let $\text{minalt}(T)$ denote the minimum altitude of triangle T.

LEMMA 7.1. [illegible] $\mu = \text{minalt}(T)$ [illegible] T [illegible] μ [illegible] T [illegible] μ [illegible] $\text{minalt}(T) \le \mu$

[illegible]. The first part of the lemma is quite trivial: Draw a line through the longest side length of T and a parallel line through the opposite vertex. These lines are distance μ apart. The argument for the converse is as follows. Without loss of generality, let the two lines be parallel to the x-axis. Let the vertices of T be numbered v_1, v_2, v_3 such that v_1 is closest to the bottom line (i.e., has minimal y-coordinate among the three vertices) and v_2 is closest to the top line. By reflecting if necessary, assume also that the x-coordinate of v_1 is less than or equal to the x-coordinate of v_2. Now draw the line v_1v_2, which is a transverse to the two parallel lines. If v_3 is below (to the right of) this line, then it is easy to see that the entire triangle T may be rotated clockwise about v_1 until v_1v_3 becomes horizontal, and during this whole rotation, all three vertices remain between the lines. On the other hand, if v_3 is above (to the left of) the line, then rotate T clockwise about v_2 until v_2v_3 becomes horizontal.

Once T has been reoriented so that one of its edges is horizontal, the claim is trivial since the altitude to the horizontal edge is a vertical line segment and hence must have length no more than μ. □

COROLLARY 7.2. [illegible] T_1, T_2 [illegible] $T_1 \subset T_2$ [illegible] $\text{minalt}(T_1) \le \text{minalt}(T_2)$

[illegible]. Draw the two parallel lines for T_2 as in the previous theorem; clearly T_1 also lies between them. □

LEMMA 7.3. [illegible] T [illegible] v_1, v_2, v_3 [illegible] T' [illegible] v_1', v_2, v_3 [illegible] A [illegible] T [illegible] T' [illegible] L [illegible]

$$1 - \eta/\mu \le \frac{\text{length}(A(L))}{\text{length}(L)} \le 1 + \eta/\mu, \tag{7.1}$$

[illegible] $\eta = |v_1v_1'|$ [illegible] μ [illegible] v_1 [illegible] v_2v_3

[illegible]. Without loss of generality, assume μ is 1 and η is replaced by $\iota = \eta/\mu$. Furthermore, without loss of generality, let T be positioned so that its v_2v_3 edge is a subsegment of the x-axis and v_1 lies on the y-axis (hence $v_1 = (0, 1)$ by the previous assumption). With these assumptions, the affine tranformation A in the question becomes a linear transformation (i.e., no additive term) since the x-axis (and the origin in particular) is invariant. The transformation maps $(1, 0)^T$ to $(1, 0)^T$ and $(0, 1)^T = v_1$ to v_1'. Let $r = v_1' - v_1$, and let r be written as $(\rho_1, \rho_2)^T$ so that $\rho_1^2 + \rho_2^2 = \iota^2$. Then A corresponds to the matrix

$$A = \begin{pmatrix} 1 & \rho_1 \\ 0 & 1 + \rho_2 \end{pmatrix}.$$

The minimum and maximum distortion of a line segment under a linear transformation is governed by the minimum and maximum singular values of the transformation.

Thus, the question now hinges on the two singular values of A. Notice that A may be regarded as a perturbation of the identity matrix, which has two singular values equal to 1. Therefore, by Corollary 8.6.2 of [3], the largest singular value of A is at most $1+\iota$, and the smallest singular value is at least $1-\iota$. These values are attainable by taking $\rho_1 = 0$ and $\rho_2 = \pm\iota$. □

LEMMA 7.4. [illegible] 4.1 [illegible] T [illegible] $\beta \geq \min(.4, \alpha)$ [illegible] $\gamma - \alpha \geq \min(.4, \alpha)$ [illegible] T' [illegible] [illegible] $\text{minalt}(T') \leq 0.9725\,\text{minalt}(T)$

[illegible] 1. The assumptions are valid for all tiles produced by PINW. For root tiles, we have ordered the angles $\alpha \leq \beta \leq \gamma$, and we know $\gamma - \alpha \geq .4$ because of the preliminary step of splitting near-equilateral triangles. For conjugates of root triangles, say $\alpha = \alpha'$, $\beta = \gamma' - \alpha'$, and $\gamma = \pi - \gamma'$, where α', β', γ' are the angles of a root tile, we also know that $\gamma - \alpha = \pi - \gamma' - \alpha' = \beta' \geq \alpha$ and that $\beta = \gamma' - \alpha' \geq .4$.

[illegible] 2. The factor 0.9725 is due to our proof technique and appears to be an overestimate. A search over a fairly dense grid of possible angles satisfying the hypotheses of the theorem indicates that the true bound is closer to 0.92.

[illegible]. We start by observing that $|bc|/|ab| = \sin\alpha/\sin\gamma$ by the law of sines. We know that either $\gamma \geq 2\alpha$ or $\gamma \geq \alpha + .4$. Furthermore, we know that either $\pi - \gamma \geq 2\alpha$ or $\pi - \gamma \geq \alpha + .4$ since $\pi - \gamma = \alpha + \beta$. Therefore, if we define $\bar{\gamma} = \min(\gamma, \pi - \gamma)$, then we can claim (a) either $\bar{\gamma} \geq 2\alpha$ or $\bar{\gamma} \geq \alpha + .4$, (b) $\bar{\gamma} \leq \pi/2$, and (c) $\sin\bar{\gamma} = \sin\gamma$, implying (d) $|bc|/|ab| = \sin\alpha/\sin\bar{\gamma}$. On the interval $[0, \pi/2]$, the sine is increasing. In the case that $\bar{\gamma} \geq 2\alpha$, we have $\sin\bar{\gamma} \geq \sin 2\alpha = 2\sin\alpha\cos\alpha \geq \sqrt{2}\sin\alpha$. The last inequality follows because $\alpha \leq \pi/4$ (by assumptions that $\bar{\gamma} \leq \pi/2$ and $\bar{\gamma} \geq 2\alpha$) so $\cos\alpha \geq 1/\sqrt{2}$. The other case is that $\bar{\gamma} \geq \alpha + .4$. Since sine is concave and increasing on $[0, \pi/2]$ the maximum value for $\sin\alpha/\sin\bar{\gamma}$ is attained at $(\alpha, \bar{\gamma}) = (\pi/2 - .4, \pi/2)$; hence $\sin\alpha/\sin\bar{\gamma} \leq .9211$. Thus,

$$|bc|/|ab| \leq .922. \tag{7.2}$$

Observe that subtriangle V, which is denoted T_{V}, is similar to T except scaled by a factor $|bc|/|ab|$. This proves $\text{minalt}(T_{\text{V}}) \leq .9211\,\text{minalt}(T)$.

Next, by similarity, $|bf|/|bc| = |bc|/|ab|$; hence, squaring (7.2) yields $|bf|/|ab| \leq .849$. This means

$$|af|/|ab| \geq .151. \tag{7.3}$$

Since $\triangle adf$ is isosceles, $|ad|/|af| \geq .5$ and $|df|/|af| \geq .5$; hence

$$|ad|/|ab| \geq .075 \tag{7.4}$$

and

$$|df|/|ab| \geq .075. \tag{7.5}$$

Next, by the law of sines applied to $\triangle cdf$, $|dc|/|df| = \sin\beta/\sin(\gamma - \alpha)$. We now take three cases: either $\beta < .4$, $\beta \in [.4, \pi/2]$, or $\beta > \pi/2$. In the first case $\beta \geq \alpha$ since the assumption in the lemma is $\beta \geq \min(.4, \alpha)$. Also, $\gamma - \alpha \geq \pi/2$ since $\gamma - \alpha + 2\alpha + \beta = \pi$ and $2\alpha + \beta < 1.2$ by the assumption for this case, so $\gamma - \alpha \geq \pi - 1.2$. This means $\sin(\gamma - \alpha) = \sin(\pi - (\gamma - \alpha)) = \sin(2\alpha + \beta)$, with $2\alpha + \beta \leq \pi/2$. Next, $2\alpha + \beta \leq 3\beta$, so $\sin(\gamma - \alpha) = \sin(2\alpha + \beta) \leq \sin(3\beta)$ (since $2\alpha + \beta \leq 3\beta$ and sine is increasing on $[0, \pi/2]$); thus $\sin\beta/\sin(\gamma - \alpha) \geq \sin\beta/\sin(3\beta)$. Now $\sin(3\beta) = \sin\beta(3\cos^2\beta - \sin^2\beta)$. Thus,

$\sin\beta/\sin(\gamma-\alpha) \geq 1/(3\cos^2\beta - \sin^2\beta)$. Since $\sin^2$ is increasing while $\cos^2$ is decreasing on $[0, .4]$, the minimum value of this fraction is when $\beta = 0$, so $\sin\beta/\sin(\gamma-\alpha) \geq 1/3$. In the second case, $\beta \in [.4, \pi/2]$. This means $\sin\beta \in [\sin(.4), 1]$ and in particular, $\sin\beta \geq .389$ so $\sin\beta/\sin(\gamma-\alpha) \geq .389$. The last case is $\beta > \pi/2$, which implies $\sin\beta = \sin(\pi-\beta) = \sin(\gamma+\alpha)$. So the quantity to analyze is $\sin(\gamma+\alpha)/\sin(\gamma-\alpha)$. Since the angles in the numerator and denominator are both less than $\pi/2$ (because $\beta > \pi/2$) and since $\gamma+\alpha > \gamma-\alpha$, we conclude that $\sin\beta/\sin(\gamma-\alpha) \geq 1$.

Thus, in all cases, we conclude that $|dc|/|df| \geq 1/3$. Combining this with (7.5) implies

$$|dc|/|ab| \geq .025. \tag{7.6}$$

Next, observe that $|ef|/|df| = \sin(\gamma-\alpha)/\sin\gamma$ by the law of sines applied to $\triangle def$. Again, we take three cases. If $\gamma - \alpha < .4$ (and hence $\gamma - \alpha \geq \alpha$; i.e., $\gamma \geq 2\alpha$, and $\alpha \leq \gamma/2$), then $2\alpha - \alpha < .4$; i.e., $\alpha \leq .4$, so $\gamma \leq \alpha + .4 \leq .8$. Since all these angles are in $[0, \pi/2]$, $\sin(\gamma-\alpha) \geq \sin(\gamma-\gamma/2) = \sin(\gamma/2)$ by monotonicity of sin in this range, and $\sin(\gamma/2)/\sin(\gamma) \geq 1/2$ by concavity of sin. The second case is $\gamma - \alpha \in [.4, \pi/2]$. In this case, $\sin(\gamma-\alpha) \geq .389$ so $\sin(\gamma-\alpha)/\sin\gamma \geq .389$. The last case is $\gamma - \alpha \geq \pi/2$. This implies that $\sin(\gamma-\alpha) = \sin(\pi-\gamma+\alpha) = \sin(2\alpha+\beta)$. The denominator becomes $\sin(\gamma) = \sin(\pi-\gamma) = \sin(\alpha+\beta)$. So we are analyzing $\sin(2\alpha+\beta)/\sin(\alpha+\beta)$, which exceeds 1 since all the angles in question are in $[0, \pi/2]$. Thus, in all cases, $|ef|/|df| \geq .389$; thus, combining with (7.5) yields

$$|ef|/|ab| \geq .0292. \tag{7.7}$$

Now we have enough inequalities to analyze $\text{minalt}(T_{\text{I}})$ where T_{I} denotes $\triangle ade$. Observe that this triangle is similar to T. Its longest side is either ae or ad (but not de, since $\alpha < \gamma$). If its longest side is ad, then we see that $|ad| = |ac| - |dc| \leq |ac| - .025|ab| \leq |ab| - .025|ab| = .9725|ab|$. (Here we use (7.6) plus the fact that $|ab| \geq |ac|$, which follows from the hypothesis of this case that $|ad| \geq |ae|$ together with the similarity of $\triangle ade$ to $\triangle abc$.) Thus, $\triangle ade$ is similar to $\triangle abc$ and is scaled down by a factor of .9725 or less.

The other case is when the longest side of $\triangle ade$ is ae. In this case, we can apply (7.7) to obtain $|ae| = |ab| - |eb| \leq |ab| - |ef| \leq (1 - .0292)|ab| \leq (1 - .0292)|ac|$. The inequality $|ab| \leq |ac|$ follows from the assumption that $|ad| \leq |ae|$ and from similarity. Thus, $\triangle ade$ is similar to $\triangle abc$ but is scaled down by a factor of less than .9708. This concludes the analysis of $\text{minalt}(T_{\text{I}})$. This same analysis applies to T_{III}, since T_{III} is congruent to T_{I} (because $\triangle adf$ is isosceles).

Next, observe that $|ae|/|ad| = \sin\beta/\sin\gamma$ by the law of sines applied to $\triangle ade$. Again, we take three cases. If $\beta < .4$ (and hence $\beta \geq \alpha$), then $\gamma > \pi/2$ as above so $\sin\gamma = \sin(\pi-\gamma) = \sin(\alpha+\beta)$; thus the quantity to analyze is $\sin\beta/\sin(\alpha+\beta)$. Using analysis as before, including steps like $\sin(\alpha+\beta) \leq \sin(2\beta) = 2\sin\beta\cos\beta \leq 2\sin\beta$, we conclude that $\sin\beta/\sin\gamma \geq 1/2$ in this case. If $\beta \in [.4, \pi/2]$, then we conclude again that $\sin\beta/\sin\gamma \geq .389$. Finally, if $\beta > \pi/2$, then $\sin\beta = \sin(\pi-\beta) = \sin(\gamma+\alpha)$, so again $\sin\beta/\sin\gamma > 1$. Thus, in all cases, $|ae|/|ad| \geq .389$. Since $|ad|/|ab| \geq .075$, we conclude that

$$|ae|/|ab| \geq .029. \tag{7.8}$$

Next, we analyze T_{II}, that is, $\triangle def$. Observe that this triangle is similar to $\triangle cfa$, and $\text{minalt}(\triangle cfa) \leq \text{minalt}(\triangle abc)$ by Corollary 7.2. The corresponding side to af is

fe. Applying (7.8) yields $|fe| = |af| - |ae| \le |af| - .029|ab| \le |af| - .029|af|$. Thus, $\text{minalt}(T_{\text{II}}) \le 0.971\,\text{minalt}(T)$.

Finally, we analyze T_{IV}, which is $\triangle cgd$. This triangle is also similar to $\triangle cfa$. Combining (7.4) with the fact that $|ab| \ge 0.5|ac|$ (since $|ab|$ is not the shortest side of $\triangle abc$ because $\gamma > \alpha$) yields $|ad| \ge .037|ac|$; thus $|cd| \le .963|ac|$. Observe that $|cd|/|ac|$ is the ratio of similarity between T_{IV} and $\triangle cfa$. ∎

The following lemma is like the previous one except with an inequality in the opposite direction.

LEMMA 7.5. [illegible] 4.1 [illegible] T [illegible] $\beta \ge \min(.4, \alpha)$ [illegible] $\gamma - \alpha \ge \min(.4, \alpha)$ [illegible] T' [illegible] $\text{minalt}(T') \ge \sigma\,\text{minalt}(T)$ [illegible] $\sigma = 0.0044$ [illegible] $\sigma = \sin\alpha$ [illegible] 3. The factor 0.0044 is due to our proof technique and appears to be an underestimate. A search over a fairly dense grid of possible angles satisfying the hypotheses of the theorem indicates that the true bound is closer to 0.125.

[illegible]. Again, we consider the five subtriangles and reuse some of the inequalities in the preceding proof. Observe that T_{I} is similar to $\triangle abc$, with ad corresponding to ab. By (7.4), $\text{minalt}(\triangle ade) \ge .075\,\text{minalt}(\triangle abc)$. The same bound applies to T_{III}, which is congruent to T_{I}.

For T_{V}, by the law of sines $|bc|/|ab| = \sin\alpha/\sin\gamma \ge \sin\alpha$. Thus, by similarity, $\text{minalt}(T_{\text{V}}) \ge \text{minalt}(T) \cdot \sin\alpha$.

Next, consider triangle $\triangle acf$. Equation (7.3) implies that if $\triangle acf$ were dilated by $1/.151$, it would completely cover $\triangle abc$. Therefore, by Corollary 7.2, $\text{minalt}(\triangle acf) \ge .151\,\text{minalt}(\triangle abc)$. Equation (7.7) implies that $|ef|/|af| \ge .0292$. Since T_{II} is similar to $\triangle acf$, we conclude that $\text{minalt}(T_{\text{II}}) \ge .0292\,\text{minalt}(\triangle acf) \ge .0292 \cdot .151\,\text{minalt}(\triangle abc)$. Note that $.0292 \cdot 0.151 \ge 0.0044$.

Finally, to analyze T_{IV}, we need to develop a new inequality. Observe that $\triangle dgf$ is similar to $\triangle acb$ and the scaling is at least .075 by (7.5); hence $|dg| \ge .075|ac|$. Meanwhile, we know $|ac| \ge 0.5|af|$ since $|ac|$ is not the shortest side of $\triangle acf$ (because it is opposite an angle of size $\alpha + \beta$, which is greater than α). Thus, $|dg| \ge .037|af|$. This means by similarity of T_{IV} to $\triangle acf$ that $\text{minalt}(T_{\text{IV}}) \ge .037\,\text{minalt}(\triangle acf) \ge .037 \cdot .151\,\text{minalt}(\triangle abc)$. ∎

LEMMA 7.6. [illegible] σ τ [illegible] $\sigma < 1$ [illegible] $\tau < 1$ [illegible] k [illegible]

$$\prod_{i=0}^{\infty}(1 - \sigma\tau^i)^k \ge 1 - k\sigma/(1-\tau)$$

[illegible]

$$\prod_{i=0}^{\infty}(1 + \sigma\tau^i)^k \le \exp(k\sigma/(1-\tau)).$$

[illegible]. The first inequality follows because $(1-a)(1-b) \ge 1-a-b$ for $a, b \in [0,1]$. Applying this repeatedly, k times for each factor in the product, $(1-\sigma)^k(1-\sigma\tau)^k \cdots (1-\sigma\tau^n)^k \ge 1 - k\sigma - k\sigma\tau - \cdots - k\sigma\tau^n \ge 1 - k\sigma\sum_{i=0}^{\infty}\tau^i = 1 - k\sigma/(1-\tau)$.

The second inequality follows by taking logs and using the inequality $\log(1+x) \le x$:

$$\log((1+\sigma)^k \cdots (1+\sigma\tau^n)^k) = k\log(1+\sigma) + k\log(1+\sigma\tau) + \cdots + k\log(1+\sigma\tau^n)$$

$$\begin{aligned}&\le k\sigma + k\sigma\tau + \cdots + k\sigma\tau^n\\ &\le k\sigma\sum_{i=0}^{\infty}\tau^i\\ &= k\sigma/(1-\tau). \qquad \square\end{aligned}$$

We now explain how to choose δ for the main algorithm. We set it to be

$$\delta = \frac{\min\{\text{minalt}(T) : T \in \mathcal{T}_*\}}{1460}. \tag{7.9}$$

The minimum altitudes in this definition are measured before any collapse-node operations begin. This choice of δ makes all the theorems work but leads to poorer aspect ratio (by a constant factor) than seems necessary. So instead, PINW 1.0 chooses δ dynamically based on the singular values of the affine transformations that could be applied during collapse-node operations. This heuristic seems to work well in practice.

The following theorem bounds the effect of all collapse-node operations, both direct and indirect.

THEOREM 7.7. [illegible] T [illegible] A [illegible] T [illegible] $\mu = \text{minalt}(T)$ [illegible] L [illegible] T [illegible] δ [illegible] (7.9) [illegible] $\text{length}(A(L))/\text{length}(L)$ [illegible]

$$\left(1-\frac{\delta}{0.75\mu}\right)^3\cdot\left(1-\frac{0.9725\delta}{0.75\mu}\right)^3\cdot\left(1-\frac{0.9725^2\delta}{0.75\mu}\right)^3\cdots$$

[illegible]

$$\left(1+\frac{\delta}{0.75\mu}\right)^3\cdot\left(1+\frac{0.9725\delta}{0.75\mu}\right)^3\cdot\left(1+\frac{0.9725^2\delta}{0.75\mu}\right)^3\cdots.$$

[illegible]. The proof of this theorem is by induction on the depth of the tile in hierarchy. The induction base is that for a root tile, there are no collapse-node operations; thus $A(L) = L$.

For a nonroot tile T, let T' be its parent triangle. By the induction hypothesis, the total distortion of a segment in T' prior to the processing of the vertices created within T' is between

$$\left(1-\frac{\delta}{0.75\mu'}\right)^3\cdot\left(1-\frac{0.9725\delta}{0.75\mu'}\right)^3\cdot\left(1-\frac{0.9725^2\delta}{0.75\mu'}\right)^3\cdots$$

and

$$\left(1+\frac{\delta}{0.75\mu'}\right)^3\cdot\left(1+\frac{0.9725\delta}{0.75\mu'}\right)^3\cdot\left(1+\frac{0.9725^2\delta}{0.75\mu'}\right)^3\cdots,$$

where $\mu' = \text{minalt}(T')$ (with minalt measured prior to any node movement). Referring to Figure 4.1 and regarding $T' = abc$ and T as one of I, II, III, IV, or V, we consider next the direct displacements of d, e, f due to collapse-node operations.

By Lemma 7.4, $0.9725\mu' \ge \mu$. Thus, for tile T, the distortion prior to the three direct displacements of d, e, f is bounded between

$$\left(1-\frac{0.9725\delta}{0.75\mu}\right)^3\cdot\left(1-\frac{0.9725^2\delta}{0.75\mu}\right)^3\cdot\left(1-\frac{0.9725^3\delta}{0.75\mu}\right)^3\cdots \tag{7.10}$$

and

$$\left(1+\frac{0.9725\delta}{0.75\mu}\right)^3\cdot\left(1+\frac{0.9725^2\delta}{0.75\mu}\right)^3\cdot\left(1+\frac{0.9725^3\delta}{0.75\mu}\right)^3\cdots. \tag{7.11}$$

By Lemma 7.6 with $k=3$, $s=\delta/(0.75\mu)$, and $t=0.9725$, infinite product (7.10) is greater than or equal to $1-3\delta/(0.75\mu(1-0.9725))$, which simplifies to $1-146\delta/\mu$. If we assume that δ satisfies (7.9), then this quantity is greater than 0.9. We now apply the three collapse-node operations of d, e, f to T, say in this order. (Not all three necessarily affect T; for example, if T is I in Figure 4.1, then moving f does not affect T.) To compute the distortion of L requires knowledge of the minimum altitude of T at the point of the algorithm when the collapse-node operation is applied. However, because the distortion so far is greater than 0.9, we know that the altitude at this step is at least 0.9μ for movement of d, which is of size δ. Therefore, the movement of d applies a new distortion between $1-\delta/(0.9\mu)$ and $1+\delta/(0.9\mu)$ by Lemma 7.3. By (7.9), this quantity is bounded between 0.95 and 1.05. Therefore, the minimum altitude of T when the collapse-node operation for e is applied is at least $0.9\mu\cdot 0.95=0.855\mu$. Thus, the collapse-node operation on e applies another distortion between $1-\delta/(0.855\mu)$ and $1+\delta/(0.855\mu)$. Again, by (7.9), this quantity is bounded between 0.95 and 1.05. So after the collapse-node operation on f, the minimum altitude of T is at least $0.855\mu\cdot 0.95=0.812\mu$. Combining these three distortions with the distortions from higher-level collapse-node operations given by (7.10) and (7.11) shows that the value of the distortion of a segment after the collapse-node operations involving T and its ancestors lies between

$$\begin{aligned}&(1-\delta/(0.9\mu))(1-\delta/(0.855\mu))(1-\delta/(0.812\mu))\\&\cdot\left(1-\frac{0.9725\delta}{0.75\mu}\right)^3\cdot\left(1-\frac{0.9725^2\delta}{0.75\mu}\right)^3\cdot\left(1-\frac{0.9725^3\delta}{0.75\mu}\right)^3\cdots\end{aligned}$$

and

$$\begin{aligned}&(1+\delta/(0.9\mu))(1+\delta/(0.855\mu))(1+\delta/(.812\mu))\\&\cdot\left(1+\frac{0.9725\delta}{0.75\mu}\right)^3\cdot\left(1+\frac{0.9725^2\delta}{0.75\mu}\right)^3\cdot\left(1+\frac{0.9725^3\delta}{0.75\mu}\right)^3\cdots.\end{aligned}$$

We can underestimate the first factor and overestimate the second by replacing 0.9, 0.855, and 0.812 all with 0.75. This proves the theorem. ∎

We now consider Properties 1 and 2 in section 6. Since the minimum altitude of a triangle is less than or equal to its shortest side length, and since the minimum altitude of any tile decreases by at most 0.75, the previous result shows that δ is sufficiently small so that no two nodes can be collapsed to the same node, and no node can have more than one choice of where it should be collapsed.

Furthermore, when we are finished with collapse-node operations, all hanging nodes are at least δ apart and at least δ from corners. Again, this is because the shortest side length is bounded below by the smallest altitude, and the smallest altitude is bounded below by a large constant multiple of δ.

We now consider the aspect ratio of the triangles in the mesh produced by PINW. As mentioned earlier, we define the [illegible] of a triangle T to be the square of the longest side length of the triangle divided by its area. Let maxside(T), area(T), and aspect(T) stand for the maximum side, area, and aspect ratio, respectively. We have

also already introduced minalt(T). Then the previous definition says that

$$\text{aspect}(T) = \text{maxside}(T)^2 / \text{area}(T).$$

Since the area is half the product of the longest side length and the minimum altitude, an equivalent definition is

$$\text{aspect}(T) = 2\,\text{maxside}(T) / \text{minalt}(T).$$

The following lemma gives another characterization of aspect ratio equivalent up to a constant factor as well as a useful property of aspect ratios.

LEMMA 7.8. [illegible] T [illegible]

(a) [illegible] θ [illegible] T [illegible] κ_1, κ_2 [illegible] $\kappa_1\,\text{aspect}(T) \le 1/\theta \le \kappa_2\,\text{aspect}(T)$ [illegible] $\kappa_1\,\text{aspect}(T) \le 1/\sin(\theta) \le \kappa_2\,\text{aspect}(T)$

(b) [illegible] λ_1, λ_2 [illegible] T [illegible] $\lambda_1/\lambda_2 \le \text{aspect}(T)/2$

[illegible]. Because both parts of this lemma are well known (see, e.g., [4]), we omit the full proof. □

We now establish via a sequence of lemmas followed by a theorem that the triangles produced by PINW have bounded aspect ratio, where the bound depends only on the sharpest angle in the input domain. For the remainder of the section, let

$$\rho^* = \max\{\text{aspect}(T) : T \in \mathcal{T}_0 \text{ or conjugate}(T) \in \mathcal{T}_0\}. \tag{7.12}$$

The first lemma relates ρ^* to the original input domain.

LEMMA 7.9. [illegible] θ [illegible] Ω [illegible] $\rho^* \le \kappa/\theta$ [illegible] κ [illegible]

[illegible]. For root tiles, this follows from the properties of the Triangle mesh generator: It is guaranteed that all triangles will have aspect ratio bounded by κ/θ. The preliminary splitting for angles too close to equilateral can decrease the sharpest angle by at most a constant factor, For a conjugate T' of a root tile T whose angles are α, β, γ in order, either its sharpest angle is α, in which case they satisfy the same bound (because α is also an angle of a root tile; Lemma 7.8(a) applies). The other case is that the sharpest angle in the conjugate is $\gamma - \alpha$, but because of the primary tripartition, this angle is no smaller than .4. □

The following lemma follows trivially from the fact that in the recursive application of ATP, all subsequent tiles are congruent either to a root tile or to its conjugate.

LEMMA 7.10. [illegible] $T \in \mathcal{T}_*$ [illegible] $\text{aspect}(T) \le \rho^*$

The next lemma shows that a leaf tile cannot be much larger than any other leaf tile.

LEMMA 7.11. [illegible] $\mathcal{T}_f$ [illegible] $\mathcal{T}_0$ [illegible]

$$\max\{\text{minalt}(T) : T \in \mathcal{T}_f\} \le \kappa\rho^* \min\{\text{minalt}(T) : T \in \mathcal{T}_f\},$$

[illegible] κ [illegible]

[illegible]. Recall that the tile selected for splitting at any given step is the one with the maximum minimum altitude. Thus, when the subdivision procedure terminates, the tile at the top of the heap will be the leaf tile with the maximum minimum altitude among all leaf tiles. Say this tile is T and its minimum altitude is μ. Now consider any

other leaf tile $T' \in \mathcal{T}_f$. Because of the assumption that no tile from $\mathcal{T}_0$ is a leaf tile, this tile T' must have arisen from ATP subdivision of some other tile T''. Because of the heap order, the minimum altitude μ'' of T'' exceeds μ. By Lemma 7.5, this means that the minimum altitude μ' of T' is at least $\min(0.0044, \sin\alpha)\mu''$. Note that $\sin\alpha \geq \kappa/\rho^*$ since α is an angle of a root tile or conjugate by Lemma 7.10. Thus, $\mu' \geq (\kappa/\rho^*)\mu'' \geq (\kappa/\rho^*)\mu$. □

Now, we consider the effect of collapse-node operations.

LEMMA 7.12. 1.22

. As explained in the proof of Theorem 7.7, the lowest possible contraction due to all collapse-node operations for any segment in any leaf tile is 0.90 or greater. Pick a tile T and let μ be the initial altitude. Applying the second part of Lemma 7.6 to the bound in the theorem shows that the maximum distortion of T is $\exp(3\delta/(0.9\mu(1-0.9725)))$, which by (7.9) is at most 1.09. Since the aspect ratio is the twice the longest side length divided by the minimum altitude, and the longest side went up by at most 1.09 while the minimum altitude changed by a factor of at least 0.90, the new aspect ratio is bounded by 1.22 times the old. □

We now come to the main result for this section about the aspect ratio of the triangles generated by PINW.

THEOREM 7.13. $\mathcal{T}_f$ $\kappa(\rho^*)^3$ κ ρ^* (7.12)

. In this proof, κ denotes a universal constant that takes on a different value for each inequality where it appears. Let $\bar{T}$ be a leaf tile. Let $\mu = \text{minalt}(\bar{T})$ and $\lambda = \text{maxside}(\bar{T})$, both measured prior to any collapse-node operations. Let T be the result of PINW's collapse-node operations applied to $\bar{T}$. As already observed in the proof of Theorem 7.7, $\text{minalt}(T) \geq 0.9\mu$ and $\text{maxside}(T) \leq 1.09\lambda$. The edges of T may contain hanging nodes. The distance between any pair of adjacent hanging nodes or between a hanging node and corner is at least δ. A node on the moving side could not be less than δ away from a node on the staying side, since in this case they would be collapsed. Also, two nodes on the staying side or two nodes on the moving side are at least distance 1460δ apart by (7.9), so even after collapse-node operations no leaf tile edge could ever shrink below δ.

Recall from (7.9) that δ is a constant multiple of $\mu_{\min}$, the minimum altitude among all leaf tiles. By Lemma 7.11, this implies

$$\delta \geq \kappa\mu/\rho^*, \tag{7.13}$$

where κ is a universal constant.

Now, let U be a triangle output by the Delaunay triangulation of T and consider the sharpest angle of U. Let $E = v_1v_2$ be the edge of U opposite the sharpest angle. There are two cases: Either v_1, v_2 lie on the same edge of T (i.e., they are consecutive hanging nodes or a hanging node and a corner node), or they are on different edges.

Start with first case. Let $E = v_1v_2$ be the edge of U lying on an edge of T. For the rest of this case, let the corners of T be a, b, c such that $v_1v_2 \subset bc$ and such that the order of these vertices is b, v_1, v_2, c. Let p be the vertex of U opposite E. Refer to Figure 7.1(a). By definition of the Delaunay triangulation, p is the first vertex hit by an expanding circle that contains both endpoints of E. This circle, if expanded farther, will eventually hit a. Either $\angle av_1v_2$ is acute or $\angle av_2v_1$ is acute; without loss

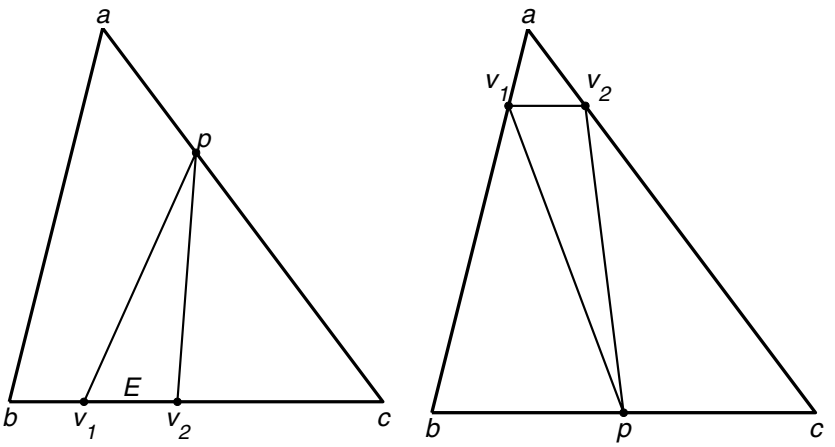


FIG. 7.1. *Constructions used in the proof of Theorem* 7.13. *Triangle v_1v_2p results from the Delaunay triangulation of tile T whose vertices are a, b, c and whose hanging nodes include v_1, v_2, p. Nodes are numbered so that v_1v_2 is the shortest side of $\triangle v_1v_2p$. Either* (a) *edge $E = v_1v_2$ lies on a single edge of T, or* (b) *v_1v_2 is interior to T.*

of generality, assume the former. The angle $\angle v_1av_2$ is at least $\kappa|E|/(\rho^*\lambda)$ by the law of sines: $\sin\angle v_1av_2 = \sin\angle av_1v_2|E|/|av_2| \geq \sin\angle av_1v_2|E|/(1.09\lambda)$. Meanwhile, $\sin\angle av_1v_2$ is bounded below by κ/ρ^* since $\angle av_1v_2$ is bounded below by $\angle abc$ but is less than $\pi/2$. Thus, $\sin\angle v_1av_2 \geq \kappa|E|/(\rho^*\lambda)$. Next, $\lambda \leq \mu\rho^*$ so $\sin\angle v_1av_2 \geq \kappa|E|/(\mu(\rho^*)^2)$. Finally, since $|E| \geq \delta$ because all hanging nodes are at least δ apart, it follows from (7.13) that $|E|/\mu \geq \kappa/\rho^*$. Therefore, $\sin\angle v_1av_2 \geq \kappa/(\rho^*)^3$. This is the angle formed by v_1av_2. The actual angle of U opposite E is v_1pv_2. But this angle is greater than or equal to v_1av_2, since the expanding Delaunay circle encounters p before it encounters a (or else $p = a$).

Next, let us consider the case that v_1 and v_2, the endpoints of the edge of U opposite its sharpest angle, do not lie on the same edge of T. Let the three vertices of T be a, b, c and let a be the T-vertex that is the common endpoint of the two T edges that contain v_1 and v_2 respectively, while we let b be the T-vertex such that ab contains v_1 and we let c be the T-vertex such that ac contains v_2. Refer to Figure 7.1(b). Consider $\angle v_1bv_2$. Since a, v_1, and b are collinear, this angle is equal to $\angle abv_2$. By the law of sines applied to $\triangle abv_2$, we have $\sin\angle abv_2 = |av_2|\sin\angle a/|bv_2|$. Now we apply the inequalities $|av_2| \geq \delta$, $\sin\angle a \geq \kappa/\rho^*$, and $|bv_2| \leq \lambda$ to conclude that $\sin\angle abv_2 = \sin\angle v_1bv_2 \geq \kappa\delta/(\rho^*\lambda)$. This is the same inequality derived in the previous case and yields the conclusion that $\sin\angle v_1bv_2 \geq \kappa/(\rho^*)^3$. Arguing again as in the previous case, the actual Delaunay triangle containing v_1v_2 may not have b as its third vertex, but if it has another vertex p, then $\angle v_1pv_2$ is greater than $\angle v_1bv_2$ by considering the expanding circle property. □

8. Isoperimetry of the final mesh. We have already proved in Theorem 5.2 that the tiling of a triangle by our generalized pinwheel subdivision has the isoperimetric property. It is straightforward to extend this result to the collection of all leaf tiles.

THEOREM 8.1. $\mathcal{T}_0, \mathcal{T}_1, \ldots$ Ω n $\mathcal{T}_n$ Ω δ_n $\delta_n \to 0$ $n \to \infty$ p, q $P, Q \in \text{Skel}(\mathcal{T}_k)$ k

$$\lim_{\substack{n\to\infty \\ n\geq k}} \operatorname{dist}_{\text{Skel}(\mathcal{T}_n)}(p,q) = \|p-q\|_\Omega.$$

. This theorem follows from Theorem 5.5 and uses the same proof technique. Let Π be the geodesic path from p to q of length $\|p-q\|_\Omega$. Since Ω is a polygon, Π is composed of a finite number of line segments. For each tile T_i in $\mathcal{T}_k$ that meets Π, consider the small segment p_iq_i that is $T_i \cap \Pi$. Then we use Theorem 5.2 to argue that this small segment p_iq_i can be approximated arbitrarily accurately. ∎

This theorem can now be extended to the final mesh output by PINW by analyzing the effect of collapse-node operations on the isoperimetric number. (The Delaunay operations do not disturb the isoperimetry result, since adding edges could only make the isoperimetric number decrease.)

The definition of isoperimetry implicit in Theorems 5.5 and 8.1 is not suitable for analyzing the output of PINW because the meshes produced by PINW are not refinements of their predecessors as the mesh size decreases. This is because the collapse-node operations move nodes differently depending on the size of the leaf tiles.

Therefore, we use the following definition. An infinite sequence of simplicial meshes $\mathcal{M}_1, \mathcal{M}_2, \ldots$ for a domain Ω has the if for each $\mathcal{M}_i$ there is a subset L_i of its vertices such that the following two properties hold. First, L_i is asymptotically dense in Ω as $i \to \infty$; i.e., for any $\epsilon > 0$, there is an i^* such that for any $x \in \Omega$ and any $i > i^*$, there exists a $v \in L_i$ such that $\|x - v\| \leq \epsilon$. Second,

$$\limsup_{i\to\infty} \left\{ \frac{\operatorname{dist}_{\text{Skel}(\mathcal{M}_i)}(x,y)}{\|x-y\|_\Omega} : x, y \in L_i; x \neq y \right\} = 1.$$

THEOREM 8.2.

. To show that PINW has this property, take a sequence of ϵ_i's tending to zero. For each i, let $\mathcal{T}_i$ be a generalized pinwheel subdivision of Ω such that each leaf cell has diameter less than $\epsilon_i/2$. Then let $\mathcal{T}_i'$ be a further subdivision of $\mathcal{T}_i$ such that for any two distinct vertices of $\mathcal{T}_i$, $\operatorname{dist}_{\text{Skel}(\mathcal{T}_i')}(x,y) \leq (1+\epsilon_i/4)\|x-y\|_\Omega$. The existence of such a $\mathcal{T}_i'$ is established by Theorem 8.1. Let μ' be the minimum altitude among leaf tiles in $\mathcal{T}_i'$. Next, further refine $\mathcal{T}_i'$ to yield a tiling $\mathcal{T}_i''$ with the property that when δ is defined by (7.9) for $\mathcal{T}_i''$ (i.e., the $\mathcal{T}_*$ appearing in (7.9) pertains to $\mathcal{T}_i''$), then this δ is sufficiently small so that

$$\exp(3\delta/(0.75\mu'(1-0.9725))) \leq 1+\epsilon_i/4 \tag{8.1}$$

and

$$1-\epsilon_i/4 \leq 1-3\delta/(0.75\mu'(1-0.9725)). \tag{8.2}$$

Now finally, take $\mathcal{M}_i$ to be the simplicial mesh output by PINW based on $\mathcal{T}_i''$, and take L_i to be the set of nodes of $\mathcal{M}_i$ that are displaced copies of the nodes of $\mathcal{T}_i$.

First, we have to show that L_i defined in this manner is asymptotically dense. The nodes of L_i are the same as the nodes of $\mathcal{T}_i$ after small displacements. Since every

cell of $\mathcal{T}_i$ has diameter less than $\epsilon_i/2$, this means that any point $x \in \Omega$ is distance at most $\epsilon_i/2$ from a vertex of $\mathcal{T}_i$. The vertices of L_i are slightly displaced, but no distance λ decreases below 0.9λ or increases to more than 1.09λ. Therefore, for any $x \in \Omega$ the perturbed set L_i contains a point v within distance $1.09 \cdot \epsilon_i/2 < \epsilon_i$ of x.

Let x, y be two distinct points in L_i. The next task is to show that $\operatorname{dist}_{\mathrm{Skel}(\mathcal{M}_i)}(x, y) \leq \|x-y\|_\Omega(1+\epsilon_i)$. Let x_0, y_0 be the positions of x, y in $\mathcal{T}_i''$ prior to all distortions caused by collapse-node operations. Note that x_0, y_0 are vertices of $\mathcal{T}_i'$ and also of $\mathcal{T}_i$ by construction. Therefore, by construction of $\mathcal{T}_i'$, there is a path P_0 in $\mathrm{Skel}(\mathcal{T}_i')$ connecting x_0 and y_0 such that $\operatorname{length}(P_0) \leq \|x_0 - y_0\|_\Omega \cdot (1 + \epsilon_i/4)$. Let the segments of P_0 be $E_1, E_2, \ldots, E_m$. Let the image of P_0 after all the collapse-node operations with their attendant distortions are applied be P, and let the images of $E_1, \ldots, E_m$ be $F_1, \ldots, F_m$. Recall that the distortions that affect a node v of a tile T in the hierarchy are those distortions associated with T and its ancestor tiles, but descendant tiles cannot move T. Therefore, by Theorem 7.7, all of the quantities $\|x - y\|/\|x_0 - y_0\|_\Omega$, $\operatorname{length}(F_i)/\operatorname{length}(E_i)$, and $\operatorname{length}(P)/\operatorname{length}(P_0)$ lie between

$$\left(1 - \frac{\delta}{0.75\mu'}\right)^3 \cdot \left(1 - \frac{0.9725\delta}{0.75\mu'}\right)^3 \cdot \left(1 - \frac{0.9725^2\delta}{0.75\mu'}\right)^3 \cdots$$

and

$$\left(1 + \frac{\delta}{0.75\mu'}\right)^3 \cdot \left(1 + \frac{0.9725\delta}{0.75\mu'}\right)^3 \cdot \left(1 + \frac{0.9725^2\delta}{0.75\mu'}\right)^3 \cdots,$$

where the δ in this formula is given by (7.9) associated with $\mathcal{T}_i''$. By Lemma 7.6, this interval is bracketed by

$$1 - 3\frac{\delta}{0.75\mu'(1 - 0.9725)}$$

and

$$\exp(3\delta/(0.75\mu'(1 - 0.9725))).$$

Then by (8.1) and (8.2), this interval is bracketed by $[1 - \epsilon_i/4, 1 + \epsilon_i/4]$. Thus,

$$\begin{aligned}\operatorname*{dist}_{\mathrm{Skel}(\mathcal{M}_i)}(x, y) &\leq (1 + \epsilon_i/4) \operatorname*{dist}_{\mathrm{Skel}(\mathcal{T}_i')}(x_0, y_0) \\ &\leq (1 + \epsilon_i/4)^2 \|x_0 - y_0\|_\Omega \\ &\leq (1 + \epsilon_i/4)^2 \|x - y\|_\Omega/(1 - \epsilon_i/4).\end{aligned}$$

Note that $(1 + \epsilon_i/4)^2/(1 - \epsilon_i/4) \leq 1 + \epsilon_i$ as long as $\epsilon_i \leq 1/2$. Thus, we have shown that for all $x, y \in L_i$, $\operatorname{dist}_{\mathrm{Skel}(\mathcal{M}_i)}(x, y) \leq (1 + \epsilon_i)\|x - y\|_\Omega$. □

9. Conclusions. We believe that this generalization of pinwheel tiling to meshing polygonal regions would aid in modeling arbitrary crack paths more accurately than the current meshing techniques. Also, this kind of substitutive mechanism for subdivision makes it easy for adaptive meshing. This work raises a number of interesting directions for future research. Among them are the following:

1. The transformation of the tiling to the mesh had the effect of increasing the aspect ratio significantly. Is there a better way to carry out this transformation to reduce the impact on aspect ratio?

2. The convergence rate of the isoperimetric number of the pinwheel tiling to 1, which was not analyzed here, is known to be extremely slow even in the case of the 1:2 tiling. Is there another approach to isoperimetry that converges faster?
3. Consider a mesh generated by placing random points in the domain under consideration and joining them with a Delaunay triangulation. Is there a limiting isoperimetric number for this family of meshes (with high probability)?
4. Another way to construct a mesh of an arbitrary polygon with limiting isoperimetric number equal to 1 is to use the 1:2 pinwheel subdivision for every coarse triangle after subjecting it to a (potentially large) affine transformation. This approach is simpler in certain respects than PINW. For example, the collapse-node operations for this algorithm need to be done only at the boundaries of the coarse triangles. The difficulty with this approach is that it spoils the "statistical rotational invariance" of the pinwheel tiling. The statistical rotational invariance property states that the set of possible directions is covered at a uniform rate as subdivision proceeds. We are unclear whether statistical rotational invariance is important for cohesive interface modeling. We suspect that our construction of generalized pinwheels has statistical rotational invariance but have no proof of this. Is a simplification of our algorithm such as this one possible?
5. Can any of this work be extended to three dimensions?

Acknowledgments. We are grateful to Marshall Bern for originally telling us about Radin's paper. We also thank the two reviewers of the short version of this paper submitted to the 2004 International Meshing Roundtable for their helpful comments. We thank the two referees of the first draft of the journal version of this paper for additional helpful comments.

REFERENCES

[1] A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *Data Structures and Algorithms*, Addison-Wesley, Reading, MA, 1983.
[2] S. C. Brenner and L. R. Scott, *The Mathematical Theory of Finite Element Methods*, Springer, New York, 1994.
[3] G. Golub and C. V. Loan, *Matrix Computations*, 3rd ed., The Johns Hopkins University Press, Baltimore, MD, 1996.
[4] P. M. Knupp, *Algebraic mesh quality metrics*, SIAM J. Sci. Comput., 23 (2001), pp. 193–218.
[5] K. D. Papoulia, C.-H. Sam, and S. Vavasis, *Time continuity in cohesive finite element modeling*, Internat. J. Numer. Methods Engrg., 58 (2003), pp. 679–701.
[6] C. Radin, *The pinwheel tilings of the plane*, Ann. of Math. (2), 139 (1994), pp. 661–702.
[7] C. Radin and L. Sadun, *The isoperimetric problem for pinwheel tilings*, Comm. Math. Phys., 177 (1996), pp. 255–263.
[8] L. Sadun, *Some generalizations of the pinwheel tilings*, Discrete Comput. Geom., 20 (1998), pp. 79–110.
[9] J. R. Shewchuk, *Triangle: Engineering a 2D quality mesh generator and Delaunay triangulator*, in Applied Computational Geometry: Towards Geometric Engineering, M. C. Lin and D. Manocha, eds., Lecture Notes in Comput. Sci. 1148, Springer, Berlin, 1996, pp. 203–222.
[10] X.-P. Xu and A. Needleman, *Numerical simulations of fast crack growth in brittle solids*, J. Mech. Phys. Solids, 42 (1994), pp. 1397–1434.

SIAM J. SCI. COMPUT.
Vol. 28, No. 4, pp. 1563–1581
© 2006 Society for Industrial and Applied Mathematics

A NEW KIND OF ACCURATE NUMERICAL METHOD FOR BACKWARD STOCHASTIC DIFFERENTIAL EQUATIONS*

WEIDONG ZHAO†, LIFENG CHEN†, AND SHIGE PENG†

Abstract. In this paper, we propose a new kind of numerical simulation method for backward stochastic differential equations (BSDEs). We discretize the continuous BSDEs on time-space discrete grids, use the Monte Carlo method to approximate mathematical expectations, and use space interpolations to compute values at nongrid points. To demonstrate the accuracy and the effectiveness of our method, several numerical examples are given.

Key words. backward stochastic differential equations, Monte Carlo method, time-space discretization

AMS subject classifications. 60H35, 65C05

DOI. 10.1137/05063341X

1. Introduction.

In this paper, we consider the numerical solutions for the backward stochastic differential equations (BSDEs)

$$\begin{aligned} -dy_t &= f(t, y_t, z_t)dt - z_t \mathrm{d}W_t, \quad t \in [0, T), \\ y_T &= \xi, \end{aligned} \tag{1.1}$$

where $W_t = (W_t^1, \ldots, W_t^d)^*$ is a d-dimensional Brownian motion defined on some complete, filtered probability space $(\Omega, \mathcal{F}, P, \{\mathcal{F}_t\}_{0 \le t \le T})$, and $\xi \in \mathcal{F}_T$, and the notation * is the transpose operator for matrix or vector.

In 1990, Pardoux and Peng in [26] obtained the unique solvability results for the nonlinear BSDEs under some standard conditions. Since then, the BSDEs have been extensively studied [28, 29, 30, 18, 1, 11, 33, 24, 25] by many researchers. In [18], the properties of the solutions of the BSDEs and their important applications in mathematical finance were discussed. Some fine properties of the BSDEs by using Malliavin derivatives [14, 15] under weaker conditions were studied in the Ph.D. thesis [33] of Zhang. The applications of the BSDEs now cover many scientific fields, such as stochastic control, turbulence fluid flow, biology, stock markets, radiation, chemical reactions, and PDEs. However, it is often difficult to solve the BSDEs analytically, even the linear BSDEs.

Many efforts have been made in the numerical methods for BSDEs [2, 3, 4, 6, 9, 12, 22, 23, 24, 25, 30, 33, 34, 35]. By using the relations between the BSDEs and the PDEs [28, 17], a four step scheme for solving the BSDEs was developed in [23, 24]. Based on the four step scheme, a characteristic difference method was proposed to solve forward-backward SDEs (FBSDEs) in [12]. These two methods depend on the numerical schemes for PDEs and for the related ordinary SDEs. In [3], Bally and Pagès studied an optimal discrete quantization method designed to compute the solution of a certain kind of reflected BSDE, and the error analysis was given in [4]. In his Ph.D. thesis [9], Chevance proposed a numerical method for BSDEs by using binomial approach to approximate the process y_t. Peng proposed an iterative

*Received by the editors June 10, 2005; accepted for publication (in revised form) May 23, 2006; published electronically September 15, 2006.
http://www.siam.org/journals/sisc/28-4/63341.html
†School of Mathematics and System Sciences, Shandong University, Jinan, Shandong, People's Republic of China (wdzhao@math.sdu.edu.cn, sduclf@math.sdu.edu.cn, peng@sdu.edu.cn).

linearization procedure to solve BSDEs in [30]. In [6], the time discrete approximation and the Monte Carlo method for solving BSDEs were studied by Bounchard and Touzi. In [34], Zhang studied a numerical approximation to the solution of the BSDEs under some weaker conditions.

Compared with the development of the numerical methods for the ordinary stochastic differential equations (SDEs) [5, 19, 20, 21, 32], the numerical methods for solving the solution of the BSDEs are far undeveloped, and effective and efficient methods are urgently needed.

In this paper, we study the numerical methods for the BSDEs (1.1). Based on the properties of the BSDEs, we propose a new accurate numerical method for the BSDEs. We fully discretize the time-space continuous problems and use the Monte Carlo method to approximate the mathematical expectations in our method. The space interpolation is used when the values at nongrid points are needed. The unknowns on each time level are totally uncoupled among the space grids, and the used data can be local for each space grid. Thus our method can be used to solve the general BSDEs in high performance parallel computers. To demonstrate the efficiency of our method, we use our method to solve some model BSDEs, and the results show that our method is efficient and is of high order.

We organize this paper as follows. In section 2, some basic notation and definitions are given. In section 3, we outline our solution procedure for one-dimensional BSDEs. Then, we generalize it and propose the numerical scheme for the general BSDEs (1.1) in section 4. Numerical experiments for some BSDE models are given in section 5. Finally, we draw some conclusions in section 6.

2. Notation and definitions. Let $(\Omega, \mathcal{F}, P, \{\mathcal{F}_t\}_{0\le t\le T})$ be a complete, filtered probability space on which a standard d-dimensional Brownian motion $W_t (0 \le t \le T)$ is defined with a finite terminal time T. Let the filtration $\{\mathcal{F}_t\}_{0\le t\le T}$ be the σ-field generated by the Brownian motion W_t, i.e., $\mathcal{F}_t = \sigma\{W_s, 0 \le s \le t\}$. All the P-null sets are augmented to each σ-field $\mathcal{F}_t$. Denote $L^2 = L^2_{\mathcal{F}}(0, T; \mathbb{R}^d)$ to be the set of all $\mathcal{F}_t$-adapted and mean-square-integrable processes valued in $\mathbb{R}^d$.

A process $(y_t, z_t) : [0, T] \times \Omega \to \mathbb{R}^m \times \mathbb{R}^{m\times d}$ is called an L^2-adapted solution of the BSDEs (1.1) if it is $\{\mathcal{F}_t\}$-adapted and square integrable and satisfies (1.1) in the sense of

$$y_t = \xi + \int_t^T f(s, y_s, z_s)ds - \int_t^T z_s \mathrm{d}W_s, \quad t \in [0, T), \tag{2.1}$$

where $f(t, y, z) : [0, T] \times \Omega \times \mathbb{R}^m \times \mathbb{R}^{m\times d} \to \mathbb{R}^m$, and $f(t, y, z)$ is an adapted stochastic process with respect to $\{\mathcal{F}_t\}(0 \le t \le T)$ for each (y, z).

Under some conditions for $f(t, y, z)$, Pardoux and Peng in [26] obtained the uniqueness solvability results for the BSDE (2.1). Properties and applications of BSDEs can be found in [18]. In our paper, we are interested in the numerical solution of the BSDE (2.1). We assume that the BSDE (2.1) admits a unique L^2-adapted solution (y_t, z_t).

Let $\mathcal{F}_s^{t,x}(t \le s \le T)$ be a σ-field generated by the Brownian motion $\{x + W_v - W_t, t \le v \le s\}$ starting from the time-space point (t, x), and let $\mathcal{F}^{t,x} = \mathcal{F}_T^{t,x}$. We use $\mathbb{E}[X]$ to denote the mathematical expectation of the random variable X and $\mathbb{E}_s^{t,x}[X]$ to denote the conditional mathematical expectation of the random variable X under the σ-field $\mathcal{F}_s^{t,x}(t \le s \le T)$; that is, $\mathbb{E}_s^{t,x}[X] = \mathbb{E}[X|\mathcal{F}_s^{t,x}]$. If $X = (X_{ij})_{m\times n}$ is a matrix or vector stochastic variable, we define the conditional mathematical expectation $\mathbb{E}_s^{t,x}[X]$ by $\mathbb{E}_s^{t,x}[X] = (\mathbb{E}_s^{t,x}[X_{ij}])_{m\times n}$.

3. Development of the numerical method. For simplicity, in this section, we start with the one-dimensional BSDE to develop our numerical method for the BSDE (2.1)

For the real axis $\mathbb{R}$, we introduce a space partition $\mathfrak{R}_h$ as

$$\mathfrak{R}_h = \{x_i | x_i \in \mathbb{R}, i \in \mathbb{Z}, x_i < x_{i+1}, \lim_{i\to+\infty} = +\infty, \lim_{i\to-\infty} = -\infty\}, \tag{3.1}$$

where $\mathbb{Z}$ is the set of all integer numbers, and all x_i's are deterministic. We use $h_i = x_{i+1} - x_i$ to denote the space step and $h = \max_{i\in\mathbb{Z}} h_i$ to denote the maximum space step.

For the time interval $[0, T]$, we introduce the time partition

$$\mathfrak{R}_{th} = \{t_i | t_i \in [0, T], i = 0, 1, \ldots, N_T, t_i < t_{i+1}, t_0 = 0, t_{N_T} = T\}, \tag{3.2}$$

where all t_n's are deterministic. Let $\Delta t_n = t_{n+1} - t_n$ be the time step, and let $\Delta t = \max_{0\le n<N_T} \Delta t_n$ be the maximum time step.

3.1. Reference equations. Let (y_t, z_t) be the adapted solution of the BSDE (2.1); then we have

$$y_t = \quad y_{t+\delta} + \int_t^{t+\delta} f(s, y_s, z_s)ds - \int_t^{t+\delta} z_s dW_s, \quad t \in [0, T), \tag{3.3}$$

where δ is a deterministic nonnegative number with $t + \delta \le T$. Taking the conditional mathematical expectation $\mathbb{E}_t^{t,x}$ on the two sides of (3.3), we deduce

$$y_t^{t,x} = \quad \mathbb{E}_t^{t,x}[y_{t+\delta}] + \int_t^{t+\delta} \mathbb{E}_t^{t,x}[f(s, y_s, z_s)]ds, \tag{3.4}$$

where $y_t^{t,x} = \mathbb{E}[y_t | \mathcal{F}_t^{t,x}]$; that is, $y_t^{t,x}$ is the value of y_t at the time-space point (t, x). With respect to the filtration $\mathcal{F}_t$, the integrand on the right-hand side of (3.4) is deterministic. We write the integral term in (3.4) in the form

$$\begin{aligned}\int_t^{t+\delta} \mathbb{E}_t^{t,x}[f(s, y_s, z_s)]ds = \quad & \delta\{(1-\theta_1)\mathbb{E}_t^{t,x}[f(t+\delta, y_{t+\delta}, z_{t+\delta})] \\ & +\theta_1 f(t, y_t^{t,x}, z_t^{t,x})\} + R_y,\end{aligned} \tag{3.5}$$

where R_y is defined by

$$R_y = \quad \int_t^{t+\delta} \{\mathbb{E}_t^{t,x}[f(s, y_s, z_s)] - (1-\theta_1)\mathbb{E}_t^{t,x}[f(t+\delta, y_{t+\delta}, z_{t+\delta})] - \theta_1 f(t, y_t^{t,x}, z_t^{t,x})\}ds$$

with the parameter $\theta_1 \in [0, 1]$. Inserting (3.5) into (3.4) leads to

$$\begin{aligned}y_t^{t,x} = \quad & \mathbb{E}_t^{t,x}[y_{t+\delta}] + \delta\{(1-\theta_1)\mathbb{E}_t^{t,x}[f(t+\delta, y_{t+\delta}, z_{t+\delta})] \\ & +\theta_1 f(t, y_t^{t,x}, z_t^{t,x})\} + R_y.\end{aligned} \tag{3.6}$$

Let $\Delta W_s = W_s - W_t$ for $t \le s \le t + \delta$; then ΔW_s is a standard Brownian motion with mean zero and variance $\sqrt{s-t}$. Now multiplying the two sides of (3.3) by $\Delta W_{t+\delta}$, and taking the mathematical expectation $\mathbb{E}_t^{t,x}$ to the two sides of the derived equation, by using Itô isometry formula we obtain

$$-\mathbb{E}_t^{t,x}[y_{t+\delta}\Delta W_{t+\delta}] = \quad \int_t^{t+\delta} \mathbb{E}_t^{t,x}[f(s, y_s, z_s)\Delta W_s]ds - \int_t^{t+\delta} \mathbb{E}_t^{t,x}[z_s]ds. \tag{3.7}$$

The two integral terms on the right-hand side of (3.7) are two standard integrals with respect to time under the filtration $\mathcal{F}_t$. We decompose the two terms on the right-hand side of (3.7) by

$$\int_t^{t+\delta} \mathbb{E}_t^{t,x}[f(s,y_s,z_s)\Delta W_s]ds$$

$$= \delta(1-\theta_2)\mathbb{E}_t^{t,x}[f(t+\delta,y_{t+\delta},z_{t+\delta})\Delta w_{t+\delta}] + R_{z1}, \tag{3.8}$$

$$-\int_t^{t+\delta} \mathbb{E}_t^{t,x}[z_s]ds = -\delta\left\{(1-\theta_2)\mathbb{E}_t^{t,x}[z_{t+\delta})] + \theta_2 z_t^{t,x}\right\} + R_{z2}, \tag{3.9}$$

where $z_t^{t,x}$ is the value of z_t at the time-space point (t,x) under the filtration $\mathcal{F}_t$, and

$$\begin{aligned} R_{z1} = &\int_t^{t+\delta} \{\mathbb{E}_t^{t,x}[f(s,y_s,z_s)\Delta W_s] - (1-\theta_2)\mathbb{E}[f(t+\delta,y_{t+\delta},z_{t+\delta})\Delta W_{t+\delta}]\}ds, \\ R_{z2} = &-\int_t^{t+\delta} \{\mathbb{E}_t^{t,x}[z_s] - (1-\theta_2)\mathbb{E}_t^{t,x}[z_{t+\delta}] - \theta_2 z_t^{t,x}\}\}ds \end{aligned}$$

with $\theta_2 \in [0,1]$. Combining the (3.7), (3.8), and (3.9), we obtain

$$\begin{aligned} -\mathbb{E}_t^{t,x}[y_{t+\delta}\Delta W_{t+\delta}] = &\quad \delta(1-\theta_2)\mathbb{E}_t^{t,x}[f(t+\delta,y_{t+\delta},z_{t+\delta})\Delta w_{t+\delta}] \\ &-\delta\left\{(1-\theta_2)\mathbb{E}_t^{t,x}[z_{t+\delta})] + \theta_2 z_t^{t,x}\right\} + R_z \end{aligned} \tag{3.10}$$

with $R_z = R_{z1} + R_{z2}$.

The equations (3.6) and (3.10) are the two fundamental reference equations obtained from (2.1). They are held for the deterministic time variable t and the deterministic parameters δ, θ_1, and θ_2. Based on these two reference equations, we then propose our numerical method in the following sections.

3.2. Discrete scheme.

3.2.1. Semidiscrete θ-scheme. In the reference equations (3.6) and (3.10), let $t = t_n$, $\delta = \Delta t_n$. Take $(y^n, z^n)(n = N_T, N_T - 1, \ldots, 0)$ as the approximations to the analytic solution (y_t, z_t) of the BSDE (2.1) at time levels $t_n(n = N_T, N_T - 1, \ldots, 0)$. Let $(y^n, z^n)(n = N_T, N_T - 1, \ldots, 0)$ satisfy (3.6) and (3.10) without terms R_y and R_z; that is, (y^n, z^n) satisfies

$$\begin{aligned} y^n = &\mathbb{E}_{t_n}^{t_n,x}[y^{n+1}] + \Delta t_n[(1-\theta_1^n)\mathbb{E}_{t_n}^{t_n,x}[f(t_{n+1},y^{n+1},z^{n+1})] \\ &+\theta_1^n f(t_n,y^n,z^n)], \end{aligned} \tag{3.11}$$

$$\begin{aligned} 0 = &\mathbb{E}_{t_n}^{t_n,x}[y^{n+1}\Delta W_{t_{n+1}}] + \Delta t_n(1-\theta_2^n)\mathbb{E}_{t_n}^{t_n,x}[f(t_{n+1},y^{n+1},z^{n+1})\Delta W_{t_{n+1}}] \\ &-\Delta t_n\{(1-\theta_2^n)\mathbb{E}_{t_n}^{t_n,x}[z^{n+1}] + \theta_2^n z^n\} \end{aligned} \tag{3.12}$$

for $n = N_T - 1, N_T - 2, \ldots, 2, 1, 0$ with the parameters θ_1^n and θ_2^n in $[0,1]$.

The equations (3.11) and (3.12) are a semidiscrete θ-scheme for the BSDE (2.1). The semidiscrete θ-scheme is nonlinear in the random variables y^n and z^n. So for the general nonlinear BSDE problems, some methods for finding (y^n, z^n) are needed. In this paper, we use Newton's method to solve (y^n, z^n). The procedure to solve (y^n, z^n) is as follows:

1. Let $y^{N_T} = \xi$.
2. If $n = N_T - 1$, let $\theta_2^n = 1$, and use (3.11) and (3.12) to solve (y^n, z^n).
3. If $0 \le n < N_T - 1$, use (3.11) and (3.12) to solve (y^n, z^n).

1. By choosing θ_1^n and θ_2^n in (3.11) and (3.12), we can obtain different methods. When we choose $\theta_1^n = \theta_2^n = 0$, we get the explicit Euler scheme. When $\theta_1^n = \theta_2^n \neq 0$, the θ-scheme is implicit. The implicit Euler scheme is the case $\theta_1^n = \theta_2^n = 1$, and the implicit Crank–Nicolson (CN) scheme is the case $\theta_1^n = \theta_2^n = \frac{1}{2}$. Our implicit θ-scheme for the BSDE (2.1) is different from the implicit schemes for solving ODEs and PDEs in that our θ-scheme is always explicit for solving z^n.

3.2.2. Fully discrete θ-scheme. To simulate (y^n, z^n) based on the semidiscrete θ-scheme (3.11) and (3.12), two more numerical approximations must be done; these are the approximation of the conditional mathematical expectation and the space approximation of (y^n, z^n). To do these, we take $x = x_i \in \mathfrak{R}_h$ in (3.11) and (3.12), and let $\hat{x}_i = x_i + \Delta W_{t_{n+1}}$. Then there is an integer $\hat{i} \in \mathbb{Z}$ such that $\hat{x}_i \in [x_{\hat{i}}, x_{\hat{i}+1}]$. Let $I_h u(\hat{x}_i)$ be an interpolation approximation of the function $u(x)$ at the space point $\hat{x}_i$ by using the values of $u(x)$ at a finite number of the space points x_j's in $\mathfrak{R}_h$ near the space point $\hat{x}_i$. The more accurate the simulations are required to be, the more points in $\mathfrak{R}_h$ near $\hat{x}_i$ need be used. For some complicated problems, some special interpolations may be taken.

For each $x = x_i \in \mathfrak{R}_h$ in (3.11) and (3.12), let $\Delta_h^k W_{t_{n+1}} (k = 1, 2, \ldots, N_E)$ be the N_E times realizations of $\sqrt{\Delta t_n} N(0,1)$, where $N(0,1)$ is a standard normal distribution with mean zero and variance 1, and let $\hat{x}_i^k = x_i + \Delta_h^k W_{t_{n+1}}$.

To approximate the conditional mathematical expectations, we use the Monte Carlo method. We write the first term on the right-hand side of (3.11) in the following form:

$$\mathbb{E}_{t_n}^{t_n,x_i}[y^{n+1}] = \mathbb{E}_{h,t_n}^{t_n,x_i}[y^{n+1}] + Err_y, \tag{3.13}$$

where

$$\mathbb{E}_{h,t_n}^{t_n,x_i}[y^{n+1}] = \frac{\sum_{k=1}^{N_E} I_h y^{n+1}(\hat{x}_i^k)}{N_E}, \quad \mathbb{E}rr_y = \mathbb{E}_{t_n}^{t_n,x_i}[y^{n+1}] - \frac{\sum_{k=1}^{N_E} I_h y^{n+1}(\hat{x}_i^k)}{N_E}.$$

The first term on the right-hand side of (3.13) is the Monte Carlo approximation to the mathematical expectation $\mathbb{E}_{t_n}^{t_n,x_i}[y^{n+1}]$, and the second term is the Monte Carlo approximation error.

Similarly we write the second term on the right-hand side of (3.11) as

$$\mathbb{E}_{t_n}^{t_n,x_i}[f(t_{n+1}, y^{n+1}, z^{n+1})] = \mathbb{E}_{h,t_n}^{t_n,x_i}[f(t_{n+1}, y^{n+1}, z^{n+1})] + Err_f \tag{3.14}$$

with the Monte Carlo approximation

$$\mathbb{E}_{h,t_n}^{t_n,x_i}[f(t_{n+1}, y^{n+1}, z^{n+1})] = \frac{\sum_{k=1}^{N_E} f(t_{n+1}, I_h y^{n+1}(\hat{x}_i^k), I_h z^{n+1}(\hat{x}_i^k))}{N_E}$$

and the corresponding approximation error

$$Err_f = \mathbb{E}_{t_n}^{t_n,x_i}[f(t_{n+1}, y^{n+1}, z^{n+1})] - \mathbb{E}_{h,t_n}^{t_n,x_i}[f(t_{n+1}, y^{n+1}, z^{n+1})].$$

Combining (3.11), (3.13), and (3.14), we obtain

$$\begin{aligned} y_i^n = \; & \mathbb{E}_{h,t_n}^{t_n,x_i}[y^{n+1}] + \Delta t_n[(1-\theta_1^n)\mathbb{E}_{h,t_n}^{t_n,x_i}[f(t_{n+1}, y^{n+1}, z^{n+1})] \\ & + \theta_1^n f(t_n, y_i^n, z_i^n)] + \Delta t_n(1-\theta_1^n) Err_f + Err_y, \end{aligned} \tag{3.15}$$

where (y_i^x, z_i^n) is the value of (y^n, z^n) at the space point x_i with respect to the filtration $\mathcal{F}_{t_n}$, and $\Delta t_n(1-\theta_1^n)Err_f + Err_y$ is the Monte Carlo approximation error.

Now, let us turn to approximate the three conditional mathematical expectations in (3.12). We write the first term on the right-hand side of (3.12) as

$$\mathbb{E}_{t_n}^{t_n,x_i}[y^{n+1}\Delta W_{t_{n+1}}] = \mathbb{E}_{h,t_n}^{t_n,x_i}[y^{n+1}\Delta W_{t_{n+1}}] + Err_{yw}, \tag{3.16}$$

where

$$\begin{aligned}\mathbb{E}_{h,t_n}^{t_n,x_i}[y^{n+1}\Delta W_{t_{n+1}}] &= \frac{\sum_{k=1}^{N_E} I_h y^{n+1}(\hat{x}_i^k)\Delta_h^k W_{t_{n+1}}}{N_E},\\ Err_{yw} &= \mathbb{E}_{t_n}^{t_n,x_i}[y^{n+1}\Delta W_{t_{n+1}}] - \mathbb{E}_{h,t_n}^{t_n,x_i}[y^{n+1}\Delta W_{t_{n+1}}],\end{aligned}$$

we write the second term on the right-hand side of (3.12) as

$$\mathbb{E}_{t_n}^{t_n,x_i}[f(t_{n+1},y^{n+1},z^{n+1})\Delta W_{t_{n+1}}] = \mathbb{E}_{h,t_n}^{t_n,x_i}[f(t_{n+1},y^{n+1},z^{n+1})\Delta W_{t_{n+1}}] + Err_{fw}, \tag{3.17}$$

where

$$\begin{aligned}\mathbb{E}_{h,t_n}^{t_n,x_i}[f(t_{n+1},y^{n+1},z^{n+1})\Delta W_{t_{n+1}}] &= \frac{\sum_{k=1}^{N_E} f(t_{n+1}, I_h y^{n+1}(\hat{x}_i^k), I_h z^{n+1}(\hat{x}_i^k))\Delta_h^k W_{t_{n+1}}}{N_E},\\ Err_{fw} &= \mathbb{E}_{t_n}^{t_n,x_i}[f(t_{n+1},y^{n+1},z^{n+1})\Delta W_{t_{n+1}}]\\ &\quad -\mathbb{E}_{h,t_n}^{t_n,x_i}[f(t_{n+1},y^{n+1},z^{n+1})\Delta W_{t_{n+1}}],\end{aligned}$$

and we write the third term on the right-hand side of (3.12) as

$$\mathbb{E}_{t_n}^{t_n,x_i}[z^{n+1}] = \mathbb{E}_{h,t_n}^{t_n,x_i}[z^{n+1}] + Err_z, \tag{3.18}$$

where

$$\mathbb{E}_{h,t_n}^{t_n,x_i}[z^{n+1}] = \frac{\sum_{k=1}^{N_E} I_h z^{n+1}(\hat{x}_i^k)}{N_E}, \quad Err_z = \mathbb{E}_{t_n}^{t_n,x_i}[z^{n+1}] - \mathbb{E}_{h,t_n}^{t_n,x_i}[z^{n+1}].$$

Now combining (3.12), (3.16), (3.17), and (3.18), we get

$$\begin{aligned}&\mathbb{E}_{h,t_n}^{t_n,x_i}[y^{n+1}\Delta W_{t_{n+1}}] + \Delta t_n(1-\theta_2^n)\mathbb{E}_{h,t_n}^{t_n,x_i}[f(t_{n+1},y^{n+1},z^{n+1})\Delta W_{t_{n+1}}]\\ &\quad -\Delta t_n\{(1-\theta_2^n)\mathbb{E}_{h,t_n}^{t_n,x_i}[z^{n+1}] + \theta_2^n z_i^n\}\\ &\quad +Err_{yw} + \Delta t_n(1-\theta_2^n)Err_{fw} - \Delta t_n(1-\theta_2^n)Err_z = 0,\end{aligned} \tag{3.19}$$

where $Err_{yw}+\Delta t_n(1-\theta_2^n)Err_{fw}-\Delta t_n(1-\theta_2^n)Err_z$ is the Monte Carlo approximation error.

From probability theory, we know that the Monte Carlo approximation errors Err_y in (3.13), Err_z in (3.18), Err_f in (3.14), Err_{yw} in (3.16), and Err_{fw} in (3.17) converge to 0 with the convergence rate $\frac{1}{\sqrt{N_E}}$ as $N_E \to \infty$. So we can choose a proper integer N_E in the Monte Carlo method to approximate the mathematical expectations with the required accuracy.

Now we take (y_i^n, z_i^n) as the approximation of the solution (y_t, z_t) at the time-space point (t_n, x_i) with $t_n \in \mathfrak{R}_{th}$ and $x_i \in \mathfrak{R}_h$. Based on the two equations (3.15) and (3.19), by omitting the Monte Carlo error terms, we propose a fully time-space discrete θ-scheme to solve (y_t, z_t), the solution of the BSDE (2.1), as for the given terminal values $y_i^{N_T} (i \in \mathfrak{R}_h)$, select θ_1^n and θ_2^n in [0,1], and find (y_i^n, z_i^n) that satisfies

$$y_i^n = \mathbb{E}_{h,t_n}^{t_n,x_i}[y^{n+1}] + \Delta t_n\{(1-\theta_1^n)\mathbb{E}_{h,t_n}^{t_n,x_i}[f(t_{n+1}, y^{n+1}, z^{n+1})] + \theta_1^n f(t_n, y_i^n, z_i^n)\}, \tag{3.20}$$

$$0 = \mathbb{E}_{h,t_n}^{t_n,x_i}[y^{n+1}\Delta W_{t_{n+1}}] + \Delta t_n(1-\theta_2^n)\mathbb{E}_{h,t_n}^{t_n,x_i}[f(t_{n+1}, y^{n+1}, z^{n+1})\Delta W_{t_{n+1}}] - \Delta t_n\{(1-\theta_2^n)\mathbb{E}_{h,t_n}^{t_n,x_i}[z^{n+1}] + \theta_2^n z_i^n\} \tag{3.21}$$

for $i \in \mathbb{Z}$ and $n = N_T - 1, N_T - 2, \ldots, 2, 1, 0$ with $\theta_2^{N_T-1} = 1$.

In the fully discrete θ-scheme (3.20) and (3.21), for each point (t_n, x_i), the (y_i^n, z_i^n) is unknown. Here we suggest a procedure to solve (y_i^n, z_i^n) from the two equations (3.20) and (3.21) for $i \in \mathbb{Z}$ and $n = N_T - 1, N_T - 2, \ldots, 2, 1, 0$. The procedure is as follows:

1. Given $y_i^{N_T}, \quad i \in \mathbb{Z}$.
2. For $n = N_T - 1$, let $\theta_2^n = 1$ in (3.21), and solve (y_i^n, z_i^n) from the two equations (3.20) and (3.21) for each $i \in \mathbb{Z}$.
3. For $0 \le n < N_T - 1$, solve (y_i^n, z_i^n) from the two equations (3.20) and (3.21) for each $i \in \mathbb{Z}$.

. . . . 2.

1. In the fully discrete θ-scheme (3.20) and (3.21), in order to solve z_i^n, it is necessary for θ_2^n not to be zero. And solving z_i^n is always explicit from (3.21). For the linear BSDE, solving y_i^n is explicit. For the nonlinear BSDE, some numerical methods to solve y_i^n are needed if $\theta_1^n \neq 0$.
2. Solving (y_i^n, z_i^n) with different $i \in \mathbb{Z}$ is totally uncoupled. We can solve $(y_i^n, z_i^n)(i \in \mathbb{Z})$ in parallel.
3. In practical applications, we are often interested in the values of (y_0, z_0) in a bounded domain. Thus from the Brownian motion theory, the data of $(y_i^{N_T}, z_i^{N_T})$ on some bounded domain is needed under some required accuracy.
4. The terminal condition is y_T. In the first computing step, we set $\theta_2^{N_T-1} = 1$ for solving $z_i^{N_T-1}$. For $\theta_2^{N_T-1} = 1$, the accuracy of the scheme is not high. To achieve the high accuracy, a special time partition step Δt_{N_T-1} is needed. For example, we can choose $\theta_1^n = \theta_2^n = \frac{1}{2}$ for $n < N_T - 1$ and choose $\theta_2^n = 1$ and let $\Delta t_n = (\Delta t_{n-1})^2$ for $n = N_T - 1$.

4. Discrete θ-scheme for general BSDEs. Based on the method of constructing the discrete θ-scheme (3.20) and (3.21) in section 3, in this section, we generalize the scheme and propose a new numerical scheme for the general BSDEs (2.1) driven by a standard d-dimensional Brownian motion $W_t = (W_t^1, W_t^2, \ldots, W_t^d)^*$, where $W_t^i (i = 1, 2, \ldots, d)$ are the independent standard one-dimensional Brownian motions, and the notation * is the transpose operator for matrix or vector.

We use $\mathbb{R}^d$ to denote the d-dimensional Euclidean space. $x \in \mathbb{R}^d$ means $x = (x^1, x^2, \ldots, x^d)$ with $x^i \in \mathbb{R} (i = 1, 2, \ldots, d)$. Let $\mathbb{Z}^d = \{i | i = (i_1, i_2, \ldots, i_d), i_k \in \mathbb{Z}, k = 1, 2, \ldots, d\}$ and $x_i = (x_{i_1}^1, x_{i_2}^2, \ldots, x_{i_d}^d)$ for $i = (i_1, i_2, \ldots, i_d) \in \mathbb{Z}^d$.

Let $(y_t, z_t) : (t, \Omega) \to \mathbb{R}^m \times \mathbb{R}^{m \times d}$ be the solution of the BSDE (2.1), where

$y_t = (y_t^1, y_t^2, \ldots, y_t^m)^*$ and $z_t = (z_t^{i,j})_{m\times d}$. Similarly to obtaining (3.3), we have

$$y_t^k = y_{t+\delta}^k + \int_t^{t+\delta} f^k(s, y_s, z_s)ds - \int_t^{t+\delta} \sum_{j=1}^{d} z_s^{k,j} dW_s^j, \quad t \in [0, T), \tag{4.1}$$

where f^k is the kth component of the vector function f. Taking the conditional mathematical expectation $\mathbb{E}_t^{t,x}$ on the two sides of (4.1), we obtain

$$y_t^{k,t,x} = \mathbb{E}_t^{t,x}[y_{t+\delta}^k] + \int_t^{t+\delta} \mathbb{E}_t^{t,x}[f^k(s, y_s, z_s)]ds, \quad t \in [0, T), t+\delta \in [0, T] \tag{4.2}$$

for $k = 1, 2, \ldots, m$. We rewrite (4.2) in the following vector form:

$$y_t^{t,x} = \mathbb{E}_t^{t,x}[y_{t+\delta}] + \int_t^{t+\delta} \mathbb{E}_t^{t,x}[f(s, y_s, z_s)]ds, \quad t \in [0, T), t+\delta \in [0, T]. \tag{4.3}$$

As we did for $m = d = 1$ in section 3, for $t \le s \le t+\delta$, we let $\Delta W_s^l = W_s^l - W_t^l (l = 1, 2, \ldots, d)$, which are the independent standard one-dimensional Brownian motions with mean zero and variance $\sqrt{s-t}$. Multiplying the two sides of (4.1) by $\Delta W_{t+\delta}^l$ and taking the conditional mathematical expectation $\mathbb{E}_t^{t,x}$ on the two sides of the derived equation, by the Itô isometry formula we then obtain

$$-\mathbb{E}_t^{t,x}[y_{t+\delta}^k \Delta W_{t+\delta}^l] = \int_t^{t+\delta} \mathbb{E}_t^{t,x}[f^k(s, y_s, z_s)\Delta W_s^l]ds - \int_t^{t+\delta} \mathbb{E}_t^{t,x}[z_s^{k,l}]ds \tag{4.4}$$

for $k = 1, 2, \ldots, m$ and $l = 1, 2, \ldots, d$, or in the matrix form

$$-\mathbb{E}_t^{t,x}[y_{t+\delta}\Delta W_{t+\delta}^*] = \int_t^{t+\delta} \mathbb{E}_t^{t,x}[f(s, y_s, z_s)\Delta W_s^*]ds - \int_t^{t+\delta} \mathbb{E}_t^{t,x}[z_s]ds. \tag{4.5}$$

Here for a matrix random variable $X = (X_{lk})_{m\times d}$, the conditional mathematical expectation $\mathbb{E}_t^{t,x}[X]$ is defined by

$$\mathbb{E}_t^{t,x}[X] = (\mathbb{E}_t^{t,x}[X^{l,j}])_{m\times d}. \tag{4.6}$$

We now introduce the partitions of the Euclidean space $\mathbb{R}^d$ and the time interval $[0, T]$. The partition of the Euclidean space $\mathbb{R}^d$ is

$$\mathfrak{R}_h^d = \overbrace{\mathfrak{R}_h \times \mathfrak{R}_h \times \cdots \times \mathfrak{R}_h}^{d \ times}, \tag{4.7}$$

where $\mathfrak{R}_h$ is defined in section 3. The partition of the time interval $[0, T]$ is $\mathfrak{R}_{th}$ defined in section 3.

Let the approximations of the conditional mathematical expectations in (4.3) and (4.5) be defined as those in (3.13), (3.14), (3.16), (3.18), and (3.17); that is,

$$\mathbb{E}_{h,t_n}^{t_n,x_i}[X] = (\mathbb{E}_{h,t_n}^{t_n,x_i}[X^{l,j}]). \tag{4.8}$$

In the computation of $\mathbb{E}_{h,t_n}^{t_n,x_i}[X^{l,j}]$, the random variable $\hat{x}_i = x_i + \Delta W_{t_{n+1}}$ for $x_i \in \mathbb{R}_h^d$ and the d-dimensional interpolation operator I_h are used.

The number of the d-dimensional interpolation points near the space point $\hat{x}_i$ depends on the required accuracy. For the high-dimensional problems, the space interpolations may be expensive. It is still a challenging problem to propose an efficient high-dimensional interpolation method. In this paper, we focus our attention

on the numerical methods for the BSDE (2.1) and use the bilinear polynomial, bicubic polynomial, and bicubic spline interpolations to solve the two-dimensional BSDE (2.1) in Example 3 in section 5.

Now let (y_i^n, z_i^n) be an approximation of (y_t, z_t) at the time-space point (t_n, x_i) with $t_n \in \mathfrak{R}_{th}$ and $x_i \in \mathfrak{R}_h^d$. Based on the two equations (4.2) and (4.5), by using the same procedure of getting the fully discrete θ-scheme (3.20) and (3.21) in section 3, we propose a new fully discrete θ-scheme for the general BSDE (2.1) as follows: Given $y_i^{N_T} (i \in \mathbb{Z}^d)$, find $(y_i^n, z_i^n) \in \mathbb{R}^m \times \mathbb{R}^{m\times d}$ that satisfies

$$y_i^n = \mathbb{E}_{h,t_n}^{t_n,x_i}[y^{n+1}] + \Delta t_n\{(1-\theta_1^n)\mathbb{E}_{h,t_n}^{t_n,x_i}[f(t_{n+1}, y^{n+1}, z^{n+1})] + \theta_1^n f(t_n, y_i^n, z_i^n)\}, \tag{4.9}$$

$$0 = \mathbb{E}_{h,t_n}^{t_n,x_i}[y^{n+1}(\Delta W_{t_{n+1}})^*] + \Delta t_n(1-\theta_2^n)\mathbb{E}_{h,t_n}^{t_n,x_i}[f(t_{n+1}, y^{n+1}, z^{n+1})(\Delta W_{t_{n+1}})^*] - \Delta t_n\{(1-\theta_2^n)\mathbb{E}_{h,t_n}^{t_n,x_i}[z^{n+1}] + \theta_2^n z_i^n\} = 0 \tag{4.10}$$

for $i \in \mathbb{Z}^d$ and $n = N_T - 1, N_T - 2, \ldots, 2, 1, 0$, where θ_1^n and θ_2^n are two parameters in [0,1] with $\theta_2^{N_T-1} = 1$.

A solution procedure for the scheme (4.9) and (4.10) can be described as follows:

1. given $y_i^{N_T}$, $i \in \mathbb{Z}^d$, $n = N_T - 1$;
2. solve $z_i^n (i \in \mathbb{Z}^{m\times d})$ from (4.10);
3. solve $y_i^n (i \in \mathbb{Z}^d)$ from (4.9);
4. if $n = 0$, stop; else let $n \leftarrow n - 1$; goto step 2.

Remark 3. Like the one-dimensional fully discrete θ-scheme, in the fully discrete θ-scheme (4.9) and (4.10), the unknowns are y_i^n and z_i^n. Solving z_i^n is explicit from (4.10). For $\theta_1^n \neq 0$, solving y_i^n is implicit. At each time level $t = t_n$, $y_l^n (l \in \mathbb{Z}^d)$ are totally uncoupled among the space grids and can be solved parallelly.

5. Numerical experiments. The aim of this section is to use some numerical examples to demonstrate the effectiveness and accuracy of our method for solving the BSDE (2.1). In the examples, the time and space step sizes are Δt and h, respectively. The positive integer N_T is the total discrete time steps $N_T = \frac{T}{\Delta t}$ with the finite terminal time T. In the tables, the notations of $|\cdot|$, CR, EMC, HS, and Scheme$^\alpha (\alpha = L, C, S)$ represent, respectively, the standard Euclidean norm in the Euclidean space $\mathbb{R}^m$ or $\mathbb{R}^{m\times d}$, the convergence rate with respect to the time step sizes, the implicit Euler Monte Carlo method related to our scheme with $\theta_1^n = \theta_2^n = 1$, the Heun's method in which the cubic polynomial interpolation operator I_h is used, and our method in which the linear polynomial, cubic polynomial, and cubic spline interpolation operators I_h are used with $\theta_1^n = \theta_2^n = \frac{1}{2}$ for $\alpha = L, C, S$, respectively.

To obtain the convergence rates for the different interpolations, we adjust the space partition step h to the time partition step Δt. As we know, for ordinary differential equations, the CN scheme is a second order method. In our method, the time discretization is based on the integral approximations with the deterministic functions, which are the conditional mathematical expectations. The details of the approximations are described in section 3. If the functions $f(t, y, z)$ and $\Phi(x)$ are sufficiently smooth, the discrete approximation errors in the time direction are $C_t \Delta t$ for $\theta_1^n \neq \frac{1}{2}$ or $\theta_2^n \neq \frac{1}{2}$, and $C_t(\Delta t)^2$ for $\theta_1^n = \theta_2^n = \frac{1}{2}$. The space interpolation error is $I_h u - u = C_h h^r$ with a positive number r, which depends on the interpolation operator I_h and the regularity of the function u with respect to the space variables. For the linear polynomial interpolation operator I_h, $r = 2$. For the cubic polynomial and cubic spline interpolation operators I_h, $r = 4$. In numerical simulations, the time

errors and the space errors should be balanced, so the space discrete step h and the time discrete step Δt should be chosen to satisfy the equality $C_h h^r = C_t(\Delta t)^2$. In our numerical experiments, we use $h = \Delta t$ when the linear polynomial interpolation is used, and use $h = (\Delta t)^{\frac{1}{2}}$ when the cubic polynomial and cubic spline interpolations are used. If the order of the accuracy in the time direction is α, then the order of the accuracy in the space is α for the linear interpolation and 2α for the cubic polynomial and cubic spline interpolations.

In the Monte Carlo approximations, we use different random sampling numbers N_E for different time partitions. The sampling numbers N_E are listed in the corresponding tables.

It is well known that the random number generator is very important for Monte Carlo method and that the accuracy of the Monte Carlo method depends not only on the sampling number N_E but also on the quality of the random number generator. In this paper, we choose the software SPRNG (Scalable Parallel Random Number Generators Library) as our random number generator. We use SPRNG to generate uniform distribution samples and then use the well-known Box–Muller method [7] to transform these samples into the ones with normal distributions. The software SPRNG and its details can be found on the website http://sprng.cs.fsu.edu/.

To compare our method with other ones, we also use the Euler method and the Heun's method [8] combined with the Monte Carlo method to solve the BSDEs in the following numerical examples, and list their errors and their convergence rates in the corresponding tables.

Example 1. Consider the following nonlinear BSDE:

$$\begin{cases} -dY_t = \lambda(Y_t)|Z_t|^2 dt - Z_t dW_t, 0 \le t \le T, \\ Y_T = \Phi(W_T), \end{cases} \tag{5.1}$$

where $\Phi(x)$ and $\lambda(x)$ are two deterministic functions of x, and W_T is the Brownian motion at time T. The analytic solution of the problem (5.1) is in the form

$$Y_0 = h^{-1}(\mathbb{E}(h(\Phi(X_T)))) \tag{5.2}$$

where the function $h(x)$ is the solution of the following ordinary differential equation:

$$-h'(x)\lambda(x) + \frac{1}{2}h''(x) = 0. \tag{5.3}$$

When $\lambda(x) = ax$, the analytic solution of (5.1) is

$$Y_0 = \frac{-C_2\sqrt{C_1 a} + \tan^{-1}(\mathbb{E}[\tan(\Phi(X_T)\sqrt{C_1 a} + C_2\sqrt{C_1 a})])}{\sqrt{C_1 a}}, \tag{5.4}$$

where a, C_1, and C_2 are parameters, $\tan(\cdot)$ is the triangle function *tangent*, and $\tan^{-1}(\cdot)$ is the inverse of the function $\tan(\cdot)$.

In this numerical example, we take $a = 0.01$, $C_1 = 1$, and $C_2 = 0$ in (5.4). We choose the terminal time $T = 1$ and the terminal function $\Phi(x) = |\cos(x)|$. The solution of (5.1) at time $t = 0$ is $Y_0 = 0.6943$ from (5.4). We list the absolute errors between the analytic solution Y_0 and the numerical solution y_0^0 and list the convergence rates in Table 5.1. The plots of $\log_2(|Y_0 - y_0^0|)$ with respect to $\log_2(N_T)$ are drawn in Figure 5.1.

TABLE 5.1
Errors and convergence rates.

	$\lvert Y_0 - y_0^0\rvert$						CR
N_T	2	4	8	16	32	64	
N_E	400	1000	4000	10000	20000	40000	
EMC	0.1259	0.0872	0.0510	0.0237	0.0177	0.0075	0.80
HS	0.0206	0.0093	0.0039	0.0015	0.0010	0.0006	0.99
Scheme^L	0.0312	0.0086	0.0021	0.0009	0.0003	0.0001	1.66
Scheme^C	0.0302	0.0060	0.0017	0.0005	0.0002	0.00008	1.71
Scheme^S	0.0242	0.0048	0.0033	0.0013	0.0002	0.00004	1.98

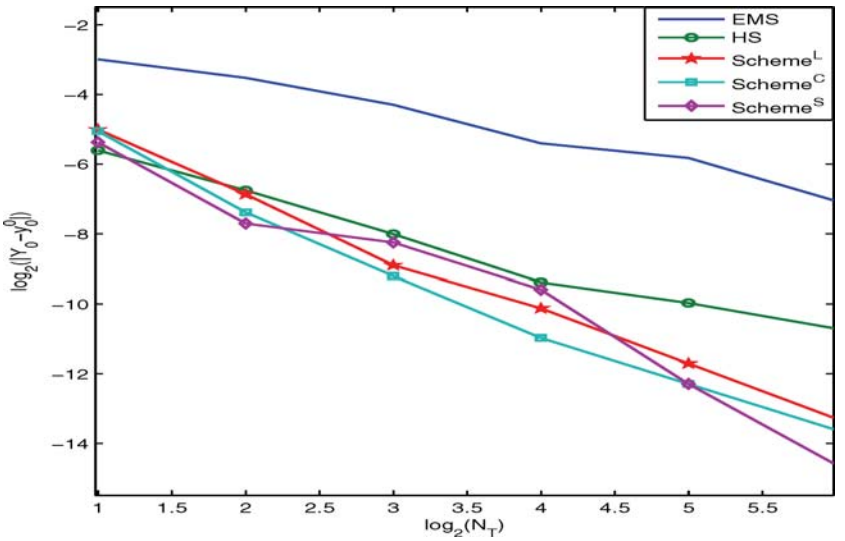


FIG. 5.1. *The plots of* $\log_2(|Y_0 - y_0^0|)$ *with respect to* $\log_2(N_T)$.

2. The BSDE is

$$\begin{cases} -dY_t = \left(\dfrac{Y_t}{2} - Z_t\right) dt - Z_t dW_t, \ 0 \le t \le T, \\ Y_T = \sin(W_T + T). \end{cases} \tag{5.5}$$

The analytic solution of (5.5) is

$$(Y_t, Z_t) = (\sin(W_t + t), \cos(W_t + t)), \tag{5.6}$$

where W_t is a standard Brownian motion. In this example, we take $T = 1$. The solution at $t = 0$ is $(Y_t, Z_t)|_{t=0} = (Y_0, Z_0) = (0, 1)$. The errors $|Y_0 - y_0^0|$ and $|Z_0 - z_0^0|$ and their convergence rates are listed in Table 5.2. We plot the lines of $\log_2(|Y_0 - y_0^0|)$ with respect to $\log_2(N_T)$ in Figure 5.2.

3. Let $\mathbf{W}_t = (W_t^1, W_t^2)^*$ be a two-dimensional Brownian motion, W_t^1 and W_t^2 independent standard one-dimensional Brownian motions. The BSDE driven by $\mathbf{W}_t$ is

$$-dY_t = \left(\frac{Y_t - d}{2}(b^2 + c^2) - Z_t \cdot \mathbf{A}\right) dt - Z_t \cdot d\mathbf{W}_t, \quad 0 \le t \le T, \tag{5.7}$$

TABLE 5.2
Errors and convergence rates.

	$\lvert Y_0 - y_0^0\rvert$						CR
N_T	2	4	8	16	32	64	
N_E	400	1000	4000	10000	20000	40000	
EMC	0.0735	0.0327	0.0179	0.0086	0.0035	0.0015	1.13
HS	0.1334	0.0348	0.0198	0.0112	0.0053	0.0020	0.82
Scheme^L	0.0395	0.0114	0.0034	0.0010	0.0004	0.00012	1.7
Scheme^C	0.0357	0.0095	0.0094	0.0020	0.0004	0.0001	1.91
Scheme^S	0.0351	0.0072	0.0023	0.0008	0.0003	0.00008	2.03
	$\lvert Z_0 - z_0^0\rvert$						CR
N_T	2	4	8	16	32	64	
N_E	400	1000	4000	10000	20000	40000	
EMC	0.2356	0.0587	0.0065	0.0502	0.0402	0.0565	0.39
HS	0.2316	0.1002	0.0720	0.0318	0.0432	0.0251	0.64
Scheme^L	0.2874	0.1056	0.0371	0.0136	0.0102	0.0035	1.27
Scheme^C	0.2706	0.0752	0.0253	0.0121	0.0059	0.0016	1.48
Scheme^S	0.2911	0.0553	0.0281	0.0033	0.0042	0.0011	1.61

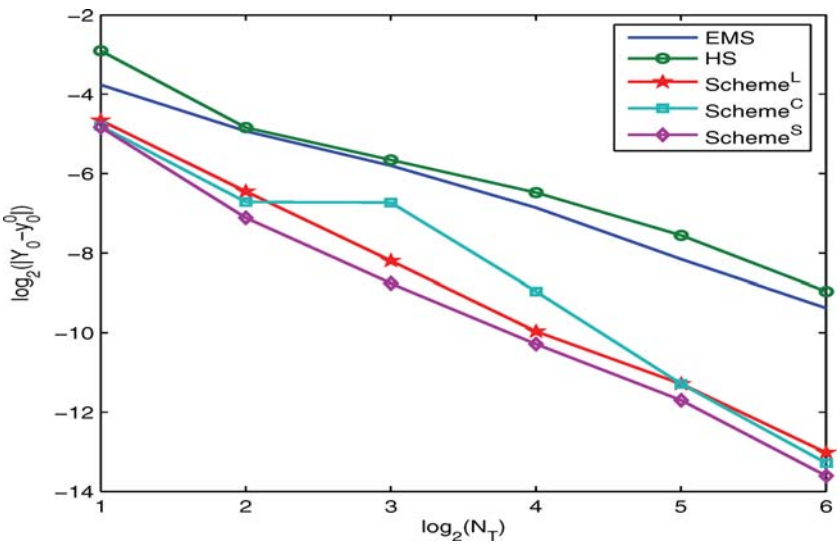


FIG. 5.2. *The plots of* $\log_2(|Y_0 - y_0^0|)$ *with respect to* $\log_2(N_T)$.

with the terminal condition $Y_T = a\sin(\mathbf{M}\cdot\mathbf{W}_T + T) + d$, where $\mathbf{A} = (\frac{1}{2b}, \frac{1}{2c})^*$, $\mathbf{M} = (b, c)$, $Z_t \cdot \mathbf{A} = \frac{Z_t^1}{2b} + \frac{Z_t^2}{2c}$, and $Z_t \cdot d\mathbf{W}_t = Z_t^1 dW_t^1 + Z_t^2 dW_t^2$ with the constant parameters a, b, c, and d. The analytic solution (Y_t, Z_t) is in the following form:

$$\begin{cases} Y_t = a\sin(\mathbf{M}\cdot\mathbf{W_t} + t) + d, \\ Z_t = (ab\cos(\mathbf{M}\cdot\mathbf{W}_t + t), ac\cos(\mathbf{M}\cdot\mathbf{W}_t + t)), \end{cases} \tag{5.8}$$

where $\mathbf{M}\cdot\mathbf{W}_t = bW_t^1 + cW_t^2$.

TABLE 5.3
Errors and convergence rates.

	$\lvert Y_0 - y_0^0\rvert$						CR
N_T	2	4	8	16	32	64	
N_E	1000	2000	6000	10000	20000	40000	
EMC	0.3698	0.2504	0.1385	0.0617	0.0276	0.0129	0.97
HS	0.2687	0.1592	0.0598	0.0329	0.0115	0.0070	1.05
Scheme^L	0.1483	0.0419	0.0106	0.0037	0.0008	0.0003	1.78
Scheme^C	0.0996	0.0174	0.0058	0.0023	0.0006	0.0002	1.80
Scheme^S	0.0353	0.0122	0.0039	0.0011	0.0003	0.00003	2.04
	$\lvert Z_0 - z_0^0\rvert$						CR
N_T	2	4	8	16	32	64	
N_E	1000	2000	6000	10000	20000	40000	
EMC	0.6987	0.3561	0.2010	0.0782	0.335	0.0129	1.14
HS	0.6239	0.3517	0.1862	0.1021	0.0435	0.0390	0.80
Scheme^L	0.3561	0.1181	0.0462	0.0113	0.0031	0.0012	1.64
Scheme^C	0.2826	0.1032	0.0394	0.0098	0.0027	0.0008	1.70
Scheme^S	0.3125	0.1092	0.0435	0.0103	0.0029	0.0007	1.76

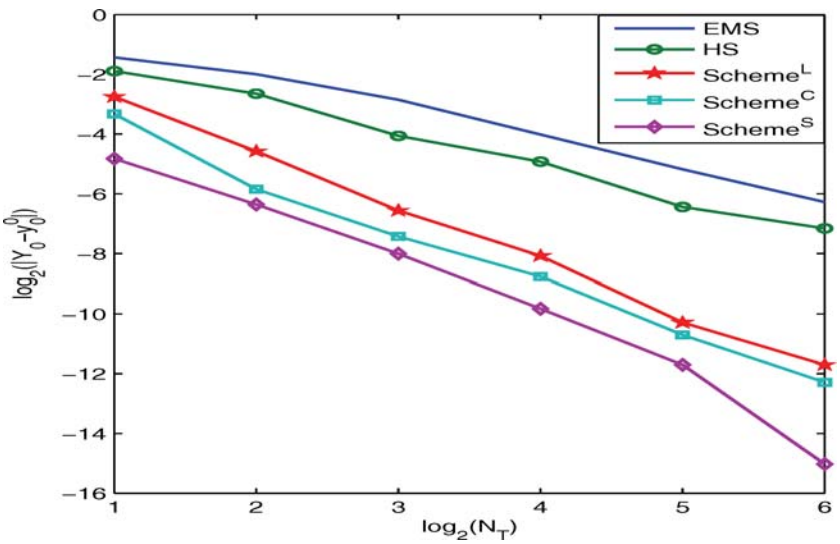


FIG. 5.3. *The plots of* $\log_2(|Y_0 - y_0^0|)$ *with respect to* $\log_2(N_T)$.

In this example, we take $T = 1$, $a = b = c = 1$, and $d = 0$. The solution at $t = 0$ is $(y_0, z_0) = (0, [1, 1])$. The absolute errors $|Y_0 - y_0^0|$ and $|Z_0 - z_0^0|$ and their convergence rates are listed in Table 5.3. The plots of $\log_2(|Y_0 - y_0^0|)$ with respect to $\log_2(N_T)$ are in Figure 5.3.

4. In this example, we use our method to solve one-dimensional evaluation problems in the stock market. Here we briefly describe the mathematical models in the BSDE form for these problems.

Consider a bond and a stock traded in a stock market. The bond price p_t and the stock price S_t, respectively, obey

$$\begin{cases} dp_t = r_t p_t dt, & t \geq 0, \\ p_0 = p, \end{cases} \tag{5.9}$$

and

$$\begin{cases} dS_t = & b_t S_t dt + \sigma_t S_t dW_t, \quad t \geq 0, \\ S_0 = & x, \end{cases} \tag{5.10}$$

where $\{W_t\}_{t\geq 0}$ is a standard Brownian motion, r_t is the return rate of the bond, b_t is the expected return rate of the stock, $\sigma_t > 0$ is the volatility of the stock, and r_t, b_t, σ_t, and ${\sigma_t}^{-1}$ are all bounded.

An investor with wealth Y_t at time t uses π_t money to buy stock and uses $Y_t - \pi_t$ to buy bond. Suppose that the stock pays dividends continuously with a bounded dividend rate $d(t, S_t)$ at the time t. Then the processes Y_t and π_t satisfy the following SDE [26, 18, 24, 25]:

$$-dY_t = -(r_t Y_t + (b_t - r_t + d(t, S_t))\pi_t)dt - \sigma_t \pi_t dW_t. \tag{5.11}$$

Let $Z_t = \sigma_t \pi_t$. Then (Y_t, Z_t) satisfies

$$-dY_t = -(r_t Y_t + (b_t - r_t + d(t, S_t))\sigma_t^{-1} Z_t)dt - Z_t dW_t. \tag{5.12}$$

When the money $Y_t - \pi_t \geq 0$, the investor invests $Y_t - \pi_t$ in the bond with the rate r_t. Otherwise, he will borrow $(Y_t - \pi_t)^- = \min\{0, Y_t - \pi_t\}$ money at the rate R_t. As we obtained (5.12), we have that (Y_t, Z_t) satisfies

$$\begin{aligned} -dY_t = \quad & -(r_t Y_t + (b_t - r_t + d(t, S_t)){\sigma_t}^{-1} Z_t \\ & +(R_t - r_t)(Y_t - \sigma_t^{-1} Z_t)^-)dt - Z_t dW_t \end{aligned} \tag{5.13}$$

for $t \in [0, T)$ with Z_t defined as the one in (5.12).

For the European call option, the terminal condition is given at the mature time T for (5.12) and (5.13) by

$$Y_T = \max\{S_T - K, 0\}, \tag{5.14}$$

where S_T is the solution S_t of the problem (5.10) at the mature time T, and K is the strike price.

When $r_t = r$, $b_t = b$, and $\sigma_t = \sigma$, and $d(t, S_t) = d$ are constants in (5.12) and (5.13), we can obtain their analytic solutions in closed forms. The analytic solution of the BSDE (5.12) with the terminal condition (5.14) is

$$\begin{cases} Y_t = V(t, S_t) = S_t e^{-d(T-t)} N(d_1(S_t)) - K e^{-r(T-t)} N(d_0(S_t)), \\ Z_t = \dfrac{\partial V}{\partial S}\sigma = S_t e^{-d(T-t)} N(d_1(S_t))\sigma, \\ d_0(S_t) = \dfrac{1}{\sigma\sqrt{T-t}} \ln\left(\dfrac{S_t}{K e^{(r-d)(T-t)}}\right) - \dfrac{1}{2}\sigma\sqrt{T-t}, \\ d_1(S_t) = d_0(S_t) + \sigma\sqrt{T-t}, \end{cases} \tag{5.15}$$

and the analytic solution of the BSDE (5.13) with the terminal condition (5.14) is

$$\begin{cases} Y_t = S_t e^{-d(T-t)} N(d_1(S_t)) - K e^{-R(T-t)} N(d_0(S_t)), \\ Z_t = S_t e^{-d(T-t)} N(d_1(S_t))\sigma, \\ d_0(S_t) = \dfrac{1}{\sigma\sqrt{T-t}} \ln\left(\dfrac{S_t}{K e^{(R-d)(T-t)}}\right) - \dfrac{1}{2}\sigma\sqrt{T-t}, \\ d_1(S_t) = d_0(S_t) + \sigma\sqrt{T-t}, \end{cases} \tag{5.16}$$

TABLE 5.4
The errors and convergence rates.

	$\lvert Y_0 - y_0^0\rvert$						CR
N_T	2	4	8	16	32	64	
N_E	400	1000	4000	10000	20000	40000	
EMC	0.1038	0.0872	0.0473	0.0310	0.0223	0.0108	0.65
HS	0.0632	0.0295	0.0109	0.0112	0.0087	0.0015	1.03
Scheme^L	0.1572	0.0451	0.0105	0.0027	0.0002	0.00005	2.30
Scheme^C	0.1201	0.0310	0.0089	0.0025	0.0009	0.0002	1.84
Scheme^S	0.0973	0.0263	0.0062	0.0020	0.0006	0.0002	1.79
	$\lvert Z_0 - z_0^0\rvert$						CR
N_T	2	4	8	16	32	64	
N_E	400	1000	4000	10000	20000	40000	
EMC	1.1769	0.2431	0.7025	0.4432	0.4607	0.5826	0.20
HS	1.8355	1.4859	0.7811	0.0756	0.1247	0.0939	0.85
Scheme^L	0.7817	0.4466	0.2712	0.0632	0.0162	0.0127	1.18
Scheme^C	0.9834	0.5382	0.2437	0.1084	0.0722	0.0384	0.94
Scheme^S	1.1326	0.8730	0.5218	0.2421	0.0989	0.0492	0.90

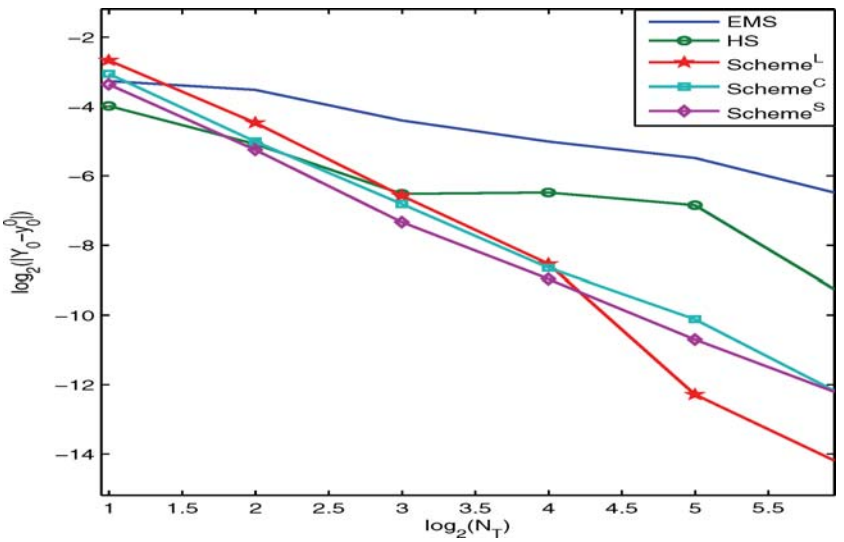


FIG. 5.4. *The plots of* $\log_2(|Y_0 - y_0^0|)$ *with respect to* $\log_2(N_T)$.

where N is the cumulative normal distribution function.

For the BSDE problem (5.12) and (5.14), we take $T = 0.33$, $K = S_0 = 100$, $r = 0.03$, $\mu = 0.05$, $d = 0.04$, and $\sigma = 0.2$. From (5.15), we know $(Y_0, Z_0) = (4.3671, 10.0950)$. We list the absolution errors $|Y_0 - y_0^0|$ and $|Z_0 - z_0^0|$ and their convergence rates in Table 5.4, and plot the errors $\log_2(|Y_0 - y_0^0|)$ with respect to $\log_2(N_T)$ in Figure 5.4.

For the BSDE problem (5.13) and (5.14), we take $T = 0.33$, $K = S_0 = 100$, $r = 0.03$, $\mu = 0.05$, $d = 0$, $\sigma = 0.2$, and $R = 0.09$. From (5.16), we have $(Y_0, Z_0) = (6.1269, 12.4796)$. We list the absolute errors $|Y_0 - y_0^0|$ and $|Z_0 - z_0^0|$ and their convergence rates in Table 5.5. We plot the absolute errors $\log_2(|Y_0 - y_0^0|)$ with respect to $\log_2(N_T)$ in Figure 5.5.

TABLE 5.5
Errors and convergence rates.

	$\lvert Y_0 - y_0^0\rvert$						CR
N_T	2	4	8	16	32	64	
N_E	400	1000	4000	10000	20000	40000	
EMC	0.1259	0.0847	0.0539	0.0332	0.0173	0.0073	0.82
HS	0.3383	0.1450	0.0446	0.0286	0.0141	0.0050	1.30
Scheme^L	0.0792	0.0406	0.0163	0.0047	0.0009	0.0001	1.89
Scheme^C	0.0895	0.0356	0.0143	0.0052	0.0013	0.0002	1.77
Scheme^S	0.0742	0.0242	0.0046	0.0022	0.0010	0.0002	1.71
	$\lvert Z_0 - z_0^0\rvert$						CR
N_T	2	4	8	16	32	64	
N_E	400	1000	4000	10000	20000	40000	
EMC	0.4382	0.7426	0.8235	0.4092	0.5211	0.4763	-0.02
HS	2.1918	1.0871	0.5013	0.4147	0.2729	0.1224	0.83
Scheme^L	0.8762	0.4042	0.2381	0.1069	0.0428	0.0105	1.28
Scheme^C	1.5382	0.7320	0.3647	0.2201	0.1052	0.0387	1.06
Scheme^S	1.8993	1.0830	0.4247	0.1835	0.0859	0.0322	1.17

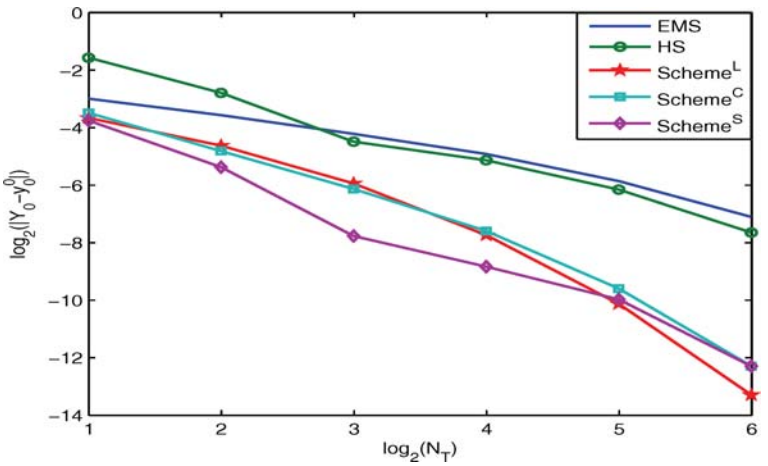


FIG. 5.5. *The plots of* $\log_2(|Y_0 - y_0^0|)$ *with respect to* $\log_2(N_T)$.

We summarize all numerical experiments as follows:

1. The accuracy of our method depends on the following:
   (a) Parameters θ_n^1 and θ_n^2. The accuracy of our method depends on the parameters θ_n^1 and θ_n^2, and if we choose them properly, the accuracy of our method can be high. Our numerical results show that the accuracy of our method is high when we choose $\theta_n^1 = \theta_n^2 = 0.5$.
   (b) Random sampling number N_E. It is known that the approximation error of the Monte Carlo method is approximately $\frac{1}{\sqrt{N_E}}$. Theoretically, if we want to obtain the accuracy ϵ, the random sampling number should approximately be $\left(\frac{1}{\epsilon}\right)^2$. For instance, when $\epsilon = (\Delta t)^2 = (\frac{T}{64})^2$, the sampling number is approximately $(64)^4$, which is a very big integer. In our simulations, for the time step $\Delta t = \frac{T}{64}$, the sampling number we used is 40000, which is much less than $(64)^4$. Even though we use the random samples much less than those needed theoretically, the accuracy

of our method is still high for solving (Y_t, Z_t) with $\theta_n^1 = \theta_n^2 = 0.5$.

(c) Space interpolations. In our method, we use the Monte Carlo method to approximate the conditional mathematical expectations and the space interpolations to compute the values at nongrid points. The accuracy of our method in space depends on the space interpolation operators and the regularity of the solutions. Our numerical results show that the higher the accuracy of the interpolation operators is, the higher the accuracy of our method is. The equidistant grid is far from optimal, especially for nonsmooth problems. To solve practical problems, some adaptive methods and some more efficient special grid interpolation operators may be used. All these will be our study projects on numerical methods for BSDEs in our future works.

(d) The regularity of the functions $f(t, y, z)$ and $\Phi(x)$. From the relation between BSDEs and PDEs [28], we know $Y_t = V(t, X_t)$ and $Z_t = \frac{\partial V(t,X_t)}{\partial x}$, where $V(t, x)$ is the solution of the PDEs, X_t is W_t in Examples 1–3, and $X_t = S_t$ in Example 4. The regularity of the solution (Y_t, Z_t) of the BSDE (2.1) depends on the regularity of the functions $f(t, y, z)$ and $\Phi(x)$. From PDE theory [31, 13], we know that the solutions of Examples 2 and 3 have smooth fourth order derivatives, and in Example 4, the solution Y_t has continuous second order derivatives with respect to S_t, and Z_t has only continuous first order derivatives. Our numerical results in Example 4 show that for nonsmooth data, the high-order interpolations cannot improve the approximate accuracy.

2. By comparison, we also solve the BSDEs in the examples by using Heun's method, in which the cubic polynomial interpolation is used in space. It is well known that Heun's method is a second order method for solving ODEs. But when used to solve the BSDEs, its accuracy is at most first order from our numerical results.

It is well known that the accuracy of the Monte Carlo method depends not only on the sampling number but also on the random number generator. We have tried several random number generators in C, Fortran, MATLAB, and the software SPRNG which is free and can be downloaded from the website http://sprng.cs.fsu.edu.

We found from our numerical experiments that the accuracy of our method depends on the random number generators, especially for solving Z_t. The high order accuracy was obtained for solving Y_t by using all the random number generators mentioned above in our numerical examples, but for solving Z_t, only by using the SPRNG random number generator was high order accuracy obtained. Since, in this paper, our aim is on the accuracy of our method, we list only the numerical results for EMC, HS and $Scheme^{\alpha}(\alpha = L, C, S)$ by using the software SPRNG to generate the random numbers. Better results may be obtained if some other better random number generators are used.

6. Conclusions. In this paper, we have proposed a new accurate numerical method to solve BSDEs at time-space grids. We use the Monte Carlo method to approximate the conditional mathematical expectations, and use local space interpolation to approximate the values of the solutions at nongrid points. Our numerical method is uncoupled not only between y_t and z_t, but also among the space grids at each time level. The data used to simulate the numerical solutions at each time-space point can be local. At the same time, the accuracy of our method can be improved

by better choosing the space interpolation operator and the parameters θ_1^n and θ_2^n. So there are large capacities of our method to solve the high-dimensional BSDEs with high accuracy by using parallel computers. To demonstrate the robustness of our methods, we performed some numerical experiments, and the results showed that our method is very effective and accurate for solving the BSDEs.

Acknowledgment. The authors would like to thank the referees for their valuable suggestions which greatly improved the paper.

REFERENCES

[1] F. Antonelli and A. Kohatsu-Higa, *Filtration stability of backward SDEs,* Stochastic Anal. Appl., 18 (2000), pp. 11–37.
[2] V. Bally, *Approximation scheme for solutions of BSDE*, in Backward Stochastic Differential Equations, Pitman Res. Notes Math. Ser. 364, Longman, Harlow, UK, 1997, pp. 177–191.
[3] V. Bally and G. Pagès, *A quantization algorithm for solving discrete time multidimensional optimal stopping problems*, Bernoulli, 6 (2003), pp. 1–47.
[4] V. Bally and G. Pagès, *Error analysis of the optimal quantization algorithm for obstacle problems*, Stochastic Process. Appl., 106 (2003), pp. 1–40.
[5] R. Biscay, J. C. Jimenez, J. J. Riera, and P. A. Valdes, *Local linearization method for the numerical solution of stochastic differential equations*, Annals Inst. Statist. Math., 48 (1996), pp. 631–644.
[6] B. Bouchard and N. Touzi, *Discrete-time approximation and Monte-Carlo simulation of backward stochastic differential equations*, Stochastic Process. Appl., 111 (2004), pp. 175–206.
[7] G. Box and M. Muller, *A note on the generation of random normal deviates*, Annals of Mathematical Statistics, 29 (1958), pp. 610–611.
[8] J .C. Butcher, *Numerical Methods for Ordinary Differential Equations*, John Wiley & Sons, New York, 2003.
[9] D. Chevance, *Numerical Methods for Backward Stochastic Differential Equations*, in Numerical Methods in Finance, L. C. G. Rogers and D. Talay, eds., Cambridge University Press, Cambridge, UK, 1997, pp. 232–244.
[10] G. C. Chow, *Dynamic Economics: Optimization by the Lagrange Method*, Oxford University Press, Oxford, UK, 1997.
[11] F. Coquet, V. Mackevicius, and J. Mémin, *Stability in D of martingales and backward equations under discretization of filtration*, Stochastic Process. Appl., 75 (1998), pp. 235–248.
[12] J. Douglas, Jr., Jin Ma, and P. Protter, *Numerical methods for forward-backward stochastic differential equations*, Ann. Appl. Prob., 6 (1996), pp. 940–968.
[13] L. C. Evans, *Partial Differential Equations*, AMS, Providence, RI, 2002.
[14] E. Fournié, J. M. Lasry, J. Lebuchoux, and P. L. Lions, *Applications of Malliavin calculus to Monte Carlo methods in finance* II, Finance Stoch., 5 (2001), pp. 201–236.
[15] E. Fournié, J. M. Lasry, J. Lebuchoux, and P. L. Lions, *Applications of Malliavin calculus to Monte Carlo methods in finance*, Finance Stoch., 3 (1999), pp. 391–412.
[16] N. Ikeda and S. Watanabe, *Stochastic Differential Equations and Diffusion Processes*, North–Holland, Amsterdam, 1981.
[17] N. El Karoui, C. Kapoudjan, E. Pardoux, S. Peng, and M. C. Quenez, *Reflected solutions of backward stochastic differential equations and related obstacle problems for PDEs*, Ann. Probab., 25 (1997), pp. 702–737.
[18] N. El Karoui, S. Peng, and M. C. Quenez, *Backward stochastic differential equations in finance*, Math. Finance, 7 (1997), pp. 1–71.
[19] P. E. Kloeden, *The systematic derivation of higher order numerical schemes for stochastic differential equations*, Milan J. Math., 70 (2002), pp. 187–207.
[20] P. E. Kloeden and E. Platen, *Numerical Solution of Stochastic Differential Equations*, 3rd printing, Springer-Verlag, Berlin, 1999.
[21] P. E. Kloeden, E. Platen, and H. Schurz, *Numerical Solution of SDE Through Computer Experiments*, 2nd printing, Springer-Verlag, Berlin, 1997.
[22] F. A. Longstaff and R. S. Schwartz, *Valuing American options by simulation: A simple least-square approach*, Review of Financial Studies, 14 (2001), pp. 113–147.

[23] J. Ma, P. Protter, J. San Martin, and S. Torres, *Numerical method for backward stochastic differential equations*, Ann. Appl. Probab., 12 (2000), pp. 302–316.
[24] J. Ma, P. Protter, and J. Yong, *Solving forward-backward stochastic differential equations explicitly—a four step scheme*, Probab. Theory Related Fields, 98 (1994), pp. 339–359.
[25] J. Ma and J. Yong, *Forward-Backward Stochastic Differential Equations and Their Applications*, Lecture Notes in Math. 1702, Springer-Verlag, New York, 1999.
[26] E. Pardoux and S. G. Peng, *Adapted solution of a backward stochastic differential equation*, Systems Control Lett., 14 (1990), pp. 55–61.
[27] S. Peng, *A general stochastic maximum principle for optimal control problems*, SIAM J. Control Optim., 28 (1990), pp. 966–979.
[28] S. Peng, *Probabilistic interpretation for systems of quasilinear parabolic partial differential equations*, Stochastics Stochastics Rep., 37 (1991), pp. 61–74.
[29] S. Peng, *A generalized dynamic programming principle and Hamilton-Jacobi-Bellman equation*, Stochastics Stochastics Rep., 38 (1991), pp. 119–134.
[30] S. Peng, *A linear approximation algorithm using BSDE*, Pacific Economic Review, 4 (1999), pp. 285–291.
[31] G. Da Prato and J. Zabczyk, *Second Order Partial Differential Equations in Hilbert Spaces*, Cambridge University Press, Cambridge, UK, 2002.
[32] R. Khasminskii and G.N. Milstein, *On estimation of the linearized drift for nonlinear stochastic differential equations*, Stoch. Dyn., 1 (2001), pp. 23–43.
[33] J. Zhang, *Some Fine Properties of Backward Stochastic Differential Equations*, Ph.D. thesis, Purdue University, West Lafayette, IN, 2001.
[34] J. Zhang, *A numerical scheme for BSDEs*, Ann. Appl. Probab. 14 (2004), pp. 459–488.
[35] Y. Zhang and W. Zheng, *Discretizing a backward stochastic differential equation*, Int. J. Math. Math. Sci., 32 (2002), pp. 103–116.

SIAM J. SCI. COMPUT.
Vol. 28, No. 4, pp. 1582–1596
© 2006 Society for Industrial and Applied Mathematics

DISCRETE-VELOCITY MODELS AND RELAXATION SCHEMES FOR TRAFFIC FLOWS*

MICHAEL HERTY†, LORENZO PARESCHI‡, AND MOHAMMED SEAID†

Abstract. We present simple discrete-velocity models for traffic flows. The novel feature in the corresponding relaxation system is the presence of nonnegative velocities only. We show that in the small relaxation limit the discrete models reduce to the Lighthill–Whitham–Richards equation. In addition we propose second order schemes combined with implicit-explicit time integrators as proper discretization of the relaxation-type system. Numerical tests are carried out on various situations in traffic flow. The results show that the proposed models are able to describe correctly the formation of backward waves induced by traffic jam.

Key words. discrete-velocity models, relaxation schemes, IMEX schemes, ENO schemes, Lighthill–Whitham–Richards equation

AMS subject classifications. 65M06, 35L60, 82B80

DOI. 10.1137/04061982X

1. Introduction.

Macroscopic modeling of vehicular traffic started with the work of Lighthill and Whitham [12]. They considered the continuity equation for the density ρ and approximated the mean velocity u by an equilibrium value $u^e(\rho)$:

$$\partial_t \rho + \partial_x(\rho u^e(\rho)) = 0. \tag{1.1}$$

The function $u^e(\rho)$ is also called a fundamental diagram, and $\rho u^e(\rho)$ is the flux. For a comparison with experimental data we refer the reader to [13]. More realistic effects are studied, for example, in [4], and among the models trying to include those effects, there are the "higher order" models by, e.g., [19, 3, 1].

In this paper we focus on simple discrete-velocity models of traffic flows which share the common properties that in the small relaxation limit reduce to a macroscopic description of traffic flow by the single conservation law (1.1).

In [8], Jin and Xin proposed the relaxation system

$$\begin{aligned} &\partial_t \rho + \partial_x q = 0, \\ &\partial_t q + a^2 \partial_x \rho = -\frac{1}{\varepsilon}\Big(q - F(\rho)\Big) \end{aligned} \tag{1.2}$$

as an approximation to the scalar hyperbolic equation

$$\partial_t \rho + \partial_x F(\rho) = 0. \tag{1.3}$$

Formally, solution of the relaxation system (1.2) approximates solution of the original equation (1.3) by order $\mathcal{O}(\varepsilon)$ if the constant a satisfies the so-called [illegible] condition [8, 7, 2, 14]

$$a^2 - F'(\rho)^2 \geq 0 \qquad \forall\ \rho, \tag{1.4}$$

*Received by the editors November 29, 2004; accepted for publication (in revised form) April 25, 2006; published electronically September 21, 2006. This research was supported by HYKE network of the European Union, contract HPRN-CT-2002-00282.

http://www.siam.org/journals/sisc/28-4/61982.html

†FB Mathematik, AG8, TU Darmstadt, 64289 Germany (herty@mathematik.tu-darmstadt.de, seaid@mathematik.tu-darmstadt.de).

‡Department of Mathematics, University of Ferrara, Ferrara, 44100 Italy (pareschi@dm.unife.it).

which means that the kinetic wave speed $F'(\rho)$ is between the two characteristic speeds. At a numerical level, the main advantage in considering a relaxation system is the fact that the nonlinear conservation law (1.3) is replaced by a semilinear system (1.2) with linear characteristic variables given by $q \pm a\rho$. Consequently, nonlinear Riemann problems, characteristic decompositions, and linear iterations are avoided in its numerical solution. We refer the reader to [10, 8, 6, 20] and references therein. All of these methods are easy to formulate and implement. Moreover, there are strong links between relaxation methods and central schemes; see, for instance, [22, 16]. Of course, for scalar conservation laws there also exist other methods using exact or approximate Riemann solvers; see, for example, [25, 26, 27].

The relaxation system (1.2) is strongly connected to discrete-velocity models in kinetic theory. In fact, it can be reformulated by considering "kinetic" variables f and g as follows:

$$\begin{aligned} \partial_t f + a\partial_x f &= -\frac{1}{\varepsilon}\left(f - \frac{1}{2}\left(\rho + \frac{q}{a}\right)\right), \\ \partial_t g - a\partial_x g &= -\frac{1}{\varepsilon}\left(g - \frac{1}{2}\left(\rho - \frac{q}{a}\right)\right), \end{aligned} \tag{1.5}$$

where $\rho = f + g$ and $q = a(f - g)$. The equations (1.5) can be viewed as an evolution system of particles with speeds $\pm a$ and interaction terms given by the right-hand side; compare [15, 9, 17].

Note that in (1.5) the kinetic variables f and g propagate with positive and negative speed, respectively. This fits well with the fact that gas particles can move in either direction. However, if we restrict our discussion to a situation where all particles have the same (nonnegative) propagation direction, then the above formulation will not describe the correct dynamics of the relaxed flow. If we consider the case of particles travelling with speeds $b > a \geq 0$, easy computations lead to the subcharacteristic condition $(F'(\rho) - a)\,(b - F'(\rho)) \geq 0$, which is not satisfied unless $a \leq F'(\rho) \leq b$. A physical field where such a situation takes place is the case of traffic flow models where cars can have only positive velocities, but a traffic jam also moves backwards. Therefore, it is our goal here to derive simple relaxation-type models with a clear "kinetic" interpretation since we allow cars to propagate only with nonnegative speeds but are still able to capture the backward propagation of the solution.

In the following section, we present the traffic flow equations used in the paper to develop a relaxation-type model. In section 3, the relaxation scheme for the model problem is detailed. This includes the space and time discretizations. The numerical results are presented in section 4. We apply our scheme to common situations like free flow, bottleneck, and traffic jam. In section 5, some concluding remarks are listed.

2. Discrete-velocity models of traffic flows. In traffic flow theory the time evolution of a vehicle density in a freeway is governed by the Lighthill–Whitham–Richards model [12]

$$\partial_t \rho + \partial_x V(\rho) = 0, \tag{2.1}$$

where $\rho(t, x)$ is the vehicle density at location x and time t, and $V(\rho)$ is the flow function given by

$$V(\rho) = \rho \max\{u_0, u_e(\rho)\}, \tag{2.2}$$

where $u_e(\rho)$ is the fundamental diagram, i.e., an equilibrium velocity, and u_0 is the minimum speed. A typical choice for $u_e(\rho)$ is, for example,

$$u_e(\rho) = u_m \left(1 - \frac{\rho}{\rho_m}\right), \tag{2.3}$$

where ρ_m and u_m are the maximum density and the maximum speed, respectively.

Remark 1. More general flux functions as well as general traffic flow models have been considered in the literature; see [1, 4]. For inhomogeneous traffic flow situations we may have an additional dependence on x, i.e., $V(\rho) = V(x, \rho)$.

2.1. Two-speed models. Let us consider the following simple discrete-velocity model:

$$\partial_t f + v_1 \partial_x f = -\frac{K(\rho)}{\varepsilon}\Big(v_1 f + v_2 g - V(\rho)\Big), \tag{2.4}$$

$$\partial_t g + v_2 \partial_x g = \frac{K(\rho)}{\varepsilon}\Big(v_1 f + v_2 g - V(\rho)\Big), \tag{2.5}$$

where $K(\rho)$ is a suitable function such that

$$\begin{cases} K(\rho) \geq 0 & \text{if} \quad \rho \leq \rho_*, \\ K(\rho) < 0 & \text{if} \quad \rho > \rho_*, \end{cases}$$

where $0 < \rho_* < \rho_m$ is a constant.

The system (2.4)–(2.5) offers the following interpretation. We have two types of cars driving with velocity $v_2 > v_1 \geq 0$. The functions f and g are probability distributions of cars with speeds v_1 and v_2, respectively. Depending on the density, $K(\rho)$ changes sign. Usually in traffic flow models, the flux function $V : \rho \to \rho u_e(\rho)$ is a concave function. Hence $K(\rho) \geq 0$ in low density regimes, whereas $K(\rho)$ should become large and negative in high density regimes so that there is a gain for f and a loss for g by the interaction terms. Therefore, more cars will attain the speed v_1. This fits well with the observation that in areas of high densities, the cars tend to slow down. The opposite phenomena is also governed by (2.4)–(2.5).

In what follows we assume that

$$K(\rho) = H\left(V'(\rho) - v_1\right), \tag{2.6}$$

where $H(\cdot)$ is a function that preserves the sign of the argument, for example, a Heaviside or a hyperbolic tangent function.

Introducing the density and the flux of cars defined as

$$\rho = f + g \qquad \text{and} \qquad J = v_1 f + v_2 g, \tag{2.7}$$

we obtain the following macroscopic system:

$$\partial_t \rho + \partial_x J = 0, \tag{2.8}$$

$$\partial_t J + (v_1 + v_2)\partial_x J - v_1 v_2 \partial_x \rho = -\frac{K(\rho)}{\varepsilon}\left(v_2 - v_1\right)\Big(J - V(\rho)\Big). \tag{2.9}$$

In the small relaxation limit ($\varepsilon \to 0$), equation (2.9) gives to leading order

$$J = V(\rho) + \frac{\varepsilon}{K(\rho)\left(v_2 - v_1\right)}\Big(\left(V'(\rho) - v_1\right)\left(V'(\rho) - v_2\right)\partial_x \rho\Big).$$

By using this variable in (2.8) we obtain

$$\partial_t \rho + \partial_x V(\rho) + \partial_x \left(\frac{\varepsilon}{K(\rho)\,(v_2 - v_1)} \left(V'(\rho) - v_1\right)\left(V'(\rho) - v_2\right) \partial_x \rho \right) = 0. \tag{2.10}$$

To ensure the dissipative nature of (2.10), it is necessary that

$$\frac{\left(V'(\rho) - v_1\right)\left(V'(\rho) - v_2\right)}{K(\rho)\,(v_2 - v_1)} \leq 0. \tag{2.11}$$

This condition is satisfied by the assumption (2.6) on the kernel $K(\rho)$ if $v_1 \leq u_m \leq v_2$. Formally, when $\varepsilon \to 0$, the relaxation system (2.8)–(2.9) converges to the original equation (2.1) provided the [illegible] condition (2.11) is satisfied.

Note that the equilibrium states for f and g can be obtained by the equalities $f + g = \rho$ and $v_1 f + v_2 g = J$ and are given by

$$E_f(\rho) = \frac{\rho}{v_2 - v_1}\left(v_2 - \frac{V(\rho)}{\rho}\right), \qquad E_g(\rho) = \frac{\rho}{v_2 - v_1}\left(\frac{V(\rho)}{\rho} - v_1\right). \tag{2.12}$$

Nonnegativity of the equilibrium states is guaranteed if

$$v_1 \leq \frac{V(\rho)}{\rho} \leq v_2, \tag{2.13}$$

which is satisfied under the natural requirements $u_0 \geq v_1$ and $u_m \leq v_2$.

[illegible] 2. It is noteworthy that when the density in the system becomes critical, i.e., $\rho \to \rho_m$, a desirable feature would be that all cars reduce their speed to the lowest speed v_1 without allowing $f + g > \rho_m$. This behavior is obtained naturally for small values of ε in agreement with the kinetic interpretation of ε as the mean free path between car particles.

As an alternative, in order to have this feature even for large values of ε we can ask that $K(\rho) \to -\infty$ as $\rho \to \rho_m$ to recover the critical equilibrium states

$$f = \frac{\rho_m}{v_2 - v_1}\left(v_2 - u_0\right), \qquad g = \frac{\rho_m}{v_2 - v_1}\left(u_0 - v_1\right), \tag{2.14}$$

which correspond to a traffic jam situation. Clearly by taking $u_0 = v_1$ we have $f = \rho_m$ and $g = 0$.

[illegible] 3. Similarly one can derive discrete-velocity models with speeds $v_i, i = 1, 2, \ldots, n$ and kinetic variables f_i. For example, if one considers the Aw–Rascle model [1] written in conservative form as

$$\partial_t \rho + \partial_x \left(\rho v(\rho, y)\right) = 0, \tag{2.15}$$

$$\partial_t y + \partial_x \left(y v(\rho, y)\right) = 0, \tag{2.16}$$

where $v(\rho, y) = (y/\rho - p(\rho))$, ρ is the total density of cars and y is the density of cars with a given "quality" (typically a subset of the total number of cars). The pressure term $p(\rho)$ is an anticipation factor that takes into account drivers' reactions to the state of traffic in front of them.

We can consider the four-speed kinetic model

$$\partial_t f + v_1 \partial_x f = -\frac{K(\rho,y)}{\varepsilon}\Big(v_1 f + v_2 g - v(\rho,y)(f+g)\Big), \tag{2.17}$$

$$\partial_t g + v_2 \partial_x g = \frac{K(\rho,y)}{\varepsilon}\Big(v_1 f + v_2 g - v(\rho,y)(f+g)\Big), \tag{2.18}$$

$$\partial_t h + v_3 \partial_x h = -\frac{K(\rho,y)}{\varepsilon}\Big(v_3 h + v_4 k - v(\rho,y)(h+k)\Big), \tag{2.19}$$

$$\partial_t k + v_4 \partial_x k = \frac{K(\rho,y)}{\varepsilon}\Big(v_3 h + v_4 k - v(\rho,y)(h+k)\Big), \tag{2.20}$$

with $v_4 > v_3 \geq 0$ and $v_2 > v_1 \geq 0$, which in macroscopic form reads

$$\partial_t \rho + \partial_x J = 0, \tag{2.21}$$

$$\partial_t y + \partial_x W = 0, \tag{2.22}$$

$$\partial_t J + (v_1+v_2)\partial_x J - v_1 v_2 \partial_x \rho = -\frac{K(\rho,y)}{\varepsilon}(v_2 - v_1)\Big(J - \rho v(\rho,y)\Big), \tag{2.23}$$

$$\partial_t W + (v_3+v_4)\partial_x W - v_3 v_4 \partial_x y = -\frac{K(\rho,y)}{\varepsilon}(v_4 - v_3)\Big(W - y v(\rho,y)\Big), \tag{2.24}$$

where

$$\rho = f + g, \qquad y = h + k, \qquad \text{and} \qquad J = v_1 f + v_2 g, \qquad W = v_3 h + v_4 k. \tag{2.25}$$

Suitable conditions must be imposed on $K(\rho, y)$ in order to satisfy the subcharacteristic condition as $\varepsilon \to 0$.

3. Relaxation schemes. We present a numerical method of approximating the discrete-velocity models in such a way that the resulting scheme is asymptotic preserving (AP) and high order accurate in space and time (we refer to [16] for more details on AP schemes).

The numerical method we construct is based on the method of lines approach. We use a high order central weighted essentially nonoscillatory (WENO) scheme as space discretization of the kinetic system. The necessity of using a central scheme comes from the fact that the limiting equation (2.1) allows solutions $\rho(t,x)$ which can propagate with either positive or negative speed (see the appendix). For the time integration we apply a third order implicit-explicit (IMEX) Runge–Kutta method.

For simplicity we will describe the numerical schemes for the two-speed discrete-velocity model. The extension to the three-speed model is straightforward.

3.1. Space discretization. Let the spatial interval be discretized in equally spaced grids with stepsize Δx, and let w_i denote the point-value of an arbitrary function w at gridpoint x_i. Then a semidiscretization of (2.4) can be written as

$$\frac{df_i}{dt} + v_1 \frac{f_{i+\frac{1}{2}} - f_{i-\frac{1}{2}}}{\Delta x} = G_i^+. \tag{3.1}$$

For simplicity in presentation and to show how a space discretization can be reconstructed we consider only the Lax–Friedrichs method. Extension to higher order reconstructions can be developed using a similar formulation. Thus, the first order Lax–Friedrichs numerical flux $f_{i+\frac{1}{2}}$ is given by

$$f_{i+\frac{1}{2}} = \frac{1}{2}\left(f_{i+1} + f_i - \frac{\Delta x}{v_1 \Delta t}(f_{i+1} - f_i)\right). \tag{3.2}$$

Similarly, the semidiscretization of (2.5) can be written as

$$\frac{dg_i}{dt} + v_2 \frac{g_{i+\frac{1}{2}} - g_{i-\frac{1}{2}}}{\Delta x} = G_i^-, \tag{3.3}$$

where $g_{i+\frac{1}{2}}$ is defined analogously by

$$g_{i+\frac{1}{2}} = \frac{1}{2}\left(g_{i+1} + g_i - \frac{\Delta x}{v_2 \Delta t}\,(g_{i+1} - g_i)\right). \tag{3.4}$$

The source terms $G_i^{\pm}$ are given by

$$G_i^{\pm} = \pm \frac{K(\rho_i)}{2\varepsilon}\Big(v_1 f_i + v_2 g_i - V(\rho_i)\Big). \tag{3.5}$$

Using (2.12) in (3.1) and (3.3) gives a first order semidiscretization of the relaxation system (2.8)–(2.9):

$$\begin{aligned} &\frac{d\rho_i}{dt} + \frac{J_{i+\frac{1}{2}} - J_{i-\frac{1}{2}}}{\Delta x} = 0, \\ &\frac{dJ_i}{dt} + (v_1 + v_2)\frac{J_{i+\frac{1}{2}} - J_{i-\frac{1}{2}}}{\Delta x} - v_1 v_2 \frac{\rho_{i+\frac{1}{2}} - \rho_{i-\frac{1}{2}}}{\Delta x} = -\frac{K(\rho_i)}{2\varepsilon}\,(v_2 - v_1)\Big(J_i - V(\rho_i)\Big), \end{aligned} \tag{3.6}$$

where

$$\rho_{i+\frac{1}{2}} = f_{i+\frac{1}{2}} + g_{i+\frac{1}{2}} \qquad \text{and} \qquad J_{i+\frac{1}{2}} = v_1 f_{i+\frac{1}{2}} + v_2 g_{i+\frac{1}{2}},$$

with $f_{i+\frac{1}{2}}$ and $g_{i+\frac{1}{2}}$ given by (3.2) and (3.4), respectively.

Next, we discuss some properties of the semidiscrete first order relaxation scheme. We split the scheme (3.1)–(3.3) and give additional conditions on $|K(\rho)|$, which guarantee positivity. We consider the system

$$\partial_t f_i = G_i^+ = \frac{K(\rho)(v_2 - v_1)}{\varepsilon}\,(E_f(\rho_i) - f_i)\,, \tag{3.7}$$

$$\partial_t g_i = G_i^- = -\frac{K(\rho)(v_2 - v_1)}{\varepsilon}\,(E_g(\rho_i) - g_i) \tag{3.8}$$

for some initial data $f_0, g_0 \geq 0$. If $f_0 \geq E_f(\rho_i)$, then $K(\rho)(E_f - f)$ in (3.7) acts as reaction term and $f_i(t) \to E_f(\rho_i) \geq 0$. For $f_0 < E_f(\rho_i)$, the solution $f(t)$ to (3.7) is monotonic decreasing. In fact $f(t) = \exp(-K(\rho_i)(v_2 - v_1)t/\varepsilon)\,(f_0 - E_f(\rho_i)) + E_f(\rho_i)$. Hence, if $K(\rho_i)$ satisfies the assumption

$$|K(\rho_i)| \leq \frac{\varepsilon}{\Delta t (v_2 - v_1)} \max\left\{\frac{E_f(\rho_i)}{E_f(\rho_i) - f_0}, \frac{E_g(\rho_i)}{E_g(\rho_i) - g_0}\right\}, \tag{3.9}$$

then $f(t)$ and $g(t)$ remain nonnegative for all $t \in (0, \Delta t)$. This gives the positivity of the semidiscrete scheme (3.1)–(3.3). The positivity is inherited by the relaxed scheme in the limit $\varepsilon \to 0$; see below.

Higher order spatial discretizations can be derived following the essentially nonoscillatory (ENO) reconstructions; compare [21, 22, 11] among others. For instance, a second order reconstruction is obtained by substituting the numerical fluxes $f_{i+\frac{1}{2}}$ and $g_{i+\frac{1}{2}}$ using the convex ENO scheme [11]. Thus,

$$f_{i+\frac{1}{2}} = R\left(v_1, f_{i+\frac{1}{2}}\right) \qquad \text{and} \qquad g_{i+\frac{1}{2}} = R\left(v_2, g_{i+\frac{1}{2}}\right), \tag{3.10}$$

where the reconstruction flux function R is defined as

$$
\begin{aligned}
R\left(\alpha, w_{i+\frac{1}{2}}\right) = & \frac{1}{2}\left(w_{i+1} + w_i - \frac{\Delta x}{\alpha \Delta t}\left(w_{i+1} - w_i\right)\right) \\
& + \frac{1}{4}\left(w_{i+1} - w_i + \frac{\Delta x}{\alpha \Delta t}\left(w_{i+1} - w_i\right)\right) \phi\left(\theta_+^\alpha\right) \\
& - \frac{1}{4}\left(w_{i+2} - w_{i+1} - \frac{\Delta x}{\alpha \Delta t}\left(w_{i+2} - w_{i+1}\right)\right) \phi\left(\theta_-^\alpha\right),
\end{aligned} \tag{3.11}
$$

with the slopes θ_+^α and θ_-^α given by

$$
\theta_+^\alpha = \frac{w_i - w_{i-1} + \frac{\Delta x}{\alpha \Delta t}\left(w_i - w_{i-1}\right)}{w_{i+1} - w_i + \frac{\Delta x}{\alpha \Delta t}\left(w_{i+1} - w_i\right)}, \qquad \theta_-^\alpha = \frac{w_{i+1} - w_i - \frac{\Delta x}{\alpha \Delta t}\left(w_{i+1} - w_i\right)}{w_{i+2} - w_{i+1} - \frac{\Delta x}{\alpha \Delta t}\left(w_{i+2} - w_{i+1}\right)},
$$

and ϕ is a slope limiter function. A simple choice is the minmod limiter

$$
\phi(\theta) = \max\bigl(0, \min\left(1, \theta\right)\bigr).
$$

Note that if we set $\phi = 0$ in (3.11), the reconstruction (3.10) reduces to the first order Lax–Friedrichs discretization (3.2) and (3.4). We should mention that any discretization that requires characteristics information will generate instabilities in the computed solution. This is due to the structure of the relaxation system which allows propagation along the characteristics in one direction only.

3.2. Time integration. The full discretization of system (2.8)–(2.9) can be obtained by the well-established IMEX methods; see, for instance [16, 5]. The method applied here is the third order TVD time splitting scheme of [10]. The details can be found in the cited reference. We only mention that the special structure of the nonlinear terms in (3.6) makes it trivial to evolve the flux terms explicitly and the source term implicitly. Neither linear algebraic equations nor nonlinear source terms arise. In addition, the obtained relaxation scheme is stable independent of ε provided that the usual CFL condition

$$
\mathrm{CFL} = \max\left\{v_1, v_2\right\} \frac{\Delta t}{\Delta x} \leq 1 \tag{3.12}
$$

holds.

3.3. Relaxed schemes. Using a Hilbert expansion for (3.6) and neglecting terms of higher order in ε, we obtain

$$
J_i = V(\rho_i)
$$

and

$$
J_{i+\frac{1}{2}} = \frac{1}{2}\left(V(\rho_{i+1}) + V(\rho_i) - \frac{\Delta x}{\Delta t}\bigl(V(\rho_{i+1}) - V(\rho_i)\bigr)\right).
$$

This yields a first order Lax–Friedrichs discretization of the flux for the semidiscrete scheme

$$
\frac{d\rho_i}{dt} + \frac{J_{i+\frac{1}{2}} - J_{i-\frac{1}{2}}}{\Delta x} = 0. \tag{3.13}
$$

Similarly, when $\varepsilon \longrightarrow 0$, the reconstruction (3.10) yields to the second order convex ENO scheme for the relaxed equation (3.13) with

$$\begin{aligned} J_{i+\frac{1}{2}} = & \frac{1}{2}\left(V(\rho_{i+1}) + V(\rho_i) - \frac{\Delta x}{\Delta t}\big(V(\rho_{i+1}) - V(\rho_i)\big)\right) \\ & + \frac{1}{4}\left(V(\rho_{i+1}) - V(\rho_i) + \frac{\Delta x}{\Delta t}\big(V(\rho_{i+1}) - V(\rho_i)\big)\right)\phi\left(\theta_+\right) \\ & - \frac{1}{4}\left(V(\rho_{i+2}) - V(\rho_{i+1}) - \frac{\Delta x}{\Delta t}\big(V(\rho_{i+2}) - V(\rho_{i+1})\big)\right)\phi\left(\theta_-\right), \end{aligned}$$

where

$$\theta_+ = \frac{V(\rho_i) - V(\rho_{i-1}) + \frac{\Delta x}{\Delta t}\left(V(\rho_i) - V(\rho_{i-1})\right)}{V(\rho_{i+1}) - V(\rho_i) + \frac{\Delta x}{\Delta t}\left(V(\rho_{i+1}) - V(\rho_i)\right)},$$

$$\theta_- = \frac{V(\rho_{i+1}) - V(\rho_i) - \frac{\Delta x}{\Delta t}\left(V(\rho_{i+1}) - V(\rho_i)\right)}{V(\rho_{i+2}) - V(\rho_{i+1}) - \frac{\Delta x}{\Delta t}\left(V(\rho_{i+2}) - V(\rho_{i+1})\right)}.$$

For equilibrium initial data the leading behavior of the relaxation scheme is governed by the associated relaxed scheme. Hence, one can recover all physical properties of the full conservation law (2.1) by our relaxation-type system with positive velocities (2.4)–(2.5). In addition, at the limit when $\varepsilon \to 0$, the time integration converges to an explicit TVD Runge–Kutta scheme, and we refer to [10] for details.

In the relaxation limit ($\varepsilon \to 0$) the first order semidiscrete limit scheme is the Lax–Friedrichs scheme. Therefore, positivity, TVD, and the discrete entropy inequality are satisfied in the limit.

4. Numerical examples. In this section we illustrate the performance of our relaxation method for different test cases on traffic flow. Throughout the numerical tests the relaxation rate is $\varepsilon = 10^{-1}$ or $\varepsilon = 10^{-10}$ and the time step Δt is chosen according to the CFL condition (3.12). The kernel $K(\rho)$ and the characteristic speeds v_1 and v_2 are chosen as

$$K(\rho) = \tanh\left(5\left(V'(\rho) - v_1\right)\right), \qquad v_1 = 0.1, \qquad v_2 = 1 \tag{4.1}$$

for the two-speed model. In all the results reported in this section for traffic flow, the spatial domain is the road interval $[0, 1]$ discretized into 200 gridpoints, and a CFL $= 0.5$ is used. We display results for the two-speed model at $\varepsilon = 10^{-1}$ and $\varepsilon = 10^{-10}$. The following examples (subsections 4.1–4.3) are selected.

4.1. Free-flow traffic situation. We turn to the traffic flow model (2.1)–(2.3) in the simplest case where cars moves freely without jams. Here we assume that $\rho_m = u_m = 1$ and that initially the cars are normally distributed around the point $x = 0.4$ (Gaussian-like initial data). The propagation of the density in the space-time domain for $\varepsilon = 10^{-1}$ and $\varepsilon = 10^{-10}$ is given in Figure 1, whereas Figure 2 shows the contour lines of density ρ. As expected, the relaxation scheme evolves the free traffic in the correct way. In Figure 3 we plot the density at $t = 0.6$ for different relaxation limits. The asymptotic convergence of the solution as $\varepsilon \to 0$ is clearly preserved by our discretizations. All of these results are computed using the second order ENO reconstruction.

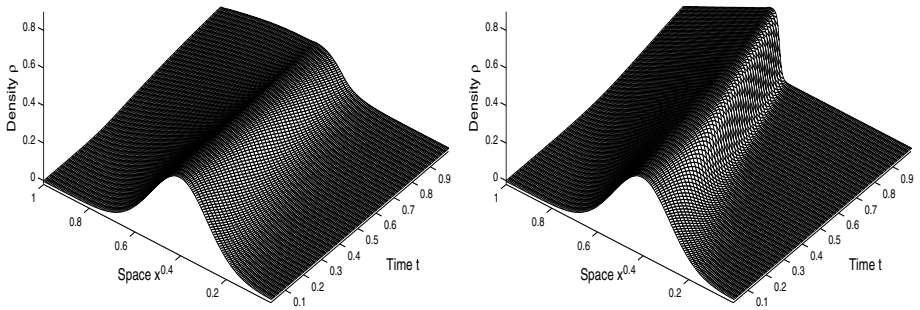


FIG. 1. *Density evolution for the free-flow traffic situation at* $\varepsilon = 10^{-1}$ *(left plot) and* $\varepsilon = 10^{-10}$ *(right plot).*

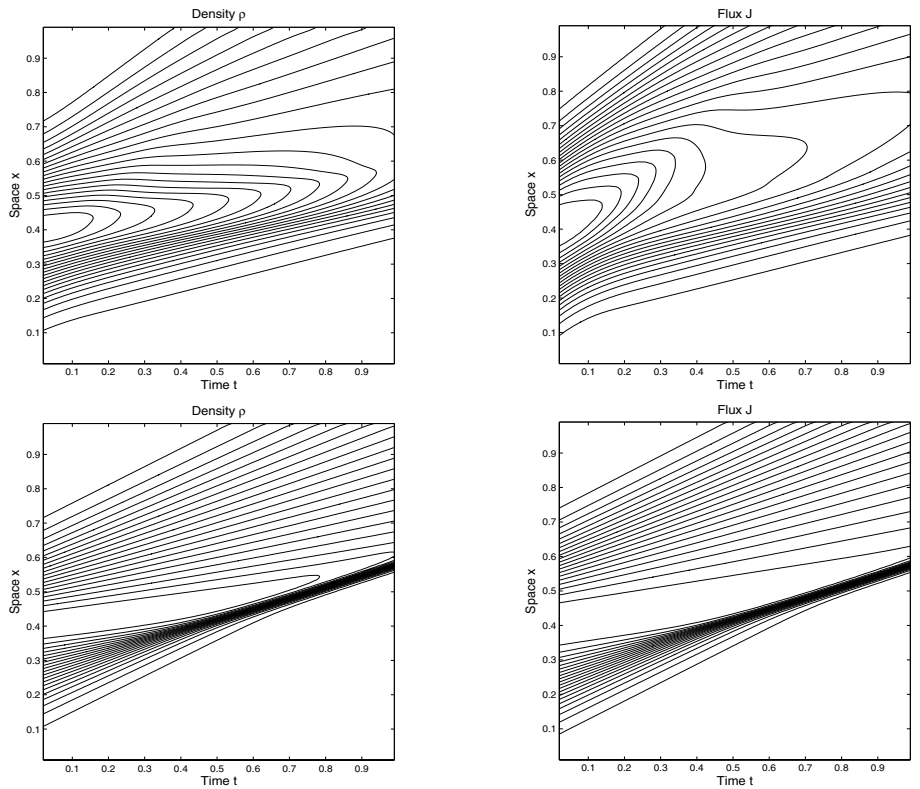


FIG. 2. *Contour plots of density* ρ *and flux* J *for the free-flow traffic situation at* $\varepsilon = 10^{-1}$ *(top) and* $\varepsilon = 10^{-10}$ *(bottom).*

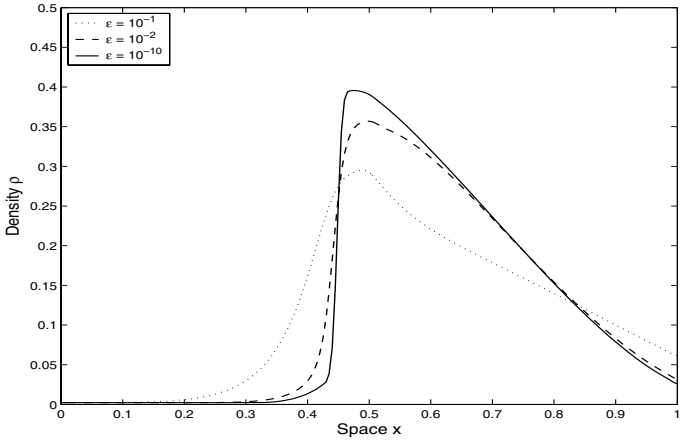


FIG. 3. *Plots of density for the free-flow traffic situation at* $t = 0.6$ *using different values of* ε.

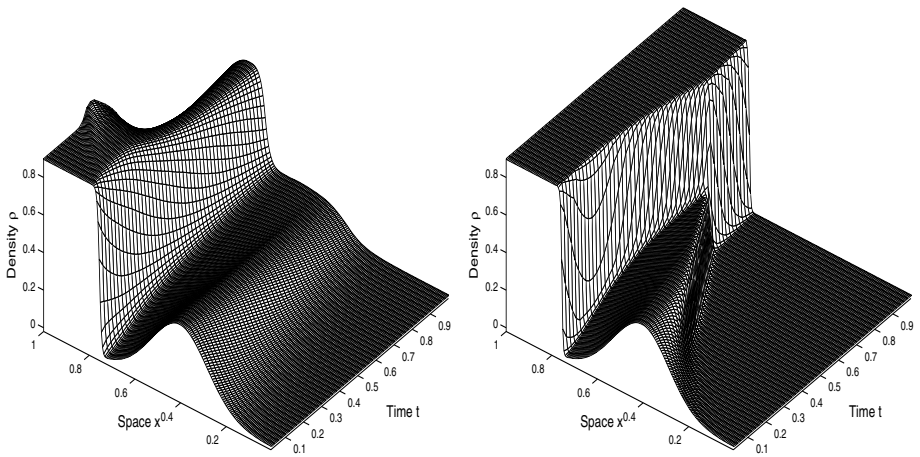


FIG. 4. *Density evolution for the jam traffic situation at* $\varepsilon = 10^{-1}$ *(left plot) and* $\varepsilon = 10^{-10}$ *(right plot).*

4.2. Jam situation. Next we consider a similar situation but with a traffic jam at $x = 0.75$. In realistic vehicular traffic flow this is equivalent to relaxed traffic upstream ($x < 0.75$) and crowded traffic downstream ($x \geq 0.75$). In such a situation one expects that the cars which are in a Gaussian-like distribution start moving into the jam, and that once they reach the jam a backwards wave should be formed. By examining the evolution of the density in Figure 4 for $\varepsilon = 10^{-10}$, we can observe that the mentioned physical features are well captured by our relaxation scheme. The contour plots of density ρ and flux J are displayed in Figure 5. It is clear that the Gaussian density distribution is moving into the jam and then propagates backwards. In the two-speed model there is an overshoot of the density at the jam due to the large value of ε. This suggests taking ε small when the value of the density is close to critical.

All of these results are computed using the second order ENO reconstruction. Figure 6 represents a comparison of density plots at $t = 0.6$ using the first order

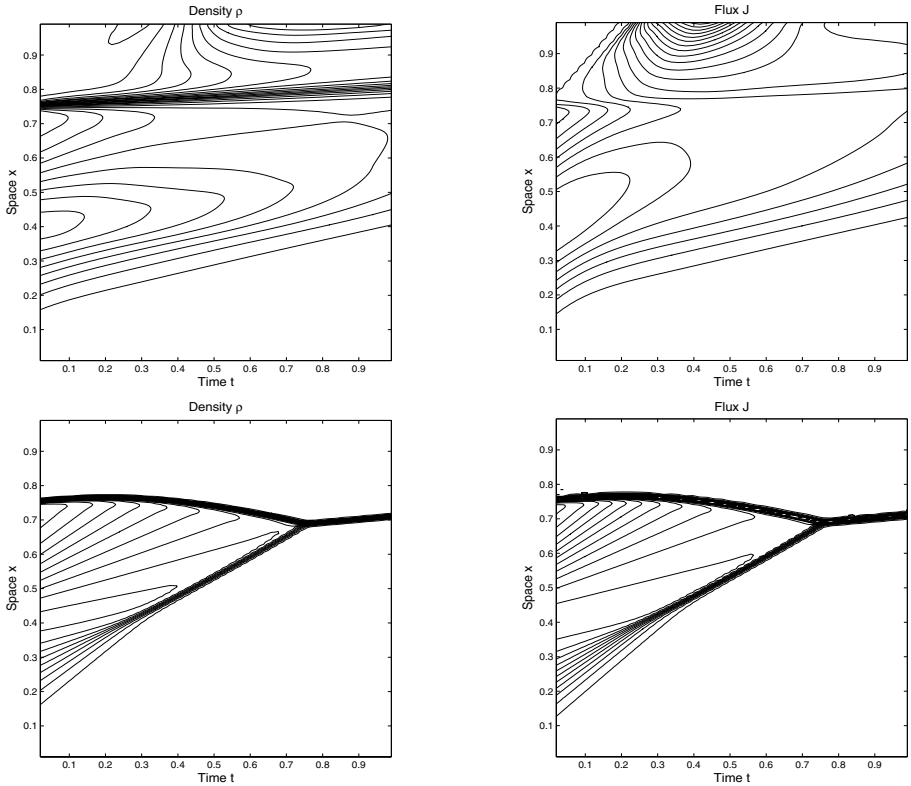


FIG. 5. *Contour plots of density ρ and flux J for the jam traffic situation at $\varepsilon = 10^{-1}$ (top) and $\varepsilon = 10^{-10}$ (bottom).*

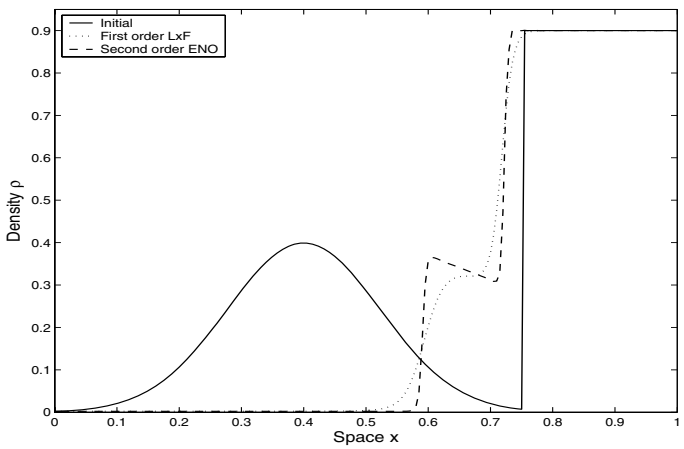


FIG. 6. *Plots of density for the jam traffic situation at $t = 0.6$ and $\varepsilon = 10^{-10}$ using the first order Lax–Friedrichs and the second order ENO discretizations.*

Lax–Friedrichs and the second order ENO discretizations. It is clear that excessive diffusion was introduced by the first order Lax–Friedrichs discretization, while the second order ENO reconstruction sharply captures the shock.

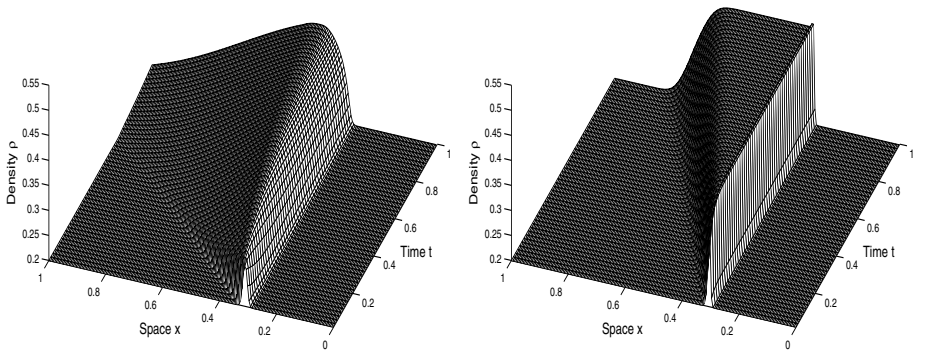


FIG. 7. *Density evolution for the bottleneck traffic situation with* $\bar{\rho} = 4$ *at* $\varepsilon = 10^{-1}$ *(left plot) and* $\varepsilon = 10^{-10}$ *(right plot).*

4.3. Bottleneck situation. Our final example is the bottleneck traffic taken from [24]. The problem statement is given by (2.1)–(2.3), where the flux function $V(\rho, x)$ also depends on the space coordinate x subject to discontinuous maximum density and velocity

$$V(x, \rho) = u_m(x)\rho\left(1 - \frac{\rho}{\rho_m(x)}\right),$$

with

$$\rho_m(x) = \begin{cases} \bar{\rho} & \text{if} \quad x < 0.3, \\ 2 & \text{if} \quad x \geq 0.3 \end{cases} \qquad \text{and} \qquad u_m(x) = \begin{cases} 1 & \text{if} \quad x < 0.3, \\ 0.6 & \text{if} \quad x \geq 0.3. \end{cases}$$

Initially the road has a constant density $\rho(x, 0) = 0.2$. We use two different values of $\bar{\rho}$, namely, $\bar{\rho} = 4$ and $\bar{\rho} = 8$. The evolution of density is given in Figure 7, and the contour plots of density ρ and flux J are given in Figure 8 at $\varepsilon = 10^{-1}$ and $\varepsilon = 10^{-10}$ using $\bar{\rho} = 4$. The results obtained for $\bar{\rho} = 8$ are shown in Figure 9. Our relaxation scheme performs well for this discontinuous situation in traffic flow.

5. Conclusions. We have presented a relaxation-type formulation for hyperbolic traffic flow problems. The advection part of these relaxation systems has only nonnegative velocities so that they can be interpreted as simple discrete-velocity models for traffic flow problems. We showed that high order central discretizations are an essential feature in order to also simulate backward traffic waves. The numerical results show that the performance of our second order numerical schemes is very attractive without solving Riemann problems or requiring special front tracking. More general relaxation systems which may be able to provide multivalued fundamental diagrams and detect stop and go waves, for example, taking the kernel $k = k(\rho, J)$ as in [23], are actually under study.

Appendix: Instability of upwind schemes. In this short appendix we show that the most natural approach for the discretization of a system like (2.4)–(2.5) becomes unstable in the small relaxation limit due to the presence of backward waves. Similar arguments can be extended to higher order upwind methods.

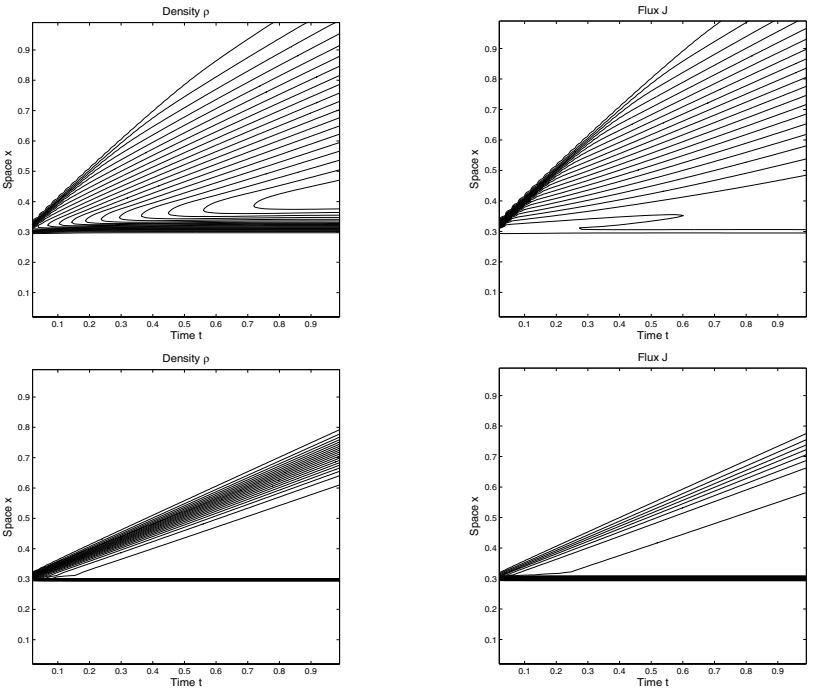


FIG. 8. *Contour plots of density ρ and flux J for the bottleneck traffic situation with $\bar{\rho} = 4$ at $\varepsilon = 10^{-1}$ (top) and $\varepsilon = 10^{-10}$ (bottom).*

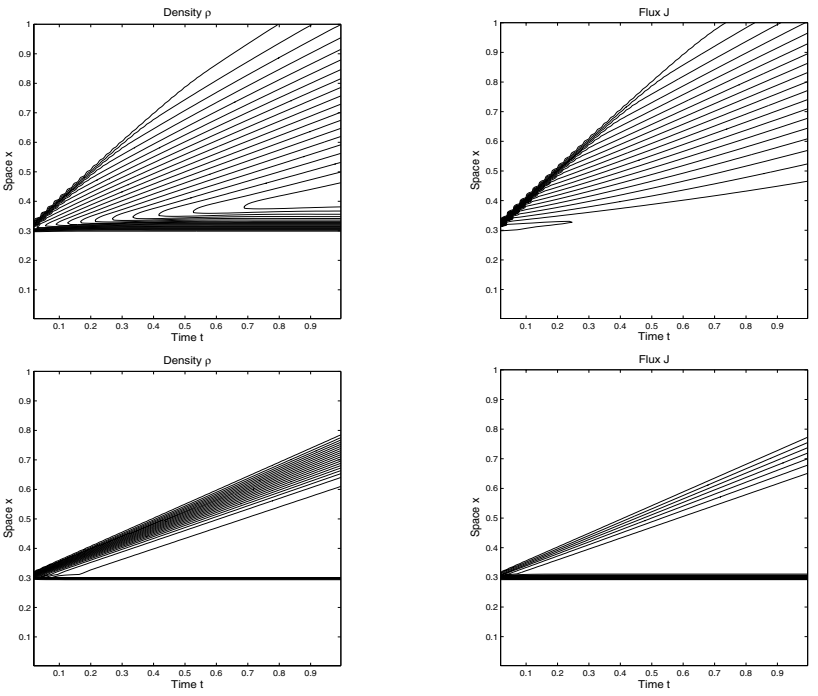


FIG. 9. *Contour plots of density ρ and flux J for the bottleneck traffic situation with $\bar{\rho} = 8$ at $\varepsilon = 10^{-1}$ (top) and $\varepsilon = 10^{-10}$ (bottom).*

The most popular way to solve system (2.4)–(2.5) is to consider, in two separate steps, an advection problem

$$\begin{aligned} &\partial_t f + v_1 \partial_x f = 0, \\ &\partial_t g + v_2 \partial_x g = 0, \end{aligned} \tag{5.1}$$

and a stiff relaxation problem

$$\begin{aligned} &\partial_t f = -\frac{K(\rho)}{2\varepsilon}\Big(v_1 f + v_2 g - V(\rho)\Big), \\ &\partial_t g = \frac{K(\rho)}{2\varepsilon}\Big(v_1 f + v_2 g - V(\rho)\Big). \end{aligned} \tag{5.2}$$

Since the advection part contains only nonnegative speeds, we can apply a simple first order upwind scheme. If we write the upwind scheme in terms of the macroscopic variables, we get

$$\begin{aligned} \rho_i^{n+1} &= \rho_i^n - \frac{\Delta t}{\Delta x}\left(J_i^n - J_{i-1}^n\right), \\ J_i^{n+1} &= J_i^n - \frac{\Delta t}{\Delta x}\left(v_1 + v_2\right)\left(J_i^n - J_{i-1}^n\right) + \frac{\Delta t}{\Delta x} v_1 v_2 \left(\rho_i^n - \rho_{i-1}^n\right). \end{aligned} \tag{5.3}$$

For small values of ε the relaxation step yields the equilibrium projections

$$f_i^n = \frac{\rho_i^n}{v_2 - v_1}\left(v_2 - \frac{V(\rho_i^n)}{\rho_i^n}\right), \qquad g_i^n = \frac{\rho_i^n}{v_2 - v_1}\left(\frac{V(\rho_i^n)}{\rho_i^n} - v_1\right), \tag{5.4}$$

corresponding to $J_i^n = V(\rho_i^n)$ which when substituted into the first equation in (5.3) gives

$$\rho_i^{n+1} = \rho_i^n - \frac{\Delta t}{\Delta x}\left(V(\rho_i^n) - V(\rho_{i-1}^n)\right). \tag{5.5}$$

Clearly (5.5) is an unstable numerical method for the Lighthill–Whitham–Richards equation when $\rho > \rho_m/2$ and thus $V'(\rho) < 0$.

Acknowledgment. M. Herty and M. Seaïd acknowledge the hospitality of the University of Ferrara in Italy, where most of their research was done.

REFERENCES

[1] A. Aw and M. Rascle, *Resurrection of "second order" models of traffic flow*, SIAM J. Appl. Math., 60 (2000), pp. 916–938.

[2] G. Q. Chen, C. D. Levermore, and T. P. Liu, *Hyperbolic conservation laws with stiff relaxation terms and entropy*, Comm. Pure Appl. Math., 47 (1994), pp. 787–830.

[3] C. F. Daganzo, *Requiem for second order fluid approximations of traffic flow*, Transportation Res. Part B., 29 (1995), pp. 277–286.

[4] C. F. Daganzo, *A behavioral theory of multi-lane traffic flow. Part* I. *Long homogenous freeway sections*, Transportation Res. Part B., 36 (2002), pp. 131–158.

[5] S. Gottlieb, C.-W. Shu, and E. Tadmor, *Strong stability-preserving high-order time discretization methods*, SIAM Rev., 43 (2001), pp. 89–112.

[6] G.-S. Jiang, D. Levy, C.-T. Lin, S. Osher, and E. Tadmor, *High-resolution nonoscillatory central schemes with nonstaggered grids for hyperbolic conservation laws*, SIAM J. Numer. Anal., 35 (1998), pp. 2147–2168.

[7] S. Jin, *Runge-Kutta methods for hyperbolic conservation laws with stiff relaxation terms*, J. Comput. Phys., 122 (1995), pp. 51–67.
[8] S. Jin and Z. Xin, *The relaxation schemes for systems of conservation laws in arbitrary space dimensions*, Comm. Pure Appl. Math., 48 (1995), pp. 235–277.
[9] A. Klar, *Relaxation schemes for a lattice-Boltzmann-type discrete velocity model and numerical Navier-Stokes limit*, J. Comput. Phys., 148 (1999), pp. 416–432.
[10] X.-G. Li, X.-J. Yu, and G.-N. Chen, *The third-order relaxation schemes for hyperbolic conservation laws*, J. Comput. Appl. Math., 138 (2002), pp. 93–108.
[11] X. Liu and S. Osher, *Convex ENO high order multi-dimensonal schemes without field by field decomposition or staggered grids*, J. Comput. Phys., 141 (1998), pp. 1–27.
[12] M. J. Lighthill and J. B. Whitham, *On kinematic waves*, Proc. R. Soc. Lond. Ser. A, 229 (1955), pp. 281–345.
[13] A. May, *Traffic Flow Fundamentals*, Prentice–Hall, Englewood Cliffs, NJ, 1990.
[14] R. Natalini, *Convergence to equilibrium for relaxation approximations of conservation laws*, Comm. Pure Appl. Math., 49 (1996), pp. 795–823.
[15] R. Natalini, *A discrete kinetic approximation of entropy solutions to multidimensional scalar conservation laws*, J. Differential Equations, 148 (1998), pp. 292–317.
[16] L. Pareschi and G. Russo, *Implicit-explicit Runge-Kutta schemes for stiff systems of differential equations*, J. Sci. Comput., 25 (2005), pp. 129–155.
[17] L. Pareschi and M. Seaïd, *A new Monte Carlo approach for conservation laws and relaxation systems*, in Proceedings of the 4th International Conference on Computational Science, Lecture Notes in Comput. Sci. 3037, Springer, Berlin, 2004, pp. 276–283.
[18] S. L. Paveri-Fontana, *On Boltzmann-like treatments for traffic flow*, Transportation Res., 9 (1975), p. 225.
[19] H. Payne, *FREFLO: A macroscopic simulation model of freeway traffic*, Transportation Res. Record, 722 (1979), pp. 68–77.
[20] M. Seaïd, *Non-oscillatory relaxation methods for the shallow water equations in one and two space dimensions*, Internat. J. Numer. Methods Fluids, 46 (2004), pp. 457–484.
[21] C.-W. Shu, *Essentially non-oscillatory and weighted essentially non-oscillatory schemes for hyperbolic conservation laws*, in Advanced Numerical Approximation of Nonlinear Hyperbolic Equations, A. Quarteroni, ed., Lecture Notes in Math. 1697, Springer, 1998, pp. 325–432.
[22] C. W. Shu and S. Osher, *Efficient implementation of essential non-oscillatory shock-capturing schemes*, J. Comput. Phys., 77 (1988), pp. 439–471.
[23] F. Siebel and W. Mauser, *On the fundamental diagram of traffic flow*, SIAM J. Appl. Math., 66 (2006), pp. 1150–1162.
[24] P. Zhang and R. Liu, *Generalization of Runge-Kutta discontinuous Galerkin method to LWR traffic flow model with inhomogeneous road conditions*, Numer. Methods Partial Differential Equations, 21 (2004), pp. 80–88.
[25] P. Zhang and R. Liu, *Hyperbolic conservation laws with space dependent flux.* I. *Characteristics theory and Riemann problem*, J. Comput. Appl. Math., 156 (2003), pp. 1–21.
[26] P. Zhang and R. Liu, *Hyperbolic conservation laws with space dependent fluxes.* II. *General study on numerical fluxes*, J. Comput. Appl. Math., 176 (2005), pp. 105–129.
[27] P. Zhang, R. Liu, and S. C. Wong, *High-resolution numerical approximation of traffic flow problems with variable lanes and free flow velocities*, Phys. Rev. E, 71 (2005), 56704.

SIAM J. SCI. COMPUT.
Vol. 28, No. 5, pp. 1597–1612
© 2006 Society for Industrial and Applied Mathematics

A NUMERICAL STUDY OF THE PROBE METHOD*

KLAUS ERHARD† AND ROLAND POTTHAST†

Abstract. The goal of this work is the numerical realization of the *probe method* suggested by Ikehata for the detection of an obstacle D in inverse scattering. The main idea of the method is to use probes in the form of point source $\Phi(\cdot, z)$ with source point z to define an indicator function $\hat{I}(z)$ which can be reconstructed from Cauchy data or far field data. The indicator function $\hat{I}(z)$ can be shown to blow off when the source point z tends to the boundary ∂D, and this behavior can be used to find D. To study the feasibility of the probe method we will use two equivalent formulations of the indicator function. We will carry out the numerical realization of the functional and show reconstructions of a sound-soft obstacle.

Key words. inverse scattering, integral equations, obstacle reconstruction, probe method, point source method, sampling methods

AMS subject classifications. 35J05, 35R30, 45A05, 45Q05

DOI. 10.1137/040607149

1. Introduction.

The central task of the discipline of inverse scattering is to extract information about scattering objects from the scattered fields u^s or their far field patterns u^∞ of acoustic, electromagnetic, or elastic waves. Here, we will restrict our attention to acoustic scattering by a scatterer $\mathcal{D}$ for a set of incident plane waves $\{u^i(\cdot, d) : d \in \mathbb{S}\}$, where $\mathbb{S}$ is the unit sphere in two or three dimensions.

Several approaches have been introduced to solve different inverse problems. Distinguishing three categories as in [9], we mention iterative algorithms, decomposition or analytic extension methods, and sampling/probe methods. For iterative algorithms we refer to [2] and the literature cited therein. Decomposition methods, such as the Kirsch–Kress scheme or the point source method, have been extensively studied in [9] and [2]. Sampling and probe methods have been suggested since 1996 by Colton, Kirsch, Potthast, Ikehata, and many others; see [10] for a recent survey.

Ikehata [4] suggested using an indicator function $\hat{I}(z)$, which is based on the behavior of the integral

$$\int_{\Omega \setminus D} |\nabla \Phi^s(\cdot, z)|^2 dx \tag{1.1}$$

and related terms in the form of point source $\Phi(\cdot, z)$ with source point z (cf. (2.6)), and on its scattered field $\Phi^s(\cdot, z)$. When z tends to the boundary, the integral will diverge. This function is calculated using Cauchy data on the boundary of the domain Ω, which can be reconstructed from the far field patterns. So far, this method has not been numerically realized. The realization is the main task of this paper.

Our main goal is to employ the techniques of the point source and singular sources method [9] to numerically realize Ikehata's probe method. To this end, we will first summarize results about the forward problem in section 2. In section 3 we summarize

*Received by the editors April 22, 2004; accepted for publication (in revised form) January 30, 2006; published electronically September 29, 2006.
http://www.siam.org/journals/sisc/28-5/60714.html
†Institute for Numerical and Applied Mathematics, University of Göttingen, Lotzestr. 16-18, D-37083 Göttingen, Germany (kserhard@math.uni-goettingen.de, http://www.num.math.uni-goettingen.de/kserhard/; potthast@scienceatlas.de, http://www.scienceatlas.de/potthast/).

Ikehata's approach to the probe method. Section 4 serves to formulate a practical realization of the probe method with the help of the techniques developed in the framework of the point source method. In section 5 we present a numerical study of the probe method and its functional. In particular, we provide two equivalent formulations for the indicator function of the probe method. The goal of these formulations is to obtain a well-posed scheme for the calculation of the indicator function and to study its behavior.

2. The solution of the boundary value problem and the Dirichlet to Neumann map. The starting points of Ikehata's probe method are the Dirichlet to Neumann (D-t-N) maps Λ for the direct obstacle boundary value problem and Λ_0 for the obstacle-free boundary value problem. In practice, the D-t-N map is often available via physical measurements. For our numerical study we simulate the D-t-N maps by solving their underlying boundary value problems explicitly. Therefore, we will first introduce both the direct obstacle boundary value problem and the obstacle-free boundary value problem and briefly present a way of solving them numerically.

The following direct obstacle boundary value problem has its origin in the acoustic scattering theory of sound-soft obstacles; see, for example, [2]. We consider a bounded domain $\Omega \subset \mathbb{R}^m$, $m = 2, 3$, with a boundary $\partial\Omega$ of class C^2 and a normal derivative ν oriented into the exterior of Ω. Then, the obstacle $D \subset \Omega$ is modeled as a domain with a boundary ∂D of class C^2 with a normal vector ν oriented into the interior of D. Therefore, the boundary of the annular region $\Omega_D := \Omega \backslash \overline{D}$ is given by $\partial\Omega_D = \partial\Omega \cup \partial D$ with a normal vector ν, which is oriented into the exterior of Ω_D; see Figure 1. With κ, we denote the complex wave number, for which $\mathrm{Re}(\kappa) > 0$ and $\mathrm{Im}(\kappa) \geq 0$ hold.

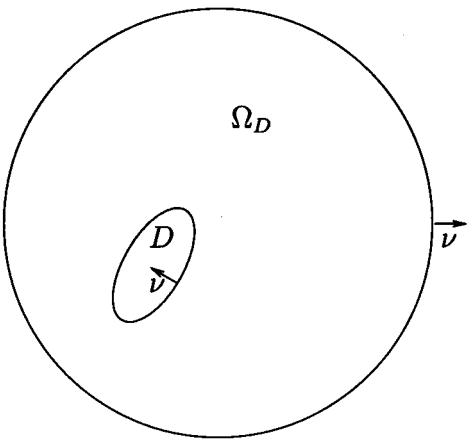


FIG. 1. *The normal ν to the boundary of the domain Ω_D.*

With this notation the direct obstacle boundary value problem and the obstacle-free boundary value problem read as follows.

Direct obstacle boundary value problem. For a given differentiable function $f \in C^{1,\alpha}(\partial\Omega)$, $0 < \alpha < 1$, with a Hölder continuous derivative and a continuous function $g \in C(\partial D)$, we want to find a function $u \in C^2(\Omega_D) \cap C(\overline{\Omega_D})$ such that

$$\Delta u + \kappa^2 u = 0 \quad \text{in } \Omega_D, \tag{2.1}$$

$$u = f \quad \text{on } \partial\Omega, \tag{2.2}$$

$$u = g \quad \text{on } \partial D. \tag{2.3}$$

. . . . . The assumption of Hölder differentiability will be needed for the evaluation of the normal derivative of the fields under consideration.

Obstacle-free boundary value problem. For a given continuous function $f \in C(\partial\Omega)$ we want to find a function $v \in C^2(\Omega) \cap C(\overline{\Omega})$ such that

$$\Delta v + \kappa^2 v = 0 \quad \text{in } \Omega, \tag{2.4}$$

$$v = f \quad \text{on } \partial\Omega. \tag{2.5}$$

We tacitly assume in the following that $-\kappa^2$ is not an eigenvalue of the Laplacian in Ω_D or in Ω. Then, the direct obstacle boundary value problem has a unique solution, which can be represented with a combined layer potential; see [3]. An analogous result holds for the obstacle-free boundary value problem, but since we do not need its numerical solution for our study of the probe method we will omit it here.

For $x, y \in \mathbb{R}^m$, we denote by

$$\Phi(x,y) = \begin{cases} \frac{i}{4} H_0^{(1)}(\kappa|x-y|), & m = 2, \\ \frac{1}{4\pi} \frac{\exp(i\kappa|x-y|)}{|x-y|}, & m = 3, \end{cases} \tag{2.6}$$

the fundamental solution of the Helmholtz equation, where $H_0^{(1)}$ is the Hankel function of first kind and order zero. Then the solution to the direct obstacle boundary value problem can be represented by the combined layer potential

$$u(x) = \int_{\partial\Omega_D} \Big(\frac{\partial\Phi(x,y)}{\partial\nu(y)} - i\eta\Phi(x,y) \Big) \varphi(y)\, ds(y), \quad x \in \Omega_D, \tag{2.7}$$

with a positive number η and a density $\varphi \in C(\partial\Omega) \times C(\partial D)$. In order for the combined layer potential (2.7) to admit the boundary values

$$b(x) := \begin{cases} f(x), & x \in \partial\Omega, \\ g(x), & x \in \partial D, \end{cases} \tag{2.8}$$

the density φ has to be a solution of the integral equation

$$(K - i\eta S)\varphi - \frac{1}{2}\varphi = b \tag{2.9}$$

on $\partial\Omega_D$, where the integral operators K and S are defined via

$$(K\varphi)(x) := \int_{\partial\Omega_D} \frac{\partial\Phi(x,y)}{\partial\nu(y)} \varphi(y)\, ds(y), \quad x \in \partial\Omega_D, \tag{2.10}$$

and

$$(S\varphi)(x) := \int_{\partial\Omega_D} \Phi(x,y)\varphi(y)\, ds(y), \quad x \in \partial\Omega_D. \tag{2.11}$$

Hence, solving the integral equation (2.9) yields a representation of the solution to the direct obstacle boundary value problem (2.1)–(2.3) in the form of (2.7); see [3] for details. We remark that due to the mapping properties of the integral operators S and K from C into $C^{0,\alpha}$ and from $C^{0,\alpha}$ into $C^{1,\alpha}$ using (2.9) and the regularity of the functions f and g on the boundary we obtain $\varphi \in C^{1,\alpha}(\partial\Omega) \times C(\partial D)$.

Next, we consider the D-t-N maps Λ and Λ_0 of the direct obstacle and obstacle-free boundary value problems.

D-t-N map Λ. The D-t-N map Λ of the direct obstacle boundary value problem maps a Hölder continuously differentiable function $f \in C^{1,\alpha}(\partial\Omega)$ to the continuous function $h \in C^{0,\alpha}(\partial\Omega)$ defined by

$$h(x) := \frac{\partial u}{\partial \nu}(x) \tag{2.12}$$

for $x \in \partial\Omega$, where u solves the direct obstacle boundary value problem (2.1)–(2.3) with a sound-soft boundary condition on ∂D, i.e., with u equal to $g \equiv 0$ on ∂D.

D-t-N map Λ_0. The D-t-N map Λ_0 of the obstacle-free boundary value problem maps a continuous function $f \in C(\partial\Omega)$ to the continuous function $h_0 \in C(\partial\Omega)$ given by

$$h_0(x) := \frac{\partial v}{\partial \nu}(x) \tag{2.13}$$

for $x \in \partial\Omega$, where v solves the obstacle-free boundary value problem (2.4)–(2.5).

As mentioned before, to implement the probe method we have to simulate the D-t-N map Λ in our numerical studies. We realize this in the following way: For a given Hölder continuously differentiable function f we solve the direct obstacle boundary value problem (2.1)–(2.3) with a sound-soft boundary condition $g \equiv 0$ on ∂D using the combined layer approach (2.7) by solving the integral equation (2.9). Then, we calculate the normal derivative of u at the boundary $\partial\Omega$ numerically in order to obtain $h = \Lambda f$.

The behavior of the layer potentials K and S at the boundary $\partial\Omega$ forbids us to apply a simple difference quotient for calculating h. Instead, we differentiate the combined layer potential (2.7) analytically to get the expression

$$h(x) = (T\varphi)(x) - i\eta(K'\varphi)(x) - i\eta\frac{1}{2}\varphi(x), \quad x \in \partial\Omega, \tag{2.14}$$

for the normal derivative on the boundary $\partial\Omega$. Here, T denotes the hypersingular integral operator

$$(T\varphi)(x) := \frac{\partial}{\partial\nu(x)} \int_{\Omega_D} \frac{\partial\Phi(x,y)}{\partial\nu(y)}\varphi(y)\, ds(y), \quad x \in \partial\Omega, \tag{2.15}$$

and K' is the adjoint operator of K, namely,

$$(K'\varphi)(x) := \int_{\Omega_D} \frac{\partial\Phi(x,y)}{\partial\nu(x)}\varphi(y)\, ds(y), \quad x \in \partial\Omega. \tag{2.16}$$

For the numerical evaluation of the hypersingular integral operator T, we consider the hypersingular part T_Ω of T, which is defined as

$$(T_\Omega\varphi_\Omega)(x) := \frac{\partial}{\partial\nu(x)} \int_{\partial\Omega} \frac{\partial\Phi(x,y)}{\partial\nu(y)}\varphi_\Omega(y)\, ds(y), \quad x \in \partial\Omega, \tag{2.17}$$

with a Hölder continuously differentiable density $\varphi_\Omega \in C^{1,\alpha}(\partial\Omega)$. The integral operator T_D is defined analogously for continuous densities $\varphi_D \in C(\partial D)$ on the interior boundary ∂D. Since the integral operator $T_D : C(\partial D) \to C(\partial\Omega)$ has an analytic

kernel, it can easily be evaluated by a numerical differentiation of the contribution $K_D\varphi_D$ from the interior boundary to the double layer potential $K\varphi$.

Next, the hypersingular operator T_Ω can be written as

$$T_\Omega\varphi_\Omega = \frac{d}{ds}S_\Omega\frac{d\varphi_\Omega}{ds} + \kappa^2\nu\cdot S_\Omega(\nu\varphi_\Omega) \tag{2.18}$$

due to Maue's identity [8] with the contribution S_Ω from the outer boundary $\partial\Omega$ to the single-layer operator S. After parametrizing (2.18) and integrating by parts, we obtain

$$\begin{aligned}&\left(\frac{d}{ds}S_\Omega\frac{d\varphi_\Omega}{ds}\right)(x(t))\\ &= \frac{1}{|x'(t)|}\int_0^{2\pi}\left\{\frac{1}{2\pi}\cot\frac{\tau-t}{2}\frac{d\varphi_\Omega(x(\tau))}{d\tau} - N(t,\tau)\varphi_\Omega(x(\tau))\right\}d\tau\end{aligned}$$

with

$$N(t,\tau) = N_1(t,\tau)\ln\left(4\sin^2\frac{t-\tau}{2}\right) + N_2(t,\tau) \tag{2.19}$$

and analytic functions $N_1(t,\tau)$ and $N_2(t,\tau)$. For the integration of these terms, we use the interpolatory quadrature rules

$$\frac{1}{2\pi}\int_0^{2\pi}\cot\frac{\tau-t}{2}f'(\tau)\,d\tau \approx \sum_{j=0}^{2n-1}T_j^{(n)}(t)f(t_j^{(n)}) \tag{2.20}$$

and

$$\int_0^{2\pi}\ln\left(4\sin^2\frac{t-\tau}{2}\right)f(\tau) \approx \sum_{j=0}^{2n-1}R_j^{(n)}(t)f(t_j^{(n)}) \tag{2.21}$$

with the weights

$$T_j^{(n)}(t) = -\frac{1}{n}\sum_{m=1}^{n-1}m\cos m(t-t_j^{(n)}) - \frac{1}{2}\cos n(t-t_j^{(n)}) \tag{2.22}$$

and

$$R_j^{(n)}(t) = -\frac{2\pi}{n}\sum_{m=1}^{n-1}\frac{1}{m}\cos m(t-t_j^{(n)}) - \frac{\pi}{n^2}\cos n(t-t_j^{(n)}). \tag{2.23}$$

Here, $t_j^{(n)} = \frac{j\pi}{n}$, $j = 0,\dots,2n-1$, are $2n$ equidistant sampling points of the interval $[0,2\pi]$. For details, we refer to the work of Kress [7].

3. The probe method. For the reconstruction of the obstacle D, the probe method uses the [illegible]

$$I(f) := \int_{\partial\Omega}\overline{(\Lambda-\Lambda_0)f(y)}\cdot f(y)\,ds(y), \tag{3.1}$$

with $f\in C^{1,\alpha}(\partial\Omega)$, and given D-t-N maps Λ and Λ_0. For any $z\in\Omega$, we will choose a domain of approximation $G(z)$ with $z\notin G(z)$ and find a sequence $(v_{n,z})_{n\in\mathbb{N}}$ of

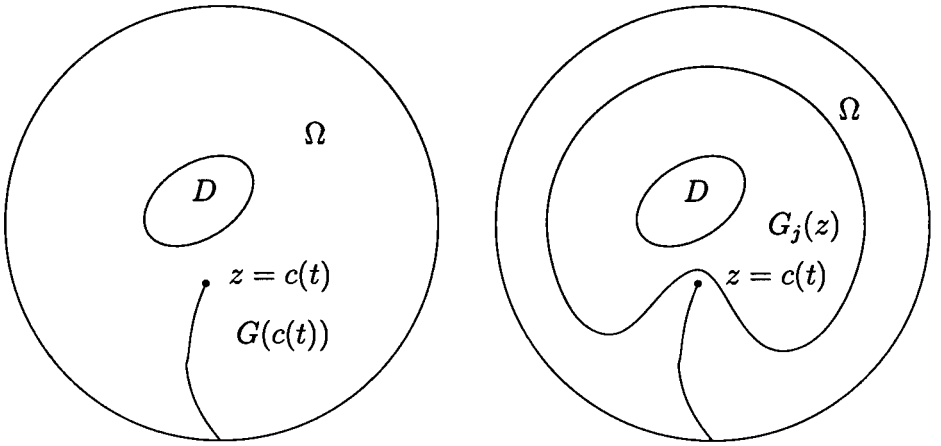


FIG. 2. *The needle domain $G(c(t))$ (left) and an approximation $G_j(z)$ (right) to the needle domain $G(c(t))$. Here, the index j indicates that we can choose a sequence of such approximation domains to successively fill the set $G(z)$. In this sense our approach can be viewed as a numerical realization of the needle approach.*

solutions $v_{n,z} \in C^2(\Omega) \cap C^{1,\alpha}(\overline{\Omega})$ to the Helmholtz equation in Ω, which approximates the point source $\Phi(\cdot, z)$ on the approximation domain $G(z)$, i.e., with

$$\lim_{n\to\infty} \|\Phi(\cdot, z) - v_{n,z}\|_{L^2(G(z))} = 0. \tag{3.2}$$

Given a point $z \in \Omega$ with $\overline{D} \subset G(z)$, $z \notin G(z)$, we use the notation

$$\hat{I}(z) := \lim_{n\to\infty} I(f_{n,z}), \tag{3.3}$$

where $f_{n,z} := v_{n,z}|_{\partial\Omega}$ and I is defined in (3.1). The function $\hat{I}$ is well defined in view of the following theorem.

THEOREM 1. [illegible] $z \in \Omega$ [illegible] $\overline{D} \subset G(z)$ $z \notin G(z)$ [illegible] $\hat{I}(z)$ [illegible] z_j [illegible] $\overline{D} \subset G(z_j)$ $z_j \notin G(z_j)$ [illegible] $j \in \mathbb{N}$, $z_j \to \partial D$, $j \to \infty$ [illegible]

$$|\mathrm{Re}\hat{I}(z_j)| \to \infty, \quad j \to \infty. \tag{3.4}$$

[illegible]. The proof can be adapted easily from Theorem 3.1 in [1]. □

In the work [1], which is based upon the work of Ikehata (see, for example, [4], [5], and [6]), a needle approach is proposed to reconstruct the obstacle D. The authors define a needle as a continuous function $c : [0,1] \longrightarrow \overline{\Omega}$ with $c(0), c(1) \in \partial\Omega$, and $c(t) \in \Omega$ for $0 < t < 1$; i.e., a needle is a path in Ω for a probe starting and ending at the boundary. Then, the function $\hat{I}$ is evaluated for the points $z = c(t)$ on the needle; cf. Figure 2. In [1] they choose the domains

$$G(z) = G(c(t)) = \Omega \setminus \{c(\tau) | 0 < \tau \leq t\} \tag{3.5}$$

for the approximation of the point source $\Phi(\cdot, c(t))$ in a Sobolev space setting under some additional constraints. In [4] the existence of such an approximating function is proved with Runge's approximation theorem, which uses the Hahn–Banach theorem and leads, therefore, to a nonconstructive approximation method.

The approximation domains (3.5) are very suitable for theoretical reasons, since the boundary can then be described easily by the set

$$\partial D = \{c(t) | t = T(c), c \text{ is a needle with } T(c) < 1\}, \tag{3.6}$$

with the impact parameter

$$T(c) = \sup \left\{ 0 < t < 1 \left\| \sup_{0 \le s \le t} |\mathrm{Re}\hat{I}(c(s))\right\| < \infty \right\}. \tag{3.7}$$

Note that $T(c) > 0$. Further, we remark that $T(c) = 1$ if and only if the needle c does not intersect $\overline{D}$.

In this paper, we will provide an explicit approximate function $v_{n,z}$ to the fundamental solution $\Phi(\cdot, z)$ on a particular class of domains $G(z)$ in terms of a Herglotz wave function without using Runge's approximation theorem. Together with the proper choice of the approximation domain G, this will be one of the key steps in the reconstruction algorithm. Figure 2 illustrates this special choice of approximation domains.

4. Numerics for the probe method.

4.1. Point source approximation by integral equations. For every $z \in \Omega$ with a corresponding approximation domain $G(z)$ having a sufficiently smooth boundary such that $z \notin G(z)$ and $-\kappa^2$ is not a Dirichlet eigenvalue for the Laplacian in $G(z)$, we can find a sequence of Herglotz wave functions

$$v_{n,z}(x) := \int_{\mathbb{S}} e^{i\kappa x \cdot d} g_{n,z}(d)\, ds(d) \tag{4.1}$$

with density functions $g_{n,z} \in L^2(\mathbb{S})$ such that the sequence $(v_{n,z})_n$ fulfills (3.2). The justification of this statement is provided by the following theorem.

THEOREM 2. [illegible] $G \subset \mathbb{R}^2$ [illegible] C^2 [illegible] $-\kappa^2$ [illegible] G [illegible] $H : L^2(\mathbb{S}) \longrightarrow L^2(\partial G)$, [illegible]

$$(Hg)(x) = \int_{\mathbb{S}} e^{i\kappa x \cdot d} g(d)\, ds(d), \tag{4.2}$$

[illegible] $L^2(\partial G)$ [illegible]. The proof can be found in the book [9, Lemma 3.1.2]. ∎

We mention (see [3]) that the point source $\Phi(\cdot, z)$ is not in the range of the compact operator H, and since H has no bounded inverse we have to apply a regularization scheme to obtain the densities $g_{n,z}$. The assumption on κ, together with the unique continuation principle and Theorem 3.15 in [2], proves the injectivity of G. Hence, we can apply a Tikhonov regularization scheme to substitute the ill-posed integral equation

$$Hg = \Phi(\cdot, z) \tag{4.3}$$

of the first kind with its regularized version

$$(\alpha I + H^* H) g = H^* \Phi(\cdot, z). \tag{4.4}$$

The operator

$$R_\alpha := (\alpha I + H^* H)^{-1} H^*, \quad \alpha > 0, \tag{4.5}$$

applied to H converges pointwise to the identity on $L^2(\mathbb{S})$,

$$\lim_{\alpha \to 0} R_\alpha H g = g, \tag{4.6}$$

for all $g \in L^2(\mathbb{S})$, and its norm satisfies $\|R_\alpha\| \leq \frac{1}{2\alpha}$. The main focus of this work is on the feasibility of the probe method, rather than the choice of α. Hence, we will choose by trial and error a fixed regularization parameter for our numerical examples.

4.2. Choice of the approximation domains. In principle, the approximation procedure of the last section can be used to construct the approximations of the point source needed for the probe method. This can be carried out by using a family of approximation domains $G_j(z)$, $j = 1, 2, 3, \ldots$, which fill step-by-step the whole approximation domain $G(z)$, as defined in (3.5). Then, for every compact subset M of $G(z)$ we can find $j \in \mathbb{N}$ such that $M \subset G_j(z)$, and on M we obtain an approximation of $\Phi(\cdot, z)$ by $v_{n,z} = H g_{\alpha,z}$ from the solution $g_{\alpha,z}$ to (4.4). However, we experienced the following instability phenomenon. The better the approximation domain $G_j(z)$ approximates the needle domain $G(z)$, the worse is the quality of the point source approximation on $G_j(z)$. For the needle domain, a numerical approximation becomes impossible. Therefore, we need to restrict our numerics to some simple approximation domains $G_j(z)$ and we do not carry out the limit $j \to \infty$; see Figure 2.

For the approximation of a point source $\Phi(\cdot, z)$ on some appropriately chosen approximation domains $G(z)$ by the Herglotz wave operator (4.2), the density $g_{\alpha,z}$ can be calculated as the approximate solution of the integral equation of the first kind (4.3) via (4.4). Hence, for each $z \in \Omega$, it is necessary to solve and regularize an ill-posed integral equation. This can be made efficient by using translation and rotations as follows. Consider approximation domains with a fixed shape, i.e.; approximation domains of the form

$$G(z) = \{M(z)x + z : x \in G_0, M(z) \in \mathrm{SO}(2, \mathbb{R})\} \tag{4.7}$$

with a fixed approximation domain $G_0 := G(0)$, corresponding to the point $z = 0$, where we assume $0 \in \Omega$ w.l.o.g.. Here $\mathrm{SO}(2, \mathbb{R})$ is the set of real, orthogonal 2×2 matrices with $\det(M) = 1$. We remark that $G(z)$ emerges from G_0 by first rotating the domain G_0 and then translating it by the vector z; i.e., the relative position of the source point z to the approximation domain $G(z)$ remains fixed for all $z \in \Omega$ with corresponding domains $G(z)$. In this case, it is sufficient to solve the ill-posed integral equation only once—for example, for the point $z = 0$—as stated by the following theorem.

THEOREM 3. [illegible] $\varepsilon > 0$ $0 \in \Omega$ [illegible] $g_{\alpha,0} \in L^2(\mathbb{S})$ [illegible]

[illegible]

$$(\alpha I + H^* H) g = H^* \Phi(\cdot, 0) \tag{4.8}$$

[illegible] ∂G_0 [illegible]

$$\|H g_{\alpha,0} - \Phi(\cdot, 0)\|_{L^2(\partial G_0)} < \varepsilon. \tag{4.9}$$

$z \in \Omega$ $G(z)$ (4.7)

$$g_{\alpha,z}(d) = e^{-i\kappa z\cdot d} g_{\alpha,0}(M(z)^t d) \tag{4.10}$$

$$\|Hg_{\alpha,z} - \Phi(\cdot, z)\|_{L^2(\partial G(z))} < \varepsilon. \tag{4.11}$$

We define

$$G'(z) := \{M(z)x : x \in G_0\}. \tag{4.12}$$

Then, we obtain for the left-hand side of (4.11) with fixed $z \in \Omega$

$$\begin{aligned}
&\left\| \int_{\mathbb{S}} e^{i\kappa x\cdot d} g_{\alpha,z}(d)\, ds(d) - \Phi(x,z) \right\|_{L^2(\partial G(z))} \\
&= \left\| \int_{\mathbb{S}} e^{i\kappa (x+z)\cdot d} g_{\alpha,z}(d)\, ds(d) - \Phi(x+z,z) \right\|_{L^2(\partial G'(z))} \\
&= \left\| \int_{\mathbb{S}} e^{i\kappa z\cdot d} e^{i\kappa x\cdot d} g_{\alpha,z}(d)\, ds(d) - \Phi(x,0) \right\|_{L^2(\partial G'(z))}.
\end{aligned} \tag{4.13}$$

With $\det(M(z)) = 1$ and $\Phi(M(z)x, 0) = \Phi(x, 0)$, (4.13) simplifies to

$$\begin{aligned}
&\left\| \int_{\mathbb{S}} e^{i\kappa z\cdot d} e^{i\kappa (M(z)x)\cdot d} g_{\alpha,z}(d)\, ds(d) - \Phi(x,0) \right\|_{L^2(\partial G_0)} \\
&= \left\| \int_{\mathbb{S}} e^{i\kappa z\cdot d} e^{i\kappa x\cdot \left(M(z)^t d\right)} g_{\alpha,z}(d)\, ds(d) - \Phi(x,0) \right\|_{L^2(\partial G_0)} \\
&= \left\| \int_{\mathbb{S}} e^{i\kappa z\cdot \left(M(z)\tilde{d}\right)} e^{i\kappa x\cdot \tilde{d}} g_{\alpha,z}(M(z)\tilde{d})\, ds(\tilde{d}) - \Phi(x,0) \right\|_{L^2(\partial G_0)}.
\end{aligned} \tag{4.14}$$

Inserting the definition of $g_{\alpha,z}$ into (4.14) ends the proof. □

The previous theorem reduces the complexity of calculating the densities $g_{\alpha,z}$ from solving a linear system to a simple pointwise vector multiplication. Choosing a fixed base domain G_0, from which all approximation domains $G(z)$ arise by a rotation and a translation by the vector z, the reconstruction algorithm for the probe method using Herglotz wave functions for approximating $\Phi(\cdot, z)$ can be written as follows.

Reconstruction algorithm for the probe method.

1. Choose a test domain G_0 such that $0 \notin G_0$ large enough.
2. For a proper choice of the regularization parameter α, calculate $g_{\alpha,0}$ by solving the regularized boundary integral equation (4.8).
3. For some evaluation points $z_j \in \Omega$, $j = 1, \ldots, N$, calculate the densities g_{α,z_j} $j = 1, \ldots, N$, via (4.10) by rotation of the argument on the unit circle and multiplication with a complex factor.
4. Calculate $f_j := v_{z_j}|_{\partial\Omega}$ and $\Lambda_0 f_j$ for $j = 1, \ldots, N$ by evaluating the Herglotz wave function (4.1) and its normal derivative, respectively.
5. (a) Solve the direct obstacle boundary value problem (2.1)–(2.3) with $f = f_j$ and $g = 0$ for $j = 1, \ldots, N$. Then, calculate Λf_j with (2.14), (simulation of the D-t-N map).
   or

(b) Calculate Λf_j by some externally given D-t-N map for $j = 1, \ldots, N$.

6. Calculate the approximation $I(f_j) \approx \hat{I}(z_j)$ for $j = 1, \ldots, N$ by evaluating the boundary integral (3.1) with the ingredients given by the previous steps. Calculate $|\mathrm{Re}(I(f_j))|$ for every $j = 1, \ldots, N$.
7. Choose some suitable cutoff constant $C > 0$ and decide if
   (a) $z_j \in D$ if $|\mathrm{Re}(I(f_j))| > C$, or
   (b) $z_j \in \mathbb{R}^m \setminus \overline{D}$ if $|\mathrm{Re}(I(f_j))| \leq C$
   for every $j = 1, \ldots, N$.

We remark here that the decision criterion is justified only in the case when $D \subset G(z_j)$. In contrast to the limiting case of the needle approach, where $D \subset G(c(t))$ is always satisfied as long as $t < T(c)$, we do not have information about whether $D \subset G(z_j)$ or not. Therefore, the decision criterion given in the reconstruction algorithm is based on heuristics. Nevertheless, in our numerical examples this decision criterion led to satisfactory results.

5. Numerical study for the probe method.

5.1. Two equivalent forms of the Ikehata functional. In this section, we want to give an equivalent formulation of the Ikehata functional (3.1) without using the D-t-N maps. To this end, we define with u_n the solution to the direct obstacle boundary value problem (2.1)–(2.3) with boundary values $f = f_n$ on $\partial\Omega$ and $g \equiv 0$ on ∂D. Furthermore v_n denotes the solution to the obstacle-free boundary value problem (2.4)–(2.5) with boundary data $f = f_n$ on $\partial\Omega$. Then, we have the following reformulation for the Ikehata functional for a sound-soft obstacle.

LEMMA 4. [illegible]

$$I(f_n) = \int_{\Omega_D} \left(|\nabla w_n|^2 - \kappa^2 |w_n|^2 \right) dx + \int_D \left(|\nabla v_n|^2 - \kappa^2 |v_n|^2 \right) dx \,, \tag{5.1}$$

[illegible] $w_n := u_n - v_n$ [illegible]. The proof can be found in [6]. For the convenience of the reader, and since we numerically study the above integrals, here we present the calculations. From Green's formula, together with $u_n = 0$ on ∂D, we obtain

$$\begin{aligned} \int_{\Omega_D} |\nabla(u_n - v_n)|^2 \, dx &= \int_{\partial\Omega} u_n \frac{\partial(\overline{u}_n - \overline{v}_n)}{\partial\nu} \, ds + \kappa^2 \int_{\Omega_D} u_n(\overline{u}_n - \overline{v}_n) \, dx \\ &\quad - \int_{\Omega_D} \nabla v_n \nabla(\overline{u}_n - \overline{v}_n) \, dx \\ &= \langle (\Lambda_D - \Lambda_0)\overline{f}_n, f_n \rangle + \kappa^2 \int_{\Omega_D} u_n(\overline{u}_n - \overline{v}_n) \, dx \\ &\quad - \int_{\Omega_D} \nabla v_n \nabla \overline{u}_n \, dx + \int_{\Omega_D} |\nabla v_n|^2 \, dx \,. \end{aligned} \tag{5.2}$$

Using $\nabla v_n \cdot \nabla \overline{u}_n = \nabla \cdot (\overline{u}_n \nabla v_n) - \overline{u}_n \Delta v_n$ and the Gaussian divergence theorem we get

$$\begin{aligned} - \int_{\Omega_D} \nabla v_n \nabla \overline{u}_n \, dx + \int_{\Omega_D} |\nabla v_n|^2 \, dx &= - \int_{\partial\Omega} \overline{u}_n \frac{\partial v_n}{\partial\nu} \, ds - \kappa^2 \int_{\Omega_D} \overline{u}_n v_n \, dx \\ &\quad + \int_{\Omega} |\nabla v_n|^2 \, dx - \int_D |\nabla v_n|^2 \, dx \,, \end{aligned}$$

and again with $\nabla v_n \cdot \nabla \overline{v}_n = \nabla \cdot (\overline{v}_n \nabla v_n) - \overline{v}_n \Delta v_n$ we observe

$$-\int_{\Omega_D} \nabla v_n \nabla \overline{u}_n \, dx + \int_{\Omega_D} |\nabla v_n|^2 \, dx = -\kappa^2 \int_{\Omega_D} \overline{u}_n v_n \, dx + \kappa^2 \int_{\Omega} |v_n|^2 \, dx - \int_D |\nabla v_n|^2 \, dx \,,$$

since $u_n - v_n = 0$ on $\partial\Omega$. With this expression we can reformulate (5.2) to

$$\begin{aligned} I_n(c(t)) &= \langle (\Lambda_D - \Lambda_0)\overline{f}_n, f_n \rangle \\ &= \int_{\Omega_D} \left(|\nabla w_n|^2 - \kappa^2 u_n(\overline{u}_n - \overline{v}_n) + \kappa^2 \overline{u}_n v_n \right) dx \\ &\quad -\kappa^2 \int_{\Omega} |v_n|^2 \, dx + \int_D |\nabla v_n|^2 \, dx \\ &= \int_{\Omega_D} \left(|\nabla w_n|^2 - \kappa^2 (u_n - v_n)(\overline{u}_n - \overline{v}_n) \right) dx + \kappa^2 \int_{\Omega_D} |v_n|^2 \, dx \\ &\quad -\kappa^2 \int_{\Omega} |v_n|^2 \, dx + \int_D |\nabla v_n|^2 \, dx \\ &= \int_{\Omega_D} \left(|\nabla w_n|^2 - \kappa^2 |w_n|^2 \right) dx + \int_D \left(|\nabla v_n|^2 - \kappa^2 |v_n|^2 \right) dx \,, \end{aligned}$$

which ends the proof. □

Since the domain of integration is unknown for both integrals, this formulation of the Ikehata functional cannot be used for reconstructions. Nevertheless, it will serve as a comparison with the D-t-N formulation (3.1) of the Ikehata functional in two ways. First, with the help of Lemma 4 we can calculate the Ikehata functional for all $z \in \Omega_D$ without any approximation of the point source $\Phi(\cdot, z)$. To this end, we replace v_n and w_n in (5.1) with $v_{\infty,z} = \Phi(\cdot, z)$ and $w_{\infty,z}$, which denotes the solution to the boundary value problem

$$\Delta w + \kappa^2 w = 0 \quad \text{in } \Omega_D, \tag{5.3}$$

$$w = 0 \quad \text{on } \partial\Omega, \tag{5.4}$$

$$w = -\Phi(\cdot, z) \quad \text{on } \partial D. \tag{5.5}$$

Second, we can also evaluate the Ikehata functional (5.1) with approximate functions to illustrate the influence of the approximation of the point source function on the Ikehata functional. For this purpose we substitute v_n and w_n in (5.1) with $v_{\alpha,z} \approx \Phi(\cdot, z)$, which is defined by (4.1) and (4.10), and with $w_{\alpha,z}$, which is the solution to the boundary value problem

$$\Delta w + \kappa^2 w = 0 \quad \text{in } \Omega_D, \tag{5.6}$$

$$w = 0 \quad \text{on } \partial\Omega, \tag{5.7}$$

$$w = -v_{\alpha,z}|_{\partial D} \quad \text{on } \partial D. \tag{5.8}$$

Both domain integrals in (5.1) can be numerically evaluated with the rectangle rule. The second integral can be easily calculated for any point $z \in \Omega_D$ with both the approximate function $v_{\alpha,z}$ (4.1) and the point source function $v_{\infty,z} = \Phi(\cdot, z)$ itself. The gradients of these functions are given by

$$\nabla v_{\alpha,z}(x) = i\kappa \int_{\mathbb{S}} d \, e^{i\kappa x \cdot d} g_{\alpha,z}(d) \, ds(d) \tag{5.9}$$

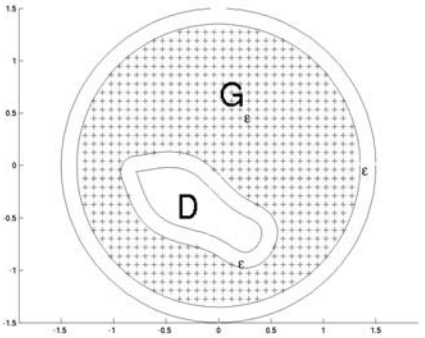


FIG. 3. *The integration domain $G_{0.15}$ for a nonconvex object.*

and

$$\nabla v_{\infty,z}(x) = -\frac{i\kappa}{4} H_1^{(1)}(\kappa|x-z|)\,\frac{x-z}{|x-z|}\,. \tag{5.10}$$

For the first integral, we have to solve the direct obstacle boundary value problem (2.1)–(2.3) with boundary values $f = 0$, $g = -v_{n,z}|_{\partial D}$ to obtain $w_{n,z}$ and with boundary data $f = 0$, $g = -\Phi(\cdot, z)|_{\partial D}$ to obtain $w_{\infty,z}$. In order to avoid numerical difficulties near the boundaries $\partial\Omega$ and ∂D, which arise from the representation of the function w by a combined layer potential, we choose an integration domain G_ε with $\varepsilon > 0$, as shown in Figure 3. Here G_ε is defined as the domain bounded by the surfaces

$$\partial D_\varepsilon := \{x - \varepsilon\nu(x) : x \in \partial D\}, \quad \varepsilon > 0, \tag{5.11}$$

and

$$\partial\Omega_\varepsilon := \{x - \varepsilon\nu(x) : x \in \partial\Omega\}, \quad \varepsilon > 0; \tag{5.12}$$

i.e., we exclude the "near-boundary" region around ∂D and $\partial\Omega$ for integration. Then, we can calculate the gradient of u in G_ε by applying the ∇ operator to the kernel functions of the integral operators S and K of the combined layer potential. In the 2D case, we obtain with $\Phi(x,y) = \frac{i}{4}H_0^{(1)}(\kappa|x-y|)$ and

$$\frac{d}{dz}H_n^{(1)}(z) = \frac{1}{2}\left(H_{n-1}^{(1)}(z) - H_{n+1}^{(1)}(z)\right) \tag{5.13}$$

the gradient

$$\begin{aligned}\nabla u(x) &= \int_{\partial\Omega_D} \nabla_x \frac{\partial\Phi(x,y)}{\partial\nu(y)}\varphi(y)\,ds(y) - i\eta\int_{\partial\Omega_D}\nabla_x\Phi(x,y)\varphi(y)\,ds(y)\\ &= \int_{\partial\Omega_D} k_1(x,y)\varphi(y)\,ds(y) - i\eta\int_{\partial\Omega_D} k_2(x,y)\varphi(y)\,ds(y),\end{aligned} \tag{5.14}$$

of u for all $x \in G_\varepsilon$ with the two kernel functions

$$\begin{aligned}k_1(x,y) = \frac{i\kappa}{4}\Bigg(&\frac{\kappa}{2}\left(H_0^{(1)}(\kappa|x-y|) - H_2^{(1)}(\kappa|x-y|)\right)(x-y)\cdot\nu(y)\frac{x-y}{|x-y|^2}\\ &+H_1^{(1)}(\kappa|x-y|)\frac{|x-y|^2\nu(y) - (x-y)\cdot\nu(y)(x-y)}{|x-y|^3}\Bigg)\end{aligned} \tag{5.15}$$

and

$$k_2(x,y) = -\frac{i\kappa}{4} H_1^{(1)}(\kappa|x-y|)\frac{x-y}{|x-y|}. \tag{5.16}$$

In the next section we give numerical examples, where we have evaluated the Ikehata functional (5.1) both with and without an approximation for the point source function $\Phi(\cdot,z)$ with the rectangular rule over the domains G_ε and D.

5.2. Numerical study of the Ikehata functional. In our numerical examples we will use a circular domain Ω of radius $r_\Omega = 2$ and a kite–shaped object D_K with boundary

$$\partial D_K := \{(-0.4\cos t - 0.3\cos 2t + 0.3, 0.6\sin t) : t \in [0,2\pi)\}, \tag{5.17}$$

as well as a boat-like obstacle D_B with boundary

$$\partial D_B := \{(0.25\cos t - 0.05\sin 4t, -0.75\sin t) : t \in [0,2\pi)\}. \tag{5.18}$$

Furthermore we will consider the translated versions

$$D_{K,v} := v + D_K = \{x + v : x \in D_K\} \tag{5.19}$$

and

$$D_{B,v} := v + D_B = \{x + v : x \in D_B\}, \tag{5.20}$$

with the translation vector $v = (-0.4, -0.7)$. The wave number will be either $\kappa = 1.0$ or $\kappa = 2.5$.

As described in the previous section, we will have to omit the "near-boundary" region around ∂D and $\partial\Omega$ when integrating $|w|^2$ and $|\nabla w|^2$ over Ω_D. For the proper choice of the parameter ε for the integration domain $G_\varepsilon \subset \Omega_D$, we first investigate the error between the true solution (and its gradient) of the direct obstacle boundary value problem and the solution (and gradient) given by the forward solver. To this end, we define with $x_i = (0,0) \in D$ and $x_e = (0,2.5) \in \mathbb{R}^2 \setminus \overline{\Omega}$ the solution

$$u_{\text{test}}(x) := \Phi(x,x_i) + \Phi(x,x_e) \tag{5.21}$$

to the Helmholtz equation in Ω_D, which satisfies the direct obstacle boundary value problem (2.1)–(2.3) with $f = u_{\text{test}}|_{\partial\Omega}$ and $g = u_{\text{test}}|_{\partial D}$. Tables 1 and 2 show the maximal error between the true solution u_{test} and the calculated solution u_{calc} on parallel surfaces $\partial\Omega_\varepsilon$ and ∂D_ε with distance $\varepsilon = 0.03$, n sampling points on the interior curve ∂D, and $3n$ sampling points on the exterior curve $\partial\Omega$ for the kite- and boat-shaped obstacles, respectively. The wave number for this numerical experiment is $\kappa = 1.0$.

Following this numerical example, we chose $n = 200$ sampling points on the interior and $3n = 600$ points on the exterior boundary of the domain Ω_D in the combined layer approach of the forward solver. Then, we can calculate the Ikehata functional with formula (5.1), where we substitute the integration domain Ω_D by G_ε with $\varepsilon = 0.03$. Figure 4 compares the Ikehata functional for the boat-shaped domain D_B and $D_{B,v}$ with the translation vector $v = (-0.4, -0.7)$ via three different calculations. First, the Ikehata functional is calculated with the D-t-N formulation (3.1), as described in sections 2–4. Second, to validate these results we calculated the

TABLE 1

The maximal error between the true solution u_{test} and the calculated solution u_{calc} and their gradients for a different number of sampling points n on a parallel surface to ∂D_K and 3n sampling points on a parallel surface to $\partial\Omega$ both in distance $\varepsilon = 0.03$.

n	50	100	200	300
$\max_{x\in\partial D_\varepsilon} \lvert u_{\text{test}} - u_{\text{calc}}\rvert$	0.090254	0.015958	0.000721	0.000034
$\max_{x\in\partial D_\varepsilon} \lvert\nabla u_{\text{test}} - \nabla u_{\text{calc}}\rvert$	5.901489	1.701766	0.145560	0.010361
$\max_{x\in\partial\Omega_\varepsilon} \lvert u_{\text{test}} - u_{\text{calc}}\rvert$	0.129217	0.012148	0.000129	0.000001
$\max_{x\in\partial\Omega_\varepsilon} \lvert\nabla u_{\text{test}} - \nabla u_{\text{calc}}\rvert$	10.96881	1.869363	0.039298	0.000633

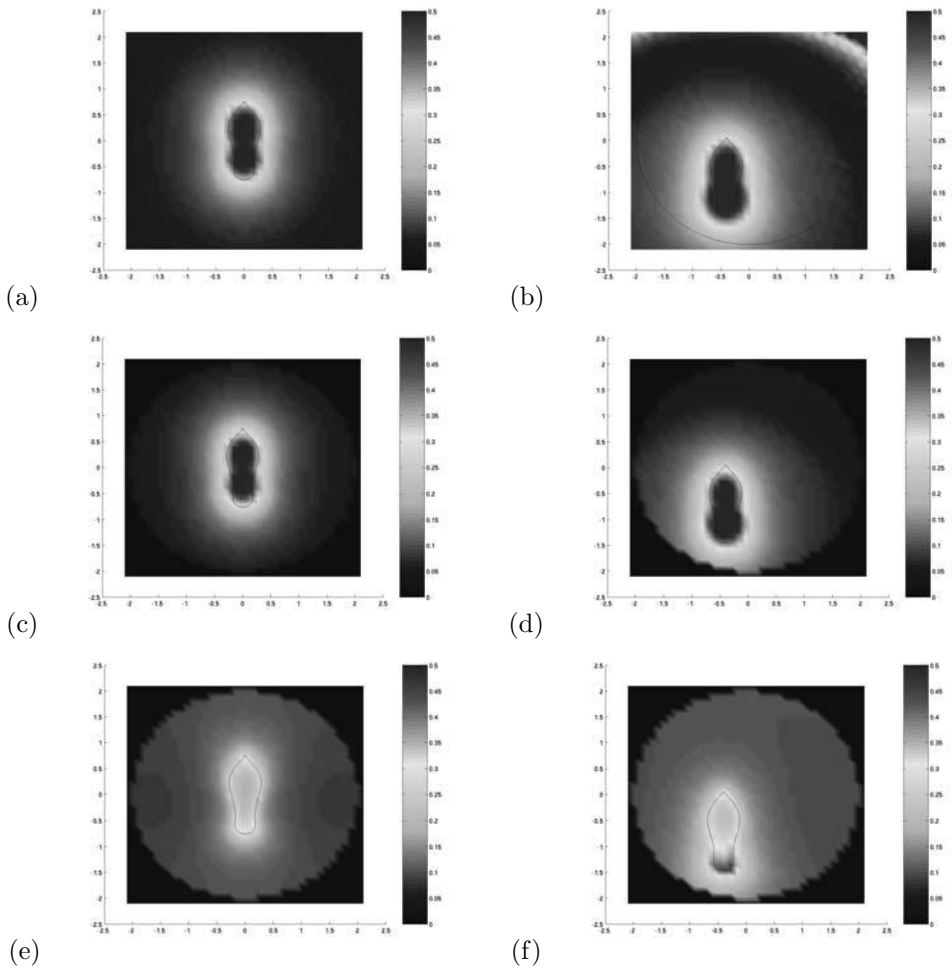


FIG. 4. (a), (c), *and* (e) *show the Ikehata functional with wave number* $\kappa = 1.0$ *for the domain* D_B, *whereas* (b), (d), *and* (f) *illustrate the Ikehata functional for the domain* $D_{B,v}$ *with translation vector* $v = (-0.4, -0.7)$. *The first row shows the Ikehata functional calculated with the D-t-N formulation* (3.1)*; the second row shows the Ikehata functional calculated via* (5.1). *Both rows have a regularization parameter of* $\alpha = 1e^{-06}$ *for the point source approximation. In the last row the Ikehata functional is plotted via* (5.1) *using the exact boundary data of a point source.*

Ikehata functional with (5.1) both with and without an approximation for the point source $\Phi(\cdot, x)$, as outlined in section 5. Note that the blow-up at the boundary can be observed only mildly due to its logarithmic behavior, i.e., numerically we observe

TABLE 2

The maximal error between the true solution u_{test} and the calculated solution u_{calc} and their gradients for a different number of sampling points n on a parallel surface to ∂D_B and 3n sampling points on a parallel surface to $\partial\Omega$ both in distance $\varepsilon = 0.03$.

n	50	100	200	300
$\max_{x\in\partial D_\varepsilon} \lvert u_{\text{test}} - u_{\text{calc}}\rvert$	0.066885	0.008551	0.000180	0.000004
$\max_{x\in\partial D_\varepsilon} \lvert\nabla u_{\text{test}} - \nabla u_{\text{calc}}\rvert$	4.970608	1.109627	0.045803	0.001476
$\max_{x\in\partial\Omega_\varepsilon} \lvert u_{\text{test}} - u_{\text{calc}}\rvert$	0.129226	0.012148	0.000129	0.000001
$\max_{x\in\partial\Omega_\varepsilon} \lvert\nabla u_{\text{test}} - \nabla u_{\text{calc}}\rvert$	10.96965	1.869363	0.039298	0.000633

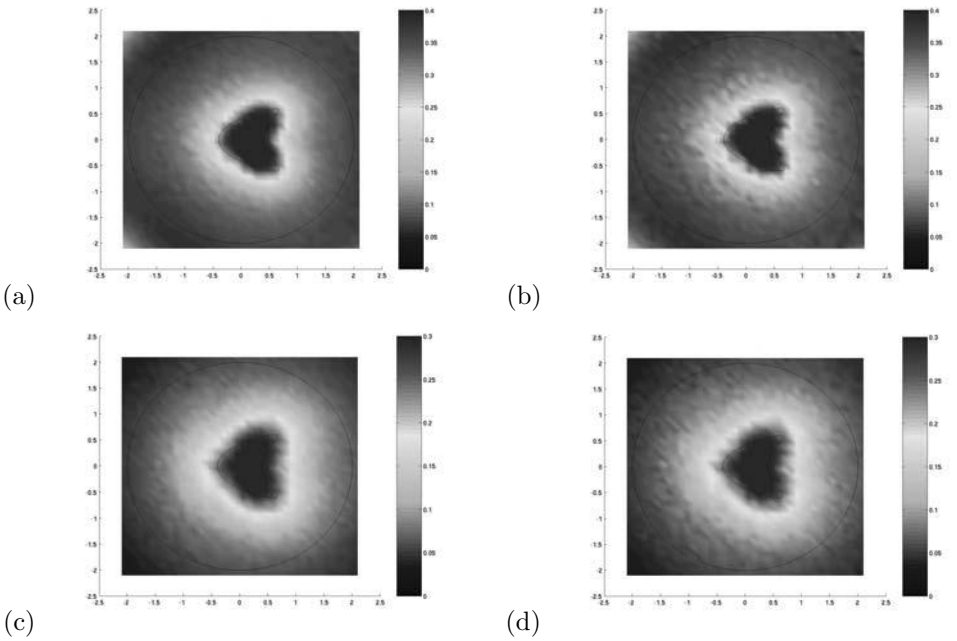


FIG. 5. (a)–(d) *show the Ikehata functional calculated with the D-t-N formulation* (3.1) *at a wave number $\kappa = 1.0$ for the domain D_K with a random error on the D-t-N map. The maximum ramdom error was* 5% (a), 10% (b), 20% (c), *and* 30% (d). *We have used the regularization parameter $\alpha = 0.0001$ for the reconstructions* (a) *and* (b) *and used $\alpha = 0.01$ for the reconstructions* (c) *and* (d).

only a modest increase of the function due to the numerical approximation and the use of sampling grids which do not come arbitrarily close to the boundary.

Finally, we present in Figure 5 reconstructions with random errors within 5% and 30% on the D-t-N map for the kite-shaped domain D_K.

REFERENCES

[1] J. CHENG, J. LIU, AND G. NAKAMURA, *Recovery of Boundaries and Types for Multiple Obstacles from the Far-Field Pattern*, EPrint Series, Department of Mathematics, Hokkaido University, Sapporo, Japan, 2005. Available online at http://eprints.math.sci.hokudai.ac.jp/archive/00000982/.

[2] D. COLTON AND R. KRESS, *Inverse Acoustic and Electromagnetic Scattering Theory*, 2nd ed., Appl. Math. Sci. 93, Springer-Verlag, Berlin, 1998.

[3] K. ERHARD AND R. POTTHAST, *The point source method for reconstructing an inclusion from boundary measurements in electrical impedance tomography and acoustic scattering*, Inverse Problems, 19 (2003), pp. 1139–1157.

[4] M. Ikehata, *Reconstruction of an obstacle from the scattering amplitude at a fixed frequency*, Inverse Problems, 14 (1998), pp. 949–954.
[5] M. Ikehata, *Reconstruction of the shape of the inclusion by boundary measurements*, Comm. Partial Differential Equations, 23 (1998), pp. 1459–1474.
[6] M. Ikehata, *Reconstruction of obstacle from boundary measurements*, Wave Motion, 30 (1999), pp. 205–223.
[7] R. Kress, *On the numerical solution of a hyper-singular integral equation in scattering theory*, J. Comput. Appl. Math., 61 (1995), pp. 345–360.
[8] A.-W. Maue, *Zur Formulierung eines allgemeinen Beugungsproblems durch eine Integralgleichung*, Z. Physik, 126 (1949), pp. 601–618.
[9] R. Potthast, *Point Sources and Multipoles in Inverse Scattering Theory*, Chapman & Hall/CRC Research Notes in Mathematics 427, Chapman & Hall/CRC, Boca Raton, FL, 2001.
[10] R. Potthast, *Sampling and probe methods—an algorithmical view*, Computing, 75 (2005), pp. 215–236.

SIAM J. SCI. COMPUT.
Vol. 28, No. 5, pp. 1613–1633
© 2006 Society for Industrial and Applied Mathematics

BENEFITS OF IEEE-754 FEATURES IN MODERN SYMMETRIC TRIDIAGONAL EIGENSOLVERS*

OSNI A. MARQUES†, E. JASON RIEDY‡, AND CHRISTOF VÖMEL†

Abstract. Bisection is one of the most common methods used to compute the eigenvalues of symmetric tridiagonal matrices. Bisection relies on the *Sturm count*: For a given shift σ, the number of negative pivots in the factorization $T - \sigma I = LDL^T$ equals the number of eigenvalues of T that are smaller than σ. In IEEE-754 arithmetic, the value ∞ permits the computation to continue past a zero pivot, producing a correct Sturm count when T is unreduced. Demmel and Li showed [*IEEE Trans. Comput.*, 43 (1994), pp. 983–992] that using ∞ rather than testing for zero pivots within the loop could significantly improve performance on certain architectures. When eigenvalues are to be computed to high relative accuracy, it is often preferable to work with LDL^T factorizations instead of the original tridiagonal T. One important example is the MRRR algorithm. When bisection is applied to the factored matrix, the Sturm count is computed from LDL^T which makes differential stationary and progressive qds algorithms the methods of choice. While it seems trivial to replace T by LDL^T, in reality these algorithms are more complicated: In IEEE-754 arithmetic, a zero pivot produces an overflow followed by an invalid exception (NaN, or "Not a Number") that renders the Sturm count incorrect. We present alternative, safe formulations that are guaranteed to produce the correct result. Benchmarking these algorithms on a variety of platforms shows that the original formulation without tests is always faster provided that no exception occurs. The transforms see speed-ups of up to 2.6× over the careful formulations. Tests on industrial matrices show that encountering exceptions in practice is rare. This leads to the following design: First, compute the Sturm count by the fast but unsafe algorithm. Then, if an exception occurs, recompute the count by a safe, slower alternative. The new Sturm count algorithms improve the speed of bisection by up to 2× on our test matrices. Furthermore, unlike the traditional tiny-pivot substitution, proper use of IEEE-754 features provides a careful formulation that imposes no input range restrictions.

Key words. IEEE-754 arithmetic, NaN arithmetic, IEEE-754 performance, differential qds algorithms, MRRR algorithm, LAPACK

AMS subject classifications. 15A18, 15A23

DOI. 10.1137/050641624

1. Introduction.

In the early 1990s, Demmel and Li demonstrated that handling exceptions carefully can improve the performance of certain numerical algorithms [8]. One such algorithm is bisection using Sturm counts to compute the eigenvalues of a symmetric tridiagonal matrix T. A Sturm count finds the number of eigenvalues less than a given shift parameter σ. Sturm count calculations can encounter a zero pivot while factoring $T - \sigma I = LDL^T$. A conventional bisection

*Received by the editors September 30, 2005; accepted for publication (in revised form) March 15, 2006; published electronically September 29, 2006. This work has been supported by a grant from the National Science Foundation (Cooperative Agreement ACI-9619020) and from the Department of Energy's SciDAC program (DE FC02 01ER25478). The authors acknowledge the contribution from Intel Corporation, Hewlett-Packard Corporation, IBM Corporation, and the National Science Foundation grant EIA-0303575 for providing hardware and software to the CITRIS Cluster used in producing these research results. The authors also thank Cray Inc. for giving them access to the X1, and NERSC for their use of the IBM SP3. This work was performed by an employee of the U.S. Government or under U.S. Government contract. The U.S. Government retains a nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or allow others to do so, for U.S. Government purposes. Copyright is owned by SIAM to the extent not limited by these rights.

http://www.siam.org/journals/sisc/28-5/64162.html

†Lawrence Berkeley National Laboratory, 1 Cyclotron Road, MS 50F-1650, Berkeley, CA 94720 (oamarques@lbl.gov, cvoemel@lbl.gov).

‡Computer Science Division, University of California, Berkeley, CA 94720 (ejr@cs.berkeley.edu).

implementation such as LAPACK's DSTEBZ substitutes a small number for zero and continues the computation [3]. This procedure has two disadvantages. Testing imposes a performance penalty, and ensuring correctness requires a range restriction on the input data.

IEEE-754 arithmetic [4, 21] provides the [illegible] ∞ (infinity) to continue past the exceptions of overflow and division by zero. This quantity propagates like any other floating-point value, ideally at full speed. In [8], this capability of IEEE-754 arithmetic allows a faster Sturm count and hence a faster implementation of tridiagonal bisection. The faster algorithm also avoids the input restrictions, removing both disadvantages.

Since [8], there has been significant progress in algorithms for the symmetric eigenvalue problem. In particular, LAPACK now includes DSTEGR, an implementation of the MRRR algorithm [23, 24, 25, 26, 9]. Two key concepts of MRRR are its use of shifted LDL^T factorizations instead of the original tridiagonal matrix T, and the [illegible] that transform one shifted factorization into another. The MRRR algorithm applies these transformations to produce a suitable relatively robust representation (RRR) for each eigenvalue. This RRR is an appropriately shifted LDL^T factorization from which the MRRR algorithm computes a corresponding eigenvector with guaranteed small residual and numerical orthogonality. In this process, the algorithm also refines some eigenvalues by bisection, again using the differential qds algorithms.

This paper focuses on using the qds algorithms in bisection. Originally, the differential qd (dqd) algorithm was proposed by Rutishauser [28, 29]. Parlett and Fernando [14, 15, 22] discovered the shifted variant (dqds) as an improvement to Demmel and Kahan's zero shift QR [7] for computing singular values of a bidiagonal matrix to high relative accuracy; see also [27].

Applying bisection to the factored matrix LDL^T using the stationary and progressive differential qds transformations is more complicated than traditional bisection directly applied to the tridiagonal T. A qds implementation in IEEE-754 arithmetic can encounter not only ∞ but also an invalid operation ∞/∞ (producing NaN, or "Not a Number") that renders the Sturm count invalid. Demmel and Li [8] suggest a paradigm for algorithms where exceptions may invalidate the results of calculations that otherwise appear simple:

(i) Use a fast (but possibly incorrect) algorithm to compute a result.
(ii) Assess the correctness of the computed result.
(iii) If the answer is not correct, use a slower and more careful algorithm to recompute the result.

This paper applies this exception-handling paradigm to qds transformations and quantifies the resulting performance on today's computer architectures. This allows us to select the best performing and most robust implementation of qds transformations for use in bisection and the MRRR algorithm.

The paper is organized as follows. Section 2 gives an overview of the various formulations of Sturm counts via qds transformations. We analyze three always correct alternatives to the basic version. One of the alternative formulations computes a correct Sturm count without any additional input range restrictions.

Section 3 is the heart of this paper. We benchmark our algorithms for input data that results in both exceptional and unexceptional execution to quantify the benefits of using IEEE-754 features and the penalties for encountering exceptions. As long as no exception occurs, the basic version is always faster. Overall, the careful variants' times are around 1.12× the time of the basic version. We refer to this as a speed-

up of 1.12 when comparing the faster basic version with the slower careful variants. When an exception occurs, the combined plain/careful computation times are about 1.63× that of the careful loop alone, which we call a slow-down of 1.63. There is one remarkable exception, the Pentium 4 Xeon, whose performance drops by factors of 18–60× in the exceptional case.

Section 4 compares these formulations when applied within bisection on real world test examples. The median speed-up of the fast algorithm when running bisection to determine the eigenvalues fairly accurately is moderate, around 1.08. The Cray X1 greatly benefits from moving exceptional tests out of the inner loop, seeing speed-ups from 1.2 to almost 2.

The potential for huge slow-downs on aberrant architectures implies that our paradigm may not be suitable for portable code. However, section 5 describes a technique to limit the penalty on almost all cases: blocking. The Sturm count computation can be divided into blocks, and the NaN test is performed on each block. Thus, if an NaN occurs, only the computation of the previous block need be repeated.

The appendix gives a short overview of important implementation details in our algorithms.

2. Sturm counts with and without IEEE-754 arithmetic. Bisection is one of the most popular methods for computing the eigenvalues of a symmetric tridiagonal matrix. It can be used to find k selected eigenvalues with $\mathcal{O}(kn)$ work given a tridiagonal matrix of dimension n. Furthermore, it is easily parallelizable.

Bisection is based on Sturm counts. This section presents various methods for computing Sturm counts. We require that the input matrix be unreduced; that is, no entry on the super- or subdiagonal is zero. This implies that the eigenvalues are simple. Additionally, we assume no input is NaN or $\pm\infty$.

2.1. Sturm counts for a tridiagonal matrix. Sylvester's law of inertia states that the (shifted) symmetric tridiagonal matrix $T - \sigma I = LDL^T$, with L being unit-bidiagonal and D being diagonal, has the same number of positive, negative, and zero eigenvalues as D; see, for example, [16].

This result can be used to find, by suitable variation of the shifts, small enclosing intervals for each eigenvalue of T. Starting with an initial interval $[V_L, V_U]$ containing the eigenvalue, we compute the number of negative pivots at the midpoint $(V_L+V_U)/2$. If this number is smaller than the index of the wanted eigenvalue, the eigenvalue lies in the right half of the interval; otherwise it lies in the left half. This bisection process can be repeated until an interval of suitably small size is found. Its implementation in finite precision arithmetic is correct provided that the Sturm count is monotonic with σ; see [19, 6].

The well-known recurrence for the pivots in D is given by

$$d(i) = (T(i,i) - \sigma) - \frac{T(i,i+1)^2}{d(i-1)}. \tag{2.1}$$

In finite precision arithmetic, the pivot $d(i-1)$ may be tiny or exactly zero. In this case, the succeeding pivot $d(i)$ overflows (provided that $T(i,i+1) \neq 0$).

A possible cure to this situation was first developed in [19]. At the beginning of the algorithm, a minimum pivot threshold [illegible] is computed. It represents the smallest divisor so that $T(i,i+1)^2/\text{pivmin}$ does not overflow. Then, whenever $|d(i-1)| <$ pivmin, the algorithm replaces $d(i-1)$ by pivmin. (The convention in LAPACK's DSTEBZ is actually to replace the pivot by negative pivmin.) This substitution limits

the input range. To prevent overflows, the inputs must satisfy $|T(i,j)| \leq \Omega^{1/2}\eta^{1/4}$, where Ω is the arithmetic's overflow threshold and η is the smallest positive normalized number.

IEEE-754 arithmetic allows us to avoid the pivmin threshold [8]. With $d(i-1) = 0$, the next pivot becomes $-\infty$, and the computation continues normally afterwards with $d(i+1) = T(i+1, i+1) - \sigma$. In [8], the authors report a speed-up of 1.2 to 1.3 of Sturm counts using the IEEE-754 version over the version based on the pivmin threshold. The impact on bisection is reported to be a speed-up of 1.14 to 1.24. Also, the IEEE-754 version does not impose any range restrictions.

2.2. Sturm counts for a tridiagonal matrix in factored form LDL^T. Instead of applying Sylvester's law of inertia to a tridiagonal matrix T, it also can be applied to its LDL^T factorization. This idea leads to differential qds transformations that are introduced in the following. They are of great importance to the MRRR algorithm [23, 24, 25, 26, 9] that requires the reliable computation of Sturm counts from shifted LDL^T factorizations instead of the original tridiagonal matrix.

In order to compute Sturm counts, MRRR uses two principal algorithms, the stationary factorization $LDL^T - \sigma I = L_+ D_+ L_+^T$ and the progressive factorization $LDL^T - \sigma I = U_- D_- U_-^T$. These are variants of an algorithm originally proposed by Rutishauser [28, 29] and are stable in a mixed sense; see [26].

While a Sturm count on T does not look very different from one on LDL^T, there are subtle and important differences between the two approaches that are summarized in the following table.

	Tridiagonal T	Factored $T = LDL^T$
Sturm sequence algorithm	$T - \sigma I = LDL^T$	Differential qds algorithms
Possible exceptional values	∞	∞, NaN
Is the final Sturm count valid?	Always	If NaN has not occurred

Algorithm 2.1 presents both the stationary (a) and progressive (b) algorithms. The variable count holds the Sturm count. Note the following:

(i) Both algorithms use the quantities $d(i)$ and $l(i)^2 d(i)$. If the Sturm count is repeated several times as in bisection, precomputing $lld(i) \equiv l(i)^2 d(i)$ is more efficient than repeatedly computing $l(i)^2 d(i)$. We use $lld(i)$ throughout the rest of the paper.

(ii) For the purposes of the Sturm sequence, it is not necessary to store the pivots $d_+(i)$ and $d_-(i)$.

(iii) The division in both algorithms is probably the most costly operation. We place the sign tests after the division to encourage optimizations that overlap the two operations.

Sturm counts on the factored matrix LDL^T via differential stationary or progressive qds transforms can produce an NaN during the computation. In the case of differential stationary qds, if $d_+(i-1) = 0$, then $|d_+(i)| = \infty$ and $d_+(i+1) = \text{NaN}$; see also Table 1(a). For the differential progressive qds, if $d_-(i+1) = 0$, then $|d_-(i)| = \infty$ and $d_-(i-1) = \text{NaN}$; see also Table 1(b).

If an NaN occurs, that NaN propagates through all subsequent computations. The resulting Sturm count is invalid in this case; the computation needs to be repeated with an NaN-free variant of the respective algorithm. This is in contrast to a Sturm sequence using T that is robust even when ∞ occurs.

TABLE 1
Variable values after a zero pivot is encountered in step $i-1$ of the stationary and the progressive variants of Algorithm 2.1.

	$i-1$	i	$i+1$
t	$-d(i-1)$	$\pm\infty$	NaN
d_+	0	$\pm\infty$	NaN

(a) Stationary qds

	$i-1$	i	$i+1$
d_-	0	$\pm\infty$	NaN
p	$\pm\infty$	NaN	NaN

(b) Progressive qds

TABLE 2
Range restrictions and possible exceptional values for the alternative formulations of the differential qds algorithms. The quantity Ω is the overflow threshold, η is the smallest positive, normalized number, and ε denotes the smallest positive number such that $1+\varepsilon$ does not evaluate to 1.

Algorithm	Exceptional values	Range restrictions
2.2: pivmin	*none*	$\lvert T(i,j)\rvert \le \Omega^{1/2}\eta^{1/4}$
2.3: $\infty/\infty \Rightarrow 1$	$\pm\infty$	*none*
2.4: $\pm\infty \Rightarrow \pm\Omega$	$\pm\infty$	$\lvert d(i)\rvert < \Omega\,\varepsilon$

ALGORITHM 2.1. The differential stationary and progressive qds algorithms compute $L_+D_+L_+^T$(stationary) $= U_-D_-U_-^T$(progressive) $= LDL^T - \sigma I$ and count the number of negative pivots. The diagonal elements of D are $d(1),\ldots,d(n)$, and the products $D(i,i)L(i,i+1)^2$ are $lld(1),\ldots,lld(n-1)$. (The off-diagonal entries of L are not used explicitly; they occur only in the products $D(i,i)L(i,i+1)^2$.) The quantity [illegible] denotes the number of eigenvalues of LDL^T that are less than σ.

```
negcount = 0
t = -σ
for i = 1 : n - 1 do
   d+(i) = d(i) + t
   t = (t/d+(i)) * lld(i) - σ
   Increase negcount if d+(i) < 0
end for
d+(n) = d(n) + t
Increase negcount if d+(n) < 0
```

(a) Stationary qds

```
negcount = 0
p = d(n) - σ
for i = n - 1 : 1 do
   d-(i + 1) = lld(i) + p
   p = (p/d-(i + 1)) * d(i) - σ
   Increase negcount if d-(i + 1) < 0
end for
d-(1) = p
Increase negcount if d-(1) < 0
```

(b) Progressive qds

2.3. NaN-free variants of differential qds transforms. There are different possible approaches to avoiding invalid operations in the differential stationary and progressive qds algorithms. In this section, we discuss three such variants of Algorithm 2.1 that are based on appropriate substitutions. A summary of the possible exceptional values and range restrictions of the various alternative formulations is given in Table 2. These are explained in the later sections.

2.3.1. Substitution of pivmin for tiny pivots. The first approach mimics the tridiagonal Sturm count procedure presented in section 2.1. Algorithm 2.2 replaces tiny and zero pivots with a minimum threshold [illegible] to prevent division by a zero pivot. It entirely avoids exceptional values. The choice of [illegible] is justified by [19] and used in LAPACK. In our tests, we use pivmin $= \eta \cdot \max_{i=1,\ldots,n-1} T(i,i+1)^2$. To avoid overflows, the initial data must satisfy $|T(i,j)| \le \Omega^{1/2}\eta^{1/4}$.

ALGORITHM 2.2. Substituting pivmin for very small pivots.

```
negcount = 0
t = -σ
for i = 1 : n - 1 do
   d+(i) = d(i) + t
   if |d+(i)| < pivmin then
      d+(i) = -pivmin
   end if
   t = (t/d+(i)) * lld(i) - σ
   Increase negcount if d+(i) < 0
end for
d+(n) = d(n) + t
Increase negcount if d+(n) < 0
```

(a) Stationary qds

```
negcount = 0
p = d(n) - σ
for i = n - 1 : 1 do
   d-(i + 1) = lld(i) + p
   if |d-(i + 1)| < pivmin then
      d-(i + 1) = -pivmin
   end if
   p = (p/d-(i + 1)) * d(i) - σ
   Increase negcount if d-(i + 1) < 0
end for
d-(1) = p
Increase negcount if d-(1) < 0
```

(b) Progressive qds

2.3.2. Substitution of ∞/∞ by 1. The second approach mimics the tridiagonal Sturm count by allowing ∞ but preventing the subsequent NaN. For the differential stationary qds algorithm (Algorithm 2.1(a)), substituting an alternate value for ∞/∞ can avoid producing an NaN. Using l'Hôpital's limit formula, we find $\lim_{t\to\infty} t/d_+(i) = \lim_{t\to\infty} t/(d(i)+t) = 1$. The same argument holds for the quotient $p/d_-(i+1)$ in the differential progressive qds algorithm (Algorithm 2.1(b)). The qds variants in Algorithm 2.3 explicitly substitute 1 for ∞/∞. Note that the infinite t (or p) from a tiny or zero pivot produces a pivot d_+ (or d_-) with the same sign; thus the substitution is always 1. This variant does not restrict the input range but does produce $\pm\infty$ values.

ALGORITHM 2.3. Substituting 1 for ∞/∞.

```
negcount = 0
t = -σ
for i = 1 : n - 1 do
   d+(i) = d(i) + t
   if |t| = ∞ and |d+(i)| = ∞ then
      q = 1
   else
      q = (t/d+(i))
   end if
   t = q * lld(i) - σ
   Increase negcount if d+(i) < 0
end for
d+(n) = d(n) + t
Increase negcount if d+(n) < 0
```

(a) Stationary qds

```
negcount = 0
p = d(n) - σ
for i = n - 1 : 1 do
   d-(i + 1) = lld(i) + p
   if |p| = ∞ and |d-(i + 1)| = ∞ then
      q = 1
   else
      q = (p/d-(i + 1))
   end if
   p = q * d(i) - σ
   Increase negcount if d-(i + 1) < 0
end for
d-(1) = p
Increase negcount if d-(1) < 0
```

(b) Progressive qds

2.3.3. Substitution of overflow and divide-by-zero by $\pm\Omega$. Following a suggestion of Kahan [19, 20], we present an alternative version which emulates saturation arithmetic, an arithmetic where all overflows produce the largest representable

number rather than ∞. Algorithm 2.4 replaces an infinite t or p with the correctly signed largest representable number $\pm\Omega$. As long as the subsequent addition in the next iteration does not overflow, division with the new pivot evaluates to 1 as in Algorithm 2.3.

In true saturation arithmetic, all overflows would produce a correctly signed Ω as a result. Because we emulate saturation arithmetic only when calculating t (or p), the subsequent addition overflows and yields an incorrect pivot $d_+(i+1)$ (or $d_-(i+1)$) if $d(i+1) \geq \Omega\varepsilon$. Thus we must restrict the input range slightly or add additional tests to fully implement saturation arithmetic. We chose the former because practical examples do not approach the input restriction $d(i) < \Omega\varepsilon$. This variant also produces infinities as intermediate values. The function application sign(t) in Algorithm 2.4 returns 1 if $t = +\infty$ and -1 if $t = -\infty$.

ALGORITHM 2.4. Substituting the largest magnitude, finite number Ω for infinite quotients (correctly signed).

```
negcount = 0
t = -σ
for i = 1 : n - 1 do
   d+(i) = d(i) + t
   t = (t/d+(i)) * lld(i) - σ
   if |t| = ∞ then
      t = sign(t) * Ω
   end if
   Increase negcount if d+(i) < 0
end for
d+(n) = d(n) + t
Increase negcount if d+(n) < 0
```

(a) Stationary qds

```
negcount = 0
p = d(n) - σ
for i = n - 1 : 1 do
   d-(i + 1) = lld(i) + p
   p = (p/d-(i + 1)) * d(i) - σ
   if |p| = ∞ then
      p = sign(p) * Ω
   end if
   Increase negcount if d-(i + 1) < 0
end for
d-(1) = p
Increase negcount if d-(1) < 0
```

(b) Progressive qds

3. A benchmark of the various differential qds algorithms. We devise two benchmarks to compare the various algorithms. For a given tridiagonal matrix V_n in factored form LDL^T, both benchmarks compute differential qds factorizations of $LDL^T - \sigma I$. In the first case, we choose σ such that no zero pivot (and hence no IEEE exception) occurs. In the second case, we choose σ to force an exception. These benchmarks allow us to compare the performance of the algorithms in the unexceptional case and to quantify the penalties when an IEEE exception occurs. For a given dimension n, the matrix V_n is a tridiagonal with all off-diagonal entries equal to one; its diagonal consists of the vector $1 : n$.

3.1. Benchmark details. The occurrence of NaN always follows a zero (or tiny) pivot in $LDL^T - \sigma I$; see Table 1(a) and (b). Furthermore, a pivot can be zero only when the shift is chosen as an eigenvalue of a leading submatrix in the stationary case, and as an eigenvalue of a trailing submatrix in the progressive case.

For the first benchmark, the shift σ is chosen well outside the Gershgorin bounds. This guarantees that no exceptional operation occurs.

The second benchmark must induce an invalid operation. For the differential stationary qds algorithms, we choose as the shift the first diagonal element $d(1)$, and an NaN occurs in step 2. For the differential progressive qds algorithm, setting

TABLE 3
Platforms and timers used for testing.

Architecture	Clock (MHz)	OS	Compiler	Timer
Athlon-XP	1200	GNU/Linux	gfortran (Debian 4.0.1-6) -O3 -fno-trapping-math -march=athlon-xp	`gettimeofday`
Itanium 2	1300	Linux	Intel ifort 9.0 -O2	`gettimeofday`
Pentium 4 Xeon, SSE	3000	Linux	Intel ifort 9.0 -O3 -xP	`gettimeofday`
Pentium 4 Xeon, x87	3000	Linux	Intel ifort 9.0 -O3 -xP -mp	`gettimeofday`
MIPS R12000	600	IRIX	SGI MIPS pro 7.3.1.3m -O2	`ETIME`
Power3	375	AIX	IBM xlf90 v8r11 -O3	PAPI
UltraSparc 2i	650	Solaris	SUN f90 forte 7.0 -O4	PAPI
Cray X1	800	UNICOS	Cray ftn 5.4.0.4 -O2	`cpu_time`
Opteron	2200	Linux	Pathscale pathf90 2.1 -O3	`cpu_time`

$lld(n-1) = -(d(n) - \sigma)$ forces the pivot $d_-(n)$ to zero for any shift σ. However, in order to prevent the matrix from becoming reducible, we must choose $\sigma \neq d(n)$. A reducible matrix would produce NaNs even in the alternative formulations of the progressive algorithm. The choices $\sigma = d(n)/2.0$ and $lld(n-1) = -d(n)/2.0$ produce an NaN regardless of any compiler optimizations or rounding settings.

We run each benchmark in double precision for matrices of dimensions from 500 to 6000 by steps of 500. Individual Sturm counts compute very quickly, so we time a loop of 50000 Sturm counts on the same data. Table 3 summarizes the architectures, compilers, and timers used for our experiments.

All tested platforms support the default IEEE-754 execution mode, producing and propagating ∞ and NaN without user-visible traps. We tested both the Pentium 4 Xeon's 80-bit x87 floating-point unit and its 64-bit SSE3 floating-point unit. We tested only the SSE2 unit on the Opteron. SSE2 and SSE3 floating-point units can perform vector operations where each operand is a pair of double-precision values. The Intel and Pathscale compilers did not vectorize the Sturm counting loops and used only the first 64-bit entry in each vector register. The Athlon platform tested supports only SSE1, and thus our tests use its x87 floating-point unit.

3.2. Benchmark results. Both benchmarks time the same Sturm count routines. Algorithms 2.2, 2.3, and 2.4 each are applied to $T = LDL^T$ for the two shifts above. To test the paradigm of computing quickly and fixing failures, we time three additional versions. Each runs the basic qds loop in Algorithm 2.1, tests the final pivot for NaN, and recomputes with runs of Algorithms 2.2, 2.3, or 2.4 if the pivot is NaN.

Figures 1, 2, and 3 present the timings for every algorithm and platform. The times are normalized into CPU cycles per entry examined by

$$\text{CPU cycles per entry} = \frac{\text{total time}}{50000 \text{ iterations} \cdot n \text{ entries}} \cdot \text{clock speed in Hz} \tag{3.1}$$

given the CPU clock speeds in Table 3. Note that the figures have different horizontal scales, and that Figure 3 provides two views of the Pentium 4 Xeon's performance when using the x87 floating-point unit. The markers have different shapes to distinguish both between progressive and stationary loops and between plain and exceptional execution. The dashed vertical lines denote the range of division latencies given in platform documentation and tuning guides [1, 17, 18, 30, 32, 31, 2]. No such

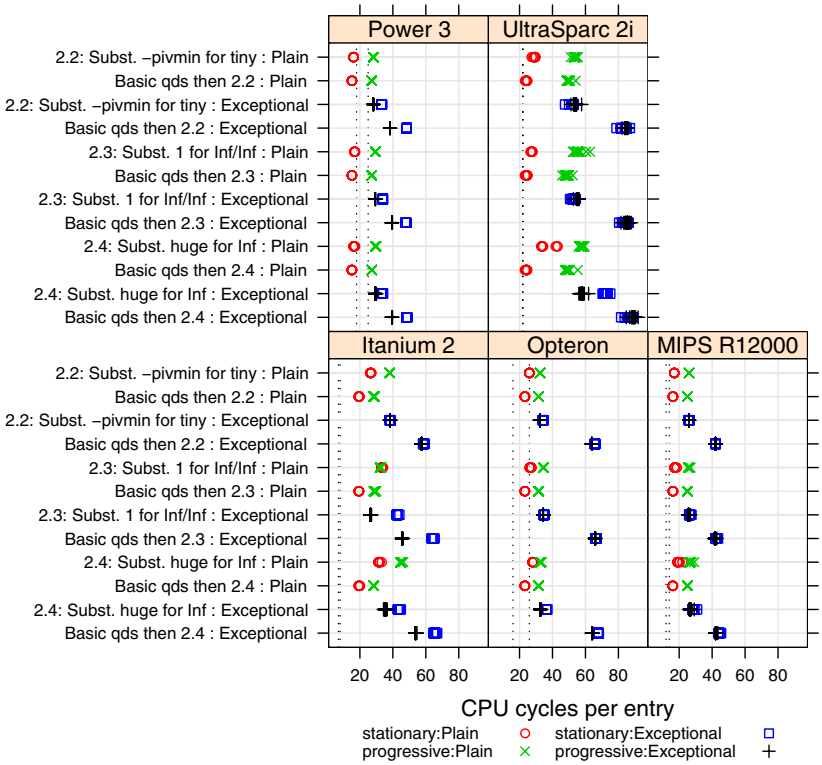


FIG. 1. *CPU cycles per entry for each qds algorithm. Vertical dashed lines show the range of documented division latencies, if known.*

information is available for the Cray X1.

3.3. Examining the benchmark results. On an unexceptional run, a basic qds Sturm count achieves a speed-up of 1.12 over the careful algorithms of section 2, where speed-up is measured by the time of the careful algorithm divided by the time of the basic qds algorithm on the same inputs. When exceptions occur, the median slow-down of the combined basic/careful qds loops over the careful loop alone is 1.63. We measure slow-down as the time of the combined loop divided by the time of the careful loop on the same inputs. The only slow-downs over 2 occur for the x87 unit on the Pentium 4 Xeon, which drastically penalizes exceptional values and suffers slow-downs from 18.93 to 66.53. No one careful algorithm's performance dominates the others, but we recommend that implementors use Algorithm 2.3 ($\infty/\infty \Rightarrow 1$) because it imposes no special range restrictions on the input data.

. Figure 4 compares the basic qds loops' times to the times of their careful counterparts when no exceptions occur. As expected, the basic qds loop runs faster. The speed-up of the basic qds loop over the corresponding careful algorithms across all platforms is given in Table 4.

Note that the stationary loops sometimes run more slowly than the progressive loops; some platforms appear to prefer sequential forward memory access over sequential reverse access. And the x87 unit on the Pentium 4 Xeon prefers Algorithm 2.2

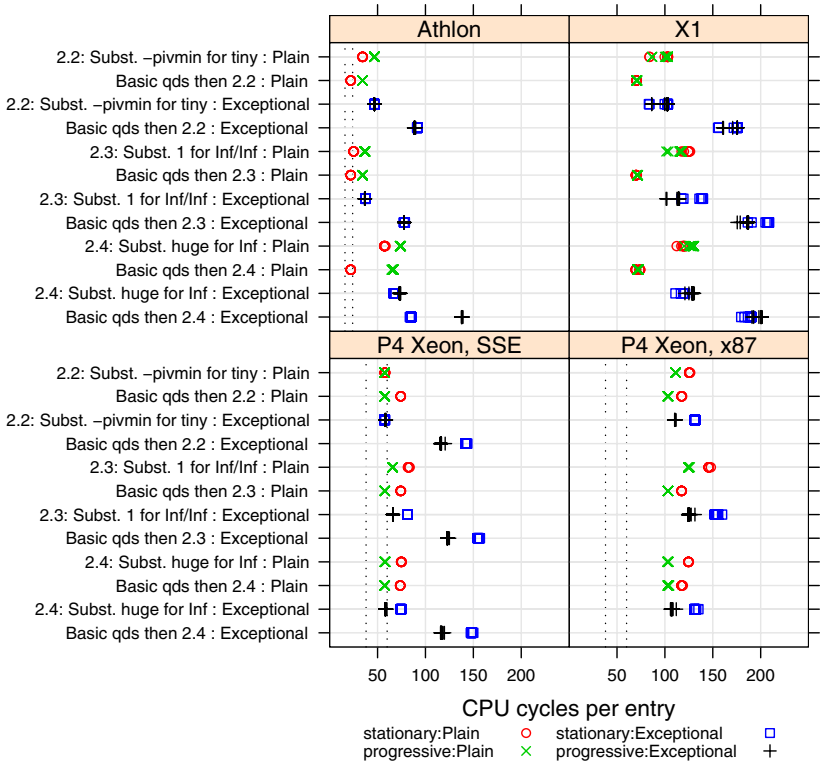


FIG. 2. *CPU cycles per entry for each qds algorithm for platforms needing a wider scale than those of Figure* 1. *Xeon values with the* x87 *unit are clipped to fit. Vertical dashed lines show the range of documented division latencies, if known.*

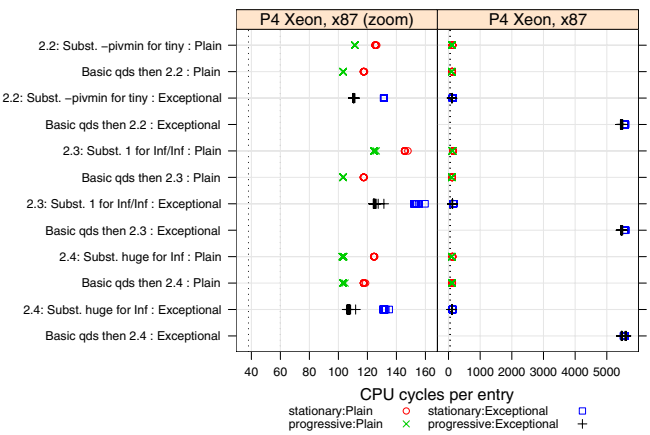


FIG. 3. *CPU cycles per entry for each qds algorithm on the Xeon with the* x87 *unit. Vertical dashed lines show the range of documented division latencies, if known.*

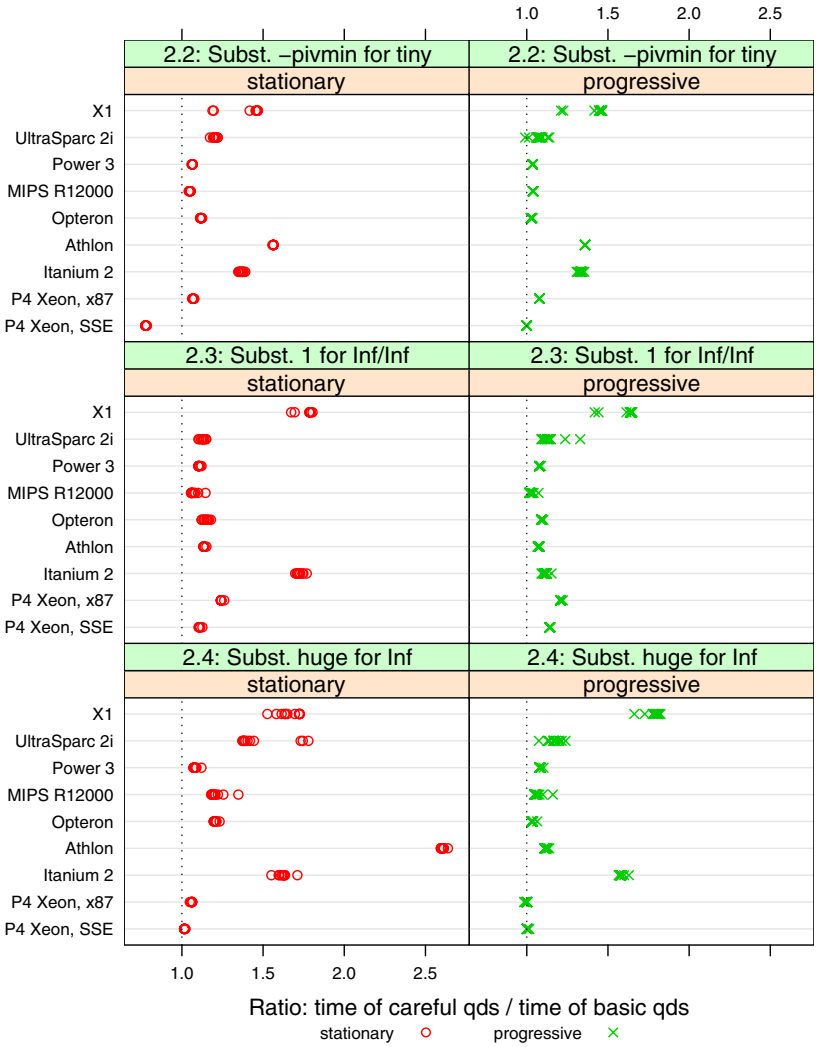


FIG. 4. *As long as no exception occurs, the basic qds loop almost always runs faster than the careful loops that test for exceptional operations. A point at* 1.2 *means the careful loop's running time was* 1.2 *times the basic loop's time; the basic loop sees a speed-up of* 20%.

TABLE 4

Ratio of careful loops' times to corresponding basic loops' times, or the speed-up of the basic loop over the careful variants, for unexceptional inputs across all platforms and both progressive and stationary qds.

Variant	Min	1st Quartile	Median	Mean	3rd Quartile	Max
All variants	0.775	1.063	1.118	1.235	1.355	2.638
2.2: pivmin	0.775	1.039	1.078	1.166	1.355	1.565
2.3: $\infty/\infty \Rightarrow 1$	1.016	1.091	1.129	1.218	1.206	1.805
2.4: $\pm\infty \Rightarrow \pm\Omega$	0.984	1.055	1.133	1.321	1.580	2.638

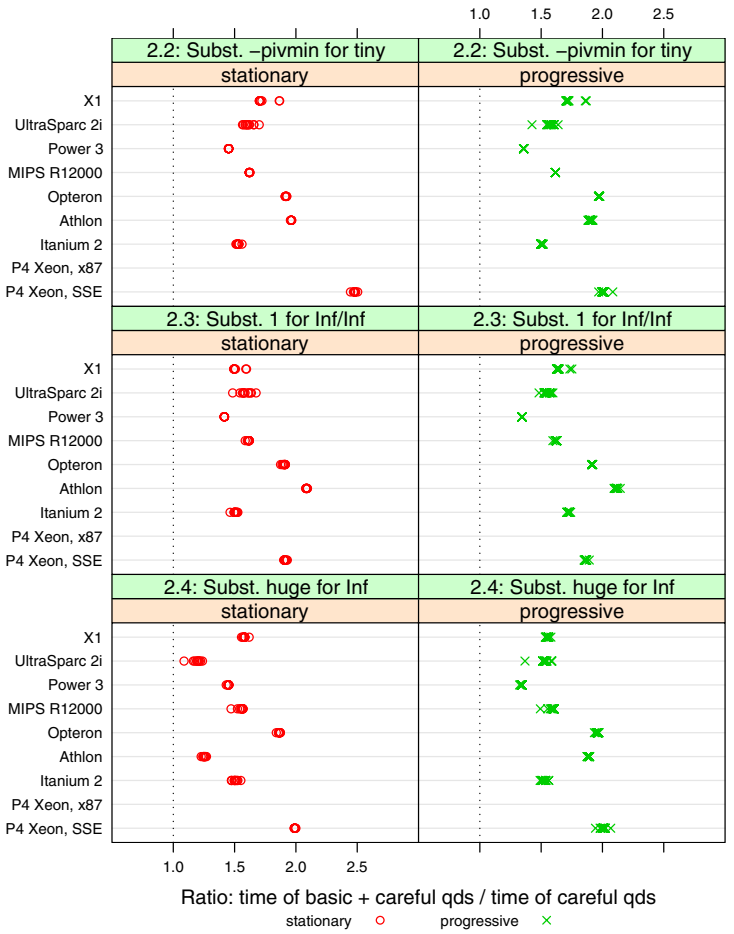


FIG. 5. *When an exception occurs, however, many platforms suffer a penalty for carrying the NaN through $O(n)$ operations. Here, a point at* 1.5 *means that the basic loop followed by the careful loop ran* 1.5 *times as long as the careful loop alone. The values for the* x87 *Xeon runs are clipped.*

(pivmin) over Algorithm 2.1's simple qds loop.

. Figure 5 shows the slow-downs from attempting a basic qds Sturm count, finding an exception, and recounting with a careful loop. The x87 unit on the Xeon processors shows an excessive slow-down, detailed in Figure 6. This is evidence that the Xeon implements IEEE-754 special values in microcode for its x87 unit. In other words, every time the Xeon's x87 unit encounters an ∞ or NaN , it flushes its internal pipelines and jumps to a special handler program inside the processor. In contrast, the Athlon's x87 unit processes these special values at full speed.

Assuming exceptions occur rarely in practice, most slow-downs in Figure 5 and Table 5 are acceptable. The median is 1.63, less than the factor of 2 that we expected. Section 5 details how breaking the loops into shorter blocks can limit impact of the worst slow-downs.

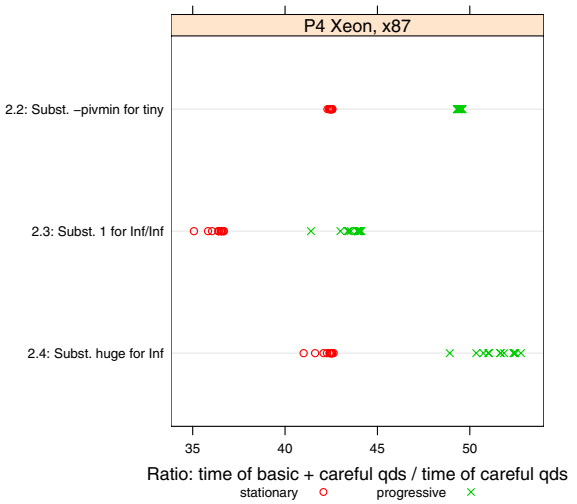


FIG. 6. *The* x87 *unit on Xeons suffers an extreme penalty for computing with NaNs, from over* 18× *to almost* 70× *slow-down.*

TABLE 5

Ratio of the failed basic loop plus the careful loop's times to the corresponding careful loops' times, or the slow-down of the attempted basic loop over the careful variants, for exceptional inputs across all platforms and both progressive and stationary qds.

Variant	Min	1st Quartile	Median	Mean	3rd Quartile	Max
All variants	1.088	1.526	1.634	6.417	1.963	52.77
2.2: pivmin	1.356	1.566	1.708	6.660	1.972	49.60
2.3: $\infty/\infty \Rightarrow 1$	1.343	1.544	1.712	5.956	1.917	44.13
2.4: $\pm\infty \Rightarrow \pm\Omega$	1.088	1.493	1.569	6.636	1.960	52.77

. To help choose one among the careful qds loops to support the basic loop, Figure 7 and Table 6 compare times of the basic loop followed by the two algorithms with range restrictions, Algorithms 2.2 and 2.4, to the time of the same loop followed by the algorithm without range restrictions, Algorithm 2.3. No algorithm completely dominates the performance of the others. We suggest that users adopt Algorithm 2.3 because it imposes no range restrictions on the data.

4. Impact on bisection. Does performance of the Sturm count impact eigensolvers? We answer this question in part by using section 2.3's algorithms in bisection. We apply bisection to the tridiagonal matrices from matrix collections and applications described in section 4.1. The bisection routine tested is a modification of the auxiliary routine DLARRB from LAPACK. We test each version of our Sturm counts individually by computing the eigenvalues to within a relative error of 4ε. Overall, the basic qds loop provides a median speed-up of only 1.08. Particular platforms such as the Cray X1 see significant improvements.

4.1. Test matrices. The symmetric tridiagonal matrices used in our experiments come from a wide range of real applications, as follows:

1. Applications in chemistry. The matrices are related to the modeling of biphenyl using the Moller–Plesset theory, the density functional theory applied to

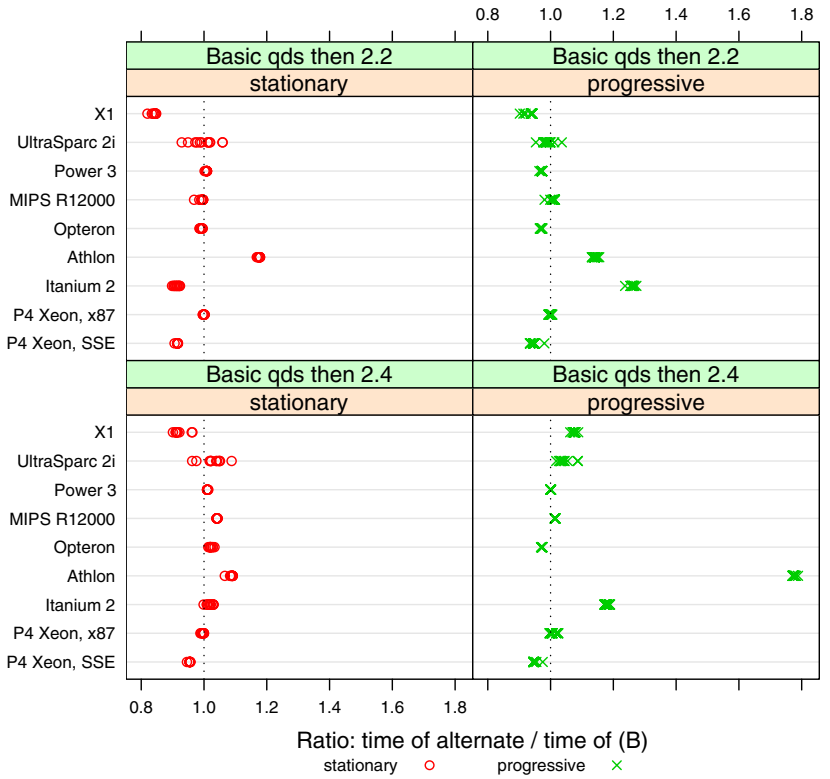


FIG. 7. *Comparing Algorithms* 2.2 *and* 2.4*, those with input range restrictions, to Algorithm* 2.3 *shows no all-around performance winner.*

TABLE 6

Ratio of the listed careful loops' times to the corresponding times for variant B ($\infty/\infty \Rightarrow 1$) *for exceptional inputs across all platforms and both progressive and stationary qds. To evaluate performance as a "backup" method, the times compared are those of the basic qds loop followed by the careful loop. We attribute the wide spread to wide architectural and compiler differences.*

Variant	Min	1st Quartile	Median	Mean	3rd Quartile	Max
2.2: pivmin	0.820	0.945	0.992	1.003	1.009	1.273
2.4: $\pm\infty \Rightarrow \pm\Omega$	0.901	0.995	1.014	1.061	1.048	1.788

determine the bulk properties of $SiOSi_6$, and the self-consistent Hartree–Fock method for solving a nonlinear Schroedinger problem. The matrices were obtained with the NWChem [13, 12] computational chemistry package and were provided by George Fann.

2. Matrices from the Harwell–Boeing collection [10, 11]. The matrices are related to the modeling of power system networks, a finite difference model for the shallow wave equations for the Atlantic and Indian Oceans, a finite element approximation to the biharmonic operator on a beam and on a plate, and structural engineering finite element analysis.

3. Matrices from the University of Florida sparse matrix collection [5]. The

TABLE 7
Matrices used in the tests were drawn from NWChem, the Harwell–Boeing collection (HB), and the University of Florida collection (UF).

Matrix	Application	Source	# of matrices	Dimensions
fann	Chemistry	NWChem	3	966–2053
bus	Power system networks	HB	4	494–1138
plat1919	Shallow wave equations	HB	1	1919
nos	Biharmonic operator	HB	2	957–960
bcsstk	Structural engineering	HB	18	1074–8012
alemdar, nasa, sts	Finite element problems	UF	17	1824–6438

TABLE 8
Ratio of bisection time using careful qds loops to corresponding bisection time with basic loops, or the bisection speed-up of the basic loop over the careful variants, for unexceptional inputs across all platforms and both progressive and stationary qds.

Variant	Min	1st Quartile	Median	Mean	3rd Quartile	Max
All variants	0.536	1.029	1.084	1.158	1.322	2.006
2.2: pivmin	0.734	1.024	1.038	1.112	1.350	1.543
2.3: $\infty/\infty \Rightarrow 1$	0.557	1.036	1.077	1.140	1.150	1.984
2.4: $\pm\infty \Rightarrow \pm\Omega$	0.536	1.034	1.119	1.223	1.432	2.006

matrices are related to finite element problems.

The matrices from 2 and 3 were tridiagonalized by means of LAPACK's DSYTRD or a simple MATLAB implementation of the Lanczos algorithm, with a starting vector filled with ones and no reorthogonalization, so as to provoke the appearance of very close eigenvalues in the resulting tridiagonal. In this case, Lanczos was run for $N \cdot k$ steps, where N is the dimension of the problem and k an integer at most 4. Table 7 gives more information about the matrices used in the tests.

4.2. Bisection performance results. Figures 8 and 9 show the CPU cycles per entry for each qds algorithm and platform. Because bisection does not run a fixed number of Sturm counts, calculating the cycles per entry requires replacing the 50000 in (3.1) with the number of calls to the qds routine in each individual execution. Section 4.1's test matrices generated no exceptions. Figure 10 provides the speed-ups for each algorithm and platform, and Table 8 summarizes those speed-ups across all platforms.

The median improvement of the basic qds loop over the careful algorithms is a modest 1.08, with a few notable slow-downs using the P4 Xeon's x87 unit. However, some platforms see substantial speed-ups. The Itanium 2 and Cray X1 both see speed-ups of at least 1.2. The Cray X1 benefits the most, achieving a maximum speed-up of 1.98 and a median speed-up of 1.66.

5. Blocking. For larger matrix sizes, the potential slow-downs shown in section 3.3 become unacceptable. Blocking the loops limits the slow-down for most cases without surrendering the basic loop's performance.

The idea is simple: Instead of performing the whole computation with the fast algorithm for the full loop, the computation is split into blocks $1 : n_1$, $n_1 + 1 : n_2$, and so forth. After the computation for the current block has finished, its correctness is assessed immediately. Thus, in the case of an invalid result, only the computation of the current block is repeated.

In our context of computing the Sturm count from a differential qds factorization,

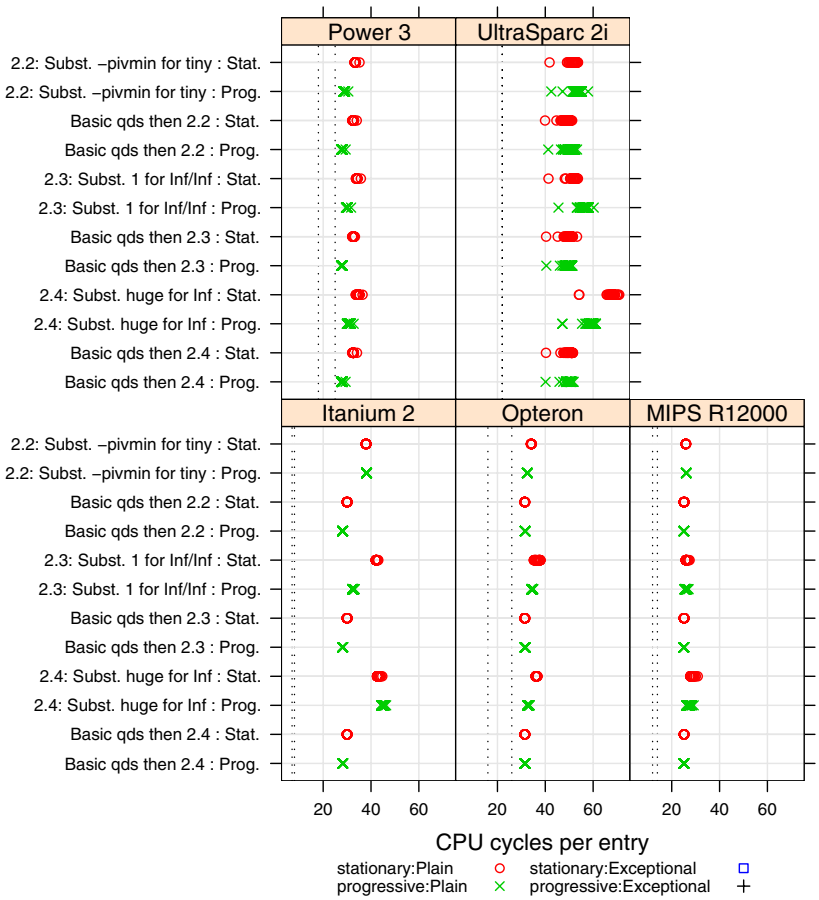


FIG. 8. *CPU cycles per entry for each qds algorithm during bisection. Vertical dashed lines show the range of documented division latencies, if known.*

blocking results in an outer loop over the blocks, and an inner loop that computes the Sturm count , the number of negative pivots in the factorization of the current block. The increment is added to the count if the factorization encounters no exceptions. Otherwise, the increment is recomputed with a careful algorithm.

The block size is a tuning parameter. Its setting has to address the trade-off between the cost for assessing the correctness of the intermediate solution, in our case the NaN test, and the potential overhead for redoing the computation. The optimal block size may depend not only on the computation costs but also on caching effects.

The benchmark test from section 3 motivates the blocking approach. The basic stationary differential qds algorithm produces an NaN in step 2 when the shift is chosen to be $d(1)$. Once that exception is handled, no further exceptions occur. Thus, the earlier the NaN is detected, the fewer unnecessary operations are performed. At the end of the first block, the NaN is detected, and only the first block's contribution is recomputed. The remainder of the blocks will proceed at full speed.

Blocking has no benefits in those very rare cases in which the computation has to be redone for block. One such example is an LDL^T factorization of a symmetric

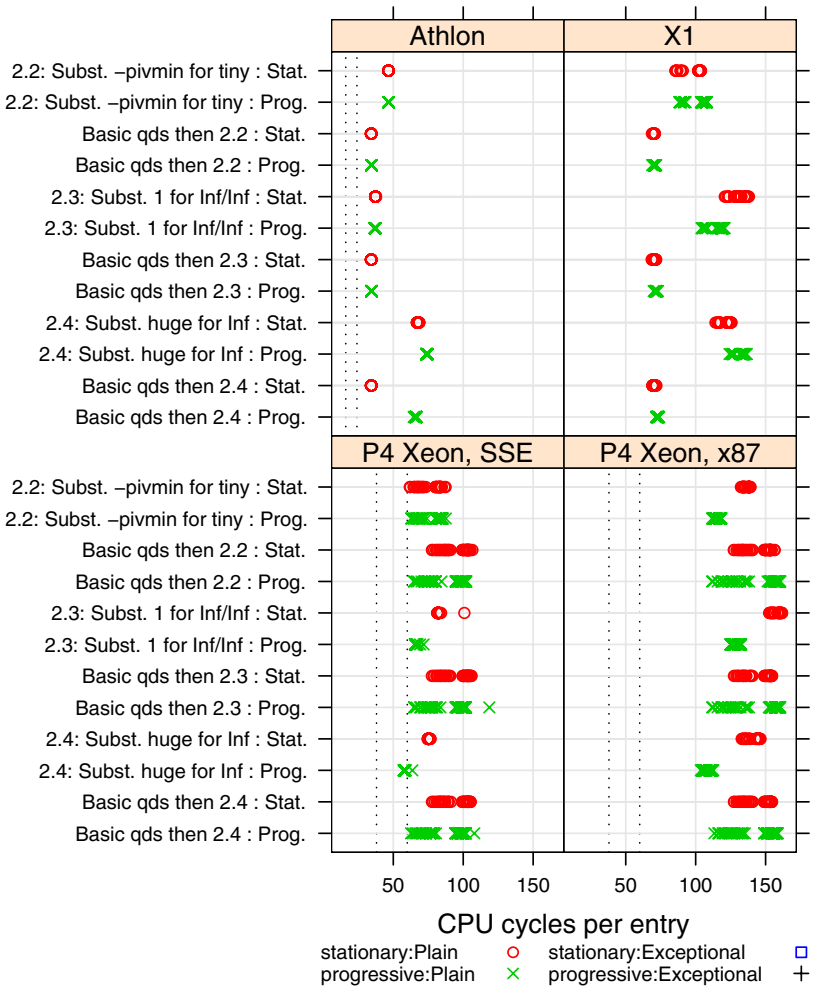


FIG. 9. *CPU cycles per entry for each qds algorithm during bisection for platforms needing a wider scale than Figure* 8. *Xeon values with the* x87 *unit are clipped to fit. Vertical dashed lines show the range of documented division latencies, if known.*

diagonally dominant tridiagonal Toeplitz matrix (e.g., the 1-2-1 matrix). We leave it to the reader to verify that for any shift in the stationary dqds algorithm and for each block size greater than two, the first pivot d_+ in each block is zero. Thus an NaN occurs in every block.

6. Conclusion. Evaluating the Sturm count with a qds factorization risks failure when a tiny or zero pivot causes an overflow, but avoiding those rare failures exacts a cost in performance. Following the example of Demmel and Li [8], we achieve better performance by attempting a fast but potentially erroneous qds factorization, testing for failure afterwards, and recomputing the factorization more carefully only when necessary. As shown in section 3.3, that strategy typically yields benchmark speed-ups over careful factorizations of around 1.12 across many platforms. However,

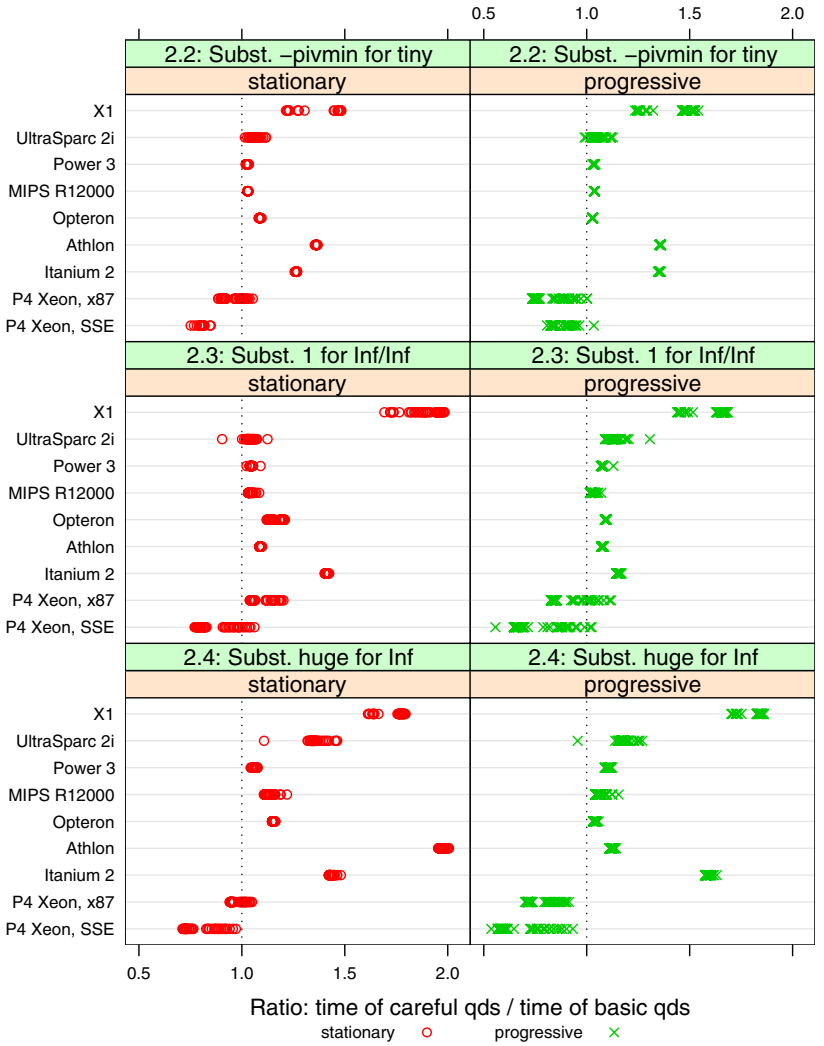


FIG. 10. *Similar improvements occur for bisection. A point at* 1.2 *means that bisection running time using the careful loop is* 1.2 *times as long as bisection time using the basic loop.*

a remarkable slow-down of up to 66× can occur for the x87 unit on the Pentium 4 Xeon in exceptional cases. For bisection on the matrices in section 4.1, we find the speed-ups vary greatly by platform and algorithm. While the median is a modest 1.08, some platforms benefit greatly. For example, bisection on the Cray X1 improves by factors of 1.2 to 2.

This paper has focused on the performance benefits of IEEE-754 features in the context of bisection. However, there are other interesting applications. We mention two related examples which should find similar benefits.

First, consider the dqds algorithm for computing the singular values of a bidiagonal matrix [14, 15, 22, 27]. There an invalid operation signals that stability has been lost. The shift to accelerate the convergence has been chosen too aggressively and

needs to be replaced by a more prudent shift.

Second, consider the MRRR algorithm's method of computing eigenvectors [23, 24, 25, 26, 9]. An invalid operation during the twisted factorization indicates that an eigenvector entry is tiny or zero. In this case, the fast product formula used to compute the eigenvector yields an incorrect result and must be replaced by a more careful formula.

Appendix A. Important implementation details. This section briefly describes important implementation details for our benchmark of the qds variants.

We describe the effort needed to test for NaNs to ensure a valid Sturm count from the unsafe loops in Algorithm 2.1. Only recently have language systems provided portable tests for NaNs. Since we cannot yet rely on those systems' availability, we provide our own test described in section A.1.

Also, Algorithms 2.3 and 2.4 must test for infinite values. Thankfully such tests can be implemented portably; see section A.2

Section A.3 discusses possible tests for the sign of a pivot.

And finally, section A.4 discusses the potential benefits and unfortunate drawbacks of using system trap handlers to treat exceptions when they occur.

A.1. Testing for NaNs. While NaN is the only IEEE-754 value for which X.EQ.X is false, many compilers in higher optimization levels replace this logical expression by a true value. In such circumstances, we have no in-line, reliable test for the presence of an NaN at run time.

For this reason, our experiments use a function of the form

```
LOGICAL FUNCTION DISNAN( DIN1, DIN2 )
DOUBLE PRECISION DIN1, DIN2
DISNAN = (DIN1.NE.DIN2)
END FUNCTION
```

that is compiled separately. Baring interprocedural analysis, the compiler must assume DIN1 and DIN2 are not related. DISNAN(X,X) thus returns true if and only if X is NaN.

Future languages will provide better support for IEEE-754 facilities. For example, the Fortran 2003 standard provides an NaN test with the IEEE_IS_NAN() function in the IEEE_ARITHMETIC module.

There is a trade-off in to test for NaNs. We could detect an NaN early by investigating the pivot in every step of the Sturm count, or we could use NaN propagation and wait and test only the final pivot. Including a test in every iteration penalizes unexceptional instances, while testing only after the loop requires loading all the data twice. Between these two extremes lies a range of blocking approaches; see section 5.

A.2. Testing for ∞. The substitutions in Algorithms 2.3 and 2.4 require a test for the presence of ∞. This can be done portably by the following code:

```
OMEGA = DLAMCH('Overflow')
ABS(X).GT.OMEGA
```

Here OMEGA (also Ω in the text) represents the overflow threshold. LAPACK provides Ω through _LAMCH('Overflow'). In Fortran 90, the enquiry function HUGE(X) returns the overflow threshold for the precision of its argument X. And Fortran 2003 provides IEEE_IS_FINITE() in the IEEE_ARITHMETIC module to test for nonfinite IEEE-754 values directly.

A.3. Testing the sign of a pivot. The straightforward way to count negative pivots in Fortran is to compare the pivot to zero, as in the following:

```
IF( PIVOT.LT.ZERO ) NEGCOUNT = NEGCOUNT + 1
```

The Cray compiler recognizes the sign test inside the inner loop and performs an interesting optimization to partially vectorize it. The pivots from a portion of the loop are copied into the intermediate vector. Then the entire vector is tested and converted into mask bits, and those mask bits are counted to provide a NEGCOUNT increment.

Another possible implementation copies the sign bit of PIVOT to an integer as 0 or 1, then adds that integer to NEGCOUNT. We lack a clear, direct way of expressing that implementation in Fortran.

A.4. Handling exceptional cases through traps. Algorithms 2.3 and 2.4 both explicitly test for exceptional operations. However, many platforms support implicit tests through ... A trap in this context is an implicit transfer of control from the exceptional operation to a ... The handler examines the exceptional operation and its operands and then returns a new result for the exceptional operation.

Trapping allows algorithms to run without explicit branches to test for exceptional cases; the tests are performed in hardware during floating-point operations. However, traps are only an optional feature in IEEE-754 [4]. While being widely implemented, each operating system and processor combination uses a different programming interface. This makes trap-handling nonportable. Furthermore, invoking a floating-point trap usually incurs expensive bookkeeping in both the processor and the operating system.

We experimented with trap handlers on three platforms with reasonable but very different interfaces: Solaris on an UltraSparc 2i, AIX on a Power3, and Linux on a Pentium 3. A single exceptional operation on any of these platforms increased the running time by a factor of at least 1000.

Acknowledgments. We would like to thank Prof. Kahan for Algorithm 2.4 and many comments. The anonymous reviewers provided very valuable advice and perspectives.

REFERENCES

[1] Advanced Micro Devices, Inc., *AMD Athlon Processor* x86 *Code Optimization Guide*, February 2002, 22007 revision K.

[2] Advanced Micro Devices, Inc., *Software Optimization Guide for AMD Athlon* 64 *and AMD Opteron*, November 2004, 25112 revision 3.05.

[3] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen, *LAPACK Users' Guide*, 3rd ed., SIAM, Philadelphia, 1999.

[4] ANSI/IEEE, *IEEE Standard for Binary Floating Point Arithmetic*, 754–1985 ed., New York, 1985.

[5] T. Davis, *University of Florida Sparse Matrix Collection*, http://www.cise.ufl.edu/research/sparse/matrices/, NA Digest, 92 (1994) (Oct. 16), 96 (1996) (July 23), and 97 (1997) (June 7).

[6] J. W. Demmel, I. S. Dhillon, and H. Ren, *On the correctness of some bisection-like parallel eigenvalue algorithms in floating point arithmetic*, Electron. Trans. Numer. Anal., 3 (1995), pp. 116–149.

[7] J. Demmel and W. Kahan, *Accurate singular values of bidiagonal matrices*, SIAM J. Sci. Statist. Comput., 11 (1990), pp. 873–912.

[8] J. W. Demmel and X. S. Li, *Faster numerical algorithms via exception handling*, IEEE Trans. Comput., 43 (1994), pp. 983–992.

[9] I. S. Dhillon, B. N. Parlett, and C. Vömel, *LAPACK Working Note* 162: *The Design and Implementation of the MRRR Algorithm*, Tech. report UCBCSD-04-1346, University of California, Berkeley, CA, 2004.
[10] I. S. Duff, R. G. Grimes, and J. G. Lewis, *Users' Guide for the Harwell-Boeing Sparse Matrix Collection (Release* I*)*, Tech. report RAL-TR-92-086, Atlas Centre, Rutherford Appleton Laboratory, Oxfordshire, UK, 1992.
[11] I. S. Duff, R. G. Grimes, and J. G. Lewis, *The Rutherford-Boeing Sparse Matrix Collection*, Tech. report RAL-TR-97-031, Atlas Centre, Rutherford Appleton Laboratory, Oxfordshire, UK, 1997.
[12] E. Apra et al., *NWChem, a Computational Chemistry Package for Parallel Computers*, version 4.7, Tech. report, Pacific Northwest National Laboratory, Richland, WA, 2005.
[13] R. A. Kendall et al., *High performance computational chemistry: An overview of NWChem a distributed parallel application*, Comput. Phys. Comm., 128 (2000), pp. 260–283.
[14] K. Fernando and B. N. Parlett, *Accurate singular values and differential qd algorithms*, Numer. Math., 67 (1994), pp. 191–229.
[15] K. V. Fernando and B. N. Parlett, *Implicit Cholesky algorithms for singular values and vectors of triangular matrices*, Numer. Linear Algebra Appl., 2 (1995), pp. 507–531.
[16] G. H. Golub and C. van Loan, *Matrix Computations*, 3rd ed., The Johns Hopkins University Press, Baltimore, MD, 1996.
[17] Intel Corporation, *Intel Itanium* 2 *Processor Reference Manual for Software Development and Optimization*, 251110–003, 2004.
[18] Intel Corporation, *IA*-32 *Intel Architecture Optimization Reference Manual*, 248966–012, 2005.
[19] W. Kahan, *Accurate Eigenvalues of a Symmetric Tridiagonal Matrix*, Technical report CS41, Computer Science Department, Stanford University, Stanford, CA, 1966 (revised 1968).
[20] W. Kahan, *A Demonstration of Presubstitution for* ∞/∞, unpublished, 2005.
[21] M. Overton, *Numerical Computing with IEEE Floating Point Arithmetic*, SIAM, Philadelphia, 2001.
[22] B. N. Parlett, *The new qd algorithms*, in Acta Numerica, 1995, Acta Numer., Cambridge University Press, Cambridge, UK, 1995, pp. 459–491.
[23] B. N. Parlett and I. S. Dhillon, *Fernando's solution to Wilkinson's problem: An application of double factorization*, Linear Algebra Appl., 267 (1997), pp. 247–279.
[24] B. N. Parlett and I. S. Dhillon, *Relatively robust representations of symmetric tridiagonals*, Linear Algebra Appl., 309 (2000), pp. 121–151.
[25] B. N. Parlett and I. S. Dhillon, *Multiple representations to compute orthogonal eigenvectors of symmetric tridiagonal matrices*, Linear Algebra Appl., 387 (2004), pp. 1–28.
[26] B. N. Parlett and I. S. Dhillon, *Orthogonal eigenvectors and relative gaps*, SIAM J. Matrix Anal. Appl., 25 (2004), pp. 858–899.
[27] B. N. Parlett and O. Marques, *An implementation of the dqds algorithm (positive case)*, Linear Algebra Appl., 309 (2000), pp. 217–259.
[28] H. Rutishauser, *Der Quotienten-Differenzen-Algorithmus*, Z. Angew. Math. Phys., 5 (1954), pp. 233–251.
[29] H. Rutishauser, *Vorlesungen über Numerische Mathematik*, Birkhäuser Verlag, Basel, 1976.
[30] Silicon Graphics, Inc., *Origin* 2000 *and Onyx*2 *Performance Tuning and Optimization Guide*, Document 007-3430-003, Mountain View, CA, 2002.
[31] Sun Microsystems, Inc., *UltraSPARC*-IIi *User's Manual*, 805-0087-01, 1997.
[32] S. Andersson, R. Bell, J. Hague, H. Holthoff, P. Mayes, J. Nakano, D. Shieh, and J. Tuccillo, *RS/*6000 *Scientific and Technical Computing: POWER*3 *Introduction and Tuning Guide*, 1st ed., IBM, Austin, TX, 1998.

SIAM J. SCI. COMPUT.
Vol. 28, No. 5, pp. 1634–1650
© 2006 Society for Industrial and Applied Mathematics

A CACHE-AWARE ALGORITHM FOR PDES ON HIERARCHICAL DATA STRUCTURES BASED ON SPACE-FILLING CURVES*

FRANK GÜNTHER†, MIRIAM MEHL‡, MARKUS PÖGL§, AND CHRISTOPH ZENGER‡

Abstract. Competitive numerical algorithms for solving partial differential equations have to work with the most efficient numerical methods like multigrid and adaptive grid refinement and thus with hierarchical data structures. Unfortunately, in most implementations, hierarchical data—typically stored in trees—cause a nonnegligible overhead in data access. To overcome this quandary—numerical efficiency versus efficient implementation—our algorithm uses space-filling curves to build up data structures which are processed linearly. In fact, the only kind of data structure used in our implementation is stacks. Thus, data access becomes very fast—even faster than the common access to nonhierarchical data stored in matrices—and, in particular, cache misses are reduced considerably. Furthermore, the implementation of multigrid cycles and/or higher order discretizations as well as the parallelization of the whole algorithm become very easy and straightforward on these data structures.

Key words. partial differential equations, space-tree, space-filling curves, cache-awareness

AMS subject classifications. 35J05, 25K99, 35R99, 65Y20, 68P05, 68V20

DOI. 10.1137/040604078

1. Introduction.

Most of the computing time in the numerical solution of partial differential equations (PDEs) is usually spent for the multiplication of a sparse matrix, representing the discrete operator, with a vector, representing the approximate solution. The amount of computing time and also the amount of memory heavily depend on the data structure used to represent the matrix and the solution vector. Many clever approaches are known, but it is usually impossible to access the memory in the course of the computation in such a way that the addresses do not "jump." As modern computer architectures use one or more cache-levels for the access of memory, jumps in the address space may cause cache misses, leading to a sometimes dramatic slow down of the computation. This fact is actually one of the most serious bottlenecks in high performance computing.

It seems to be very difficult to avoid these jumps in general. Therefore, we restrict our attention to a more special situation which is nevertheless general enough to be useful in many applications. The grid in our context is restricted to grids associated with space-trees [2], but we allow local (adaptive) refinement because adaptivity is crucial for the efficient solution of many problems. Moreover, space-trees allow a simple implementation of modern multilevel methods (additive or multiplicative) using appropriate hierarchical bases or generating systems [14], as we shall see in what follows. Hierarchical bases or generating systems also allow the simple handling of hanging nodes appearing in cases of local refinement. Complicated geometries can also be described easily by space-trees if some care is taken with a reasonably accurate discretization near the boundary.

As a main result of this paper, we show that in this restricted context we can avoid random access in the memory almost completely by using only a fixed and

*Received by the editors February 11, 2004; accepted for publication (in revised form) February 2, 2006; published electronically October 6, 2006.
http://www.siam.org/journals/sisc/28-5/60407.html
†Audi AG, Postfach 100457, 85045 Ingolstadt, Germany (frank.guenther@audi.de).
‡Institut für Informatik, TU München, Boltzmannstraße 3, 85748 Garching, Germany (mehl@in.tum.de, zenger@in.tum.de).
§Rohde & Schwarz GmbH, Muehlderferstr. 15, 8171 Muenchen, Germany (poegl@in.tum.de).

small number of stacks independent of the number of nodes in the grid. Stacks can be considered as the most simple data structures used in computer science. The two basic operations allowed on stacks are push and pop, where push puts data on top of a pile and pop takes data from the top of a pile. It is immediately clear that subsequent accesses to memory can move only from one memory location to the previous or next location, and thus stacks can be implemented very efficiently on modern computer architectures.[1] As a consequence, the organization of our algorithms has to be done in such a way that the data needed in the sequence of operations are always on top of one of the stacks, avoiding transports from one stack to another to get access to elements deeper in the stack.

The present algorithm described in this paper is based on a standard finite element discretization, but the principle behind it can be adapted to other discretization schemes as well. The leaves of the space-tree describe the finite elements and, during the accumulation process for the operator matrix, which is not explicitly stored but instead directly applied to the approximate solution, these elements are visited in an order described by a space-filling curve, the so-called Peano curve.

Space-filling curves are a well-known device for designing efficient algorithms in computer graphics (see, e.g., [35, 37]). In the context of numerical simulations based on space-trees, space-filling curves are already an established tool for some key-based addressing of grid elements and/or nodes and, in particular, data parallel implementations (see, e.g., [28, 40, 41, 15, 16, 23, 24, 25, 32, 34, 18, 27, 26, 9]). The grid partitioning algorithm defined by space-filling curves can be shown to have linear complexity and to give quasi-optimal partitions with respect to cut-sizes [40]. In addition, the partitioning is well suited for multilevel grids [40, 41, 15, 16, 24, 32, 18, 27].

It is also known that—due to locality properties of the curves—reordering grid cells according to the numbering induced by a space-filling curve improves cache-efficiency (see, e.g., [1]). Similar benefits of reordering data along space-filling curves can also be observed for other applications such as matrix transposition [7] or matrix multiplication [8]. We go one step further and, in addition to the reordering of cells, construct stacks for which we do not need any addressing and/or hashing, as we always know that all data needed lie on top of one of the stacks and can thus be accessed in an even more cache-efficient way.

In contrast to other approaches to cache-optimizations for PDE solvers that work with some hardware-oriented strategies like specialized data padding [20], the efficiency of our algorithm does not depend on the particular setting of cache parameters like cache-line length, associativity, etc. In the literature, such algorithms which are cache-aware by concept without detailed knowledge of the cache parameters are also called cache-oblivious [6, 12, 31]. The general need for cache-optimization on the software side can easily be deduced from the impossibility of the implementation of an optimal replacement strategy for the cache-lines on the hardware side [36].

Section 2 gives some basic background information on the operator evaluation during a run along a space-filling curve; in section 3, the essence of our algorithm, the construction of stacks with the help of the Peano curve, is described. Section 4 describes the realization of some numerical algorithms like multigrid and evaluation of the operator matrix on adaptively refined grids. Finally, in section 5, we give some results—processing times, number of cache misses, etc.—for some test examples. Section 6 briefly describes the extension to the three-dimensional case including numerical results.

[1] Conflicts in the cache between different stacks are minimal due to the small number of stacks, as we could show by the extremely high cache hit-rates achieved (see section 5).

2. Operator accumulation along space-filling curves. In the mathematical definition, a space-filling curve is a surjective continuous mapping of the unit interval $[0;1]$ to a compact d-dimensional domain Ω with positive measure. In our context, Ω is always the unit square, or the unit cube in three dimensions. As we look at multilevel adaptive rectangular grids, we restrict ourselves to recursively defined, self-similar space-filling curves with rectangular recursive decomposition of the domain. These curves are given by a simple generating template and a recursive refinement procedure which describes the (rotated or mirrored) application of the generating template in subcells of the domain to be covered [33]. Prominent representatives of this class of space-filling curves are the Hilbert curve and the Peano curve. See Figure 2.1 for some iterates of the two-dimensional curves.

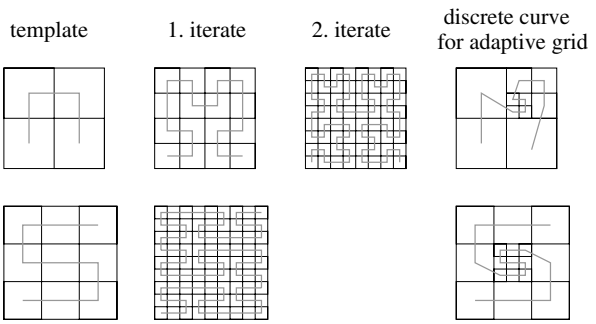


FIG. 2.1. *Generating templates, first iterates, and discrete curve on an adaptively refined grid for two-dimensional Hilbert and Peano curves.*

In fact, as our grids have finite resolution, the iterates—so-called discrete space-filling curves—are what we need (instead of the continuous space-filling curves). If we work with adaptively refined grids, the iterate that we use in a particular part of our domain depends on the local resolution (see also Figure 2.1 for some examples).

In our algorithm, this discrete space-filling curve defines the processing order of grid cells, corresponding to the leaves of our space-tree in a single-level context or to all nodes of the space-tree in a multilevel context. The application of the operator-matrix to the vector of data is done in a strictly cell-oriented way. For this, we decompose the discrete operator into parts per cell, which accumulate to the result of the operator evaluation after one run over all grid cells. This method is standard for finite element methods (see, e.g., [5]) but can be generalized to "local" discrete operators, which means that for the evaluation in one grid point, only direct neighbors are needed. In some cases—if the operator can be composed of a small number of local operators—this restriction can even be weakened. To illustrate the cell-oriented operator decomposition, we look at the one-dimensional three-point stencil

$$\begin{bmatrix} 1 & -2 & 1 \end{bmatrix}.$$

This simple stencil can be decomposed into a cell-part

$$\begin{bmatrix} 1 & -1 \end{bmatrix}$$

for the grid cell at the left-hand side of the respective point and

$$\begin{bmatrix} -1 & 1 \end{bmatrix}$$

for the grid cell at the right-hand side.

Thus, if we look at a space-tree with function values located at the vertices of cells, we have to construct our stacks such that all data from the respective cell-vertices lie on top of the stacks when we enter the cell during our run along the discrete space-filling curve.

3. Construction of stacks.

3.1. Regular grids with nodal data. To show the interaction of discrete space-filling curves and data stacks, we will start with a simple regular two-dimensional grid and nodal data. Figure 3.1 shows a grid where the processing order of grid cells is given by a discrete Peano curve.

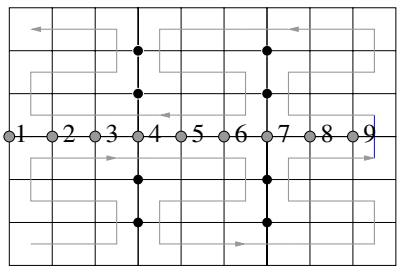


FIG. 3.1. *Example in two dimensions using the Peano curve.*

Let us have a closer look at data points on the middle line marked by 1 to 9. In the lower part of the domain, these data are processed linearly from 1 to 9; in the upper part, vice versa from 9 to 1. For the Peano curve and a regular grid, this linear forward and backward processing of the middle line can be shown for arbitrarily fine grids as well. Analogously, all other grid points can be organized on lines which are processed linearly forward and backward if we use the Peano curve to define the processing order of cells. Therefore, we organize all data within the well-known concept of stacks where we only have two possibilities of data access:

- push(data,number): write data on top of stack number,
- pop(number): read uppermost element of stack number.

Here, data is a structure in which all physical information of a grid point is held—for example the velocity vector and the type of boundary conditions.

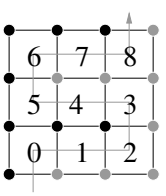


FIG. 3.2. *Stacks resulting from the usage of the Peano curve.*

A closer look at the two-dimensional example in Figure 3.2 shows that it is sufficient to have two stacks: $1D_{black}$ for the points on the right-hand side of the curve and $1D_{grey}$ for the points on the left-hand side of the curve. These stacks represent the evolving lines of points along the curve as described above. The behavior of a point concerning read and write operations on stacks can be predicted in a locally

deterministic way. We only have to classify the points as "inner points" or "boundary points" and respect the progression of the curve.

If we have a closer look at the example for the Peano curve, we can already extract two important properties that our space-filling curves have to fulfil: First, the processing order of grid points on a line has to be inverted if on our way along the Peano curve we switch from the domain on the right-hand side of the line to the domain on the left-hand side. This is quite easy to fulfill in the two-dimensional case, but in the three-dimensional case the same has to hold for grid points on planes (parallel to the coordinate planes). Second, a refinement of the grid must not destroy the order of existing points, but insert new points only between existing points. We could not find any Hilbert curve in three dimensions fulfilling these two properties, and there are good reasons to assume that this is not possible at all. Thus, we restrict our attention to Peano curves in the following. For the Peano curve even generalizations to four or more dimensions are straightforward.

It is a disadvantage of the Peano curve that in the refinement process each side of a cube is partitioned into three equal parts, leading to ternary trees instead of the standard binary trees used for quad-trees or octrees. This implies that the number of grid points grows like 3^d instead of 2^d in d-dimensional space from level to level. This implies that for a three-dimensional problem the number of grid points is multiplied by a factor of 27 if we add another refinement level.

An efficient algorithm deduced from the concepts described above passes the grid cell-by-cell and pushes/pops data to/from stacks deterministically and automatically. It will be cache-aware by design (independent of the actual cache-parameters such as cache-line-length, cache-size, etc.) because using linear stacks is a good idea in conjunction with caches that always load a whole block of neighboring data (cache-line) at once and, in addition, with modern processors using prefetching techniques.

3.2. Iterations on data. In the previous section, we have talked only about the intermediate storage of data within one solver iteration, but we have not yet mentioned where data come from when they are needed for the first time during a solver iteration and where we put them when they are "ready" (that is, when all work for the current situation is done at the corresponding point). For this purpose, we have to enhance our stack system by some input and output stacks to store all data between iterations. For this, we use so-called $2D$- or plane-stacks (as they include all data of the whole two-dimensional computational domain). The input stack contains all vertex data in the order of their first usage during one iteration. In fact, it acts as an input stream rather than a stack. Analogously, to store the output of the iteration, we again write all points to a $2D$-stack—the output stack or output stream—as soon as they are ready. It can easily be seen that, if we process the grid cells in the opposite direction in the next iteration, we can directly use the output stack as an input stream again. We apply this repeatedly and thus change the processing direction of grid cells after each iteration.

3.3. Extension to adaptive grids and hierarchical data. Up to this point, we have restricted our considerations to regular grids to explain the general idea of our algorithm, but the full potential of our approach becomes obvious only when we look at adaptively refined grids and, in the general case, hierarchical data in connection with generating systems [14]. This leads to more than one degree of freedom per grid point and function on coarse grid levels.

Before we can define our stacks for this case, we need an algorithm which now recursively visits the cells of different grid levels in a top-down depth-first process.

Since the discrete space-filling curve is defined by a recursion itself, this order can be deduced directly from the space-filling curve of the respective level. Figure 3.3 shows an example for such an order. The first cell visited is the coarsest cell, containing the whole domain. Next, the cells of the first refinement level are processed according to the first iterate of the Peano curve. As soon as we reach a cell which is further refined (in this example the middle cell), we process all "sons" of this cell according to the respective part of the next iterate of the Peano curve before we return to the next cell of the first refinement level.

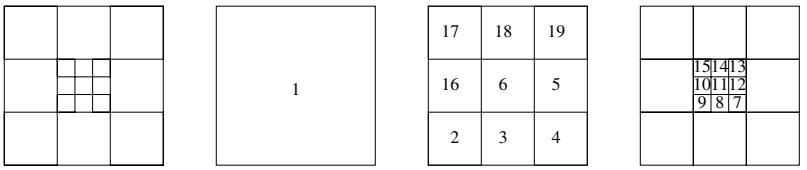


FIG. 3.3. *Example for the order of hierarchical cells defined by the discrete Peano curve.*

In our stack context, we have to assure even then that predictable and linear data access to and from stacks is possible. As points are visited on different grid levels now, we have to assure that grid points of coarser grid levels lie above those of finer levels in our stacks. Therefore, we have to change two basic properties of the concept described above: First, we have to realize restrictions and interpolations in addition to the pure evaluation of difference stencils, and thus we have to consider cells consisting of 3×3 finer cells and thus containing 16 grid points instead of single cells with only 4 grid points. Second, we have to introduce four "colors" instead of two (in Figure 3.4, the four colors are marked by crosses, circles, triangles, and squares) and a second type of stacks, so-called $0D$-stacks, in addition to the $1D$-stacks. The $0D$-stacks can be interpreted as an intermediate storage for vertex data to avoid the hiding of information associated with different levels, whereas the $1D$-stacks represent certain grid lines or parts of them, respectively. Again, we have a small and fixed number of stacks which is, in particular, independent of the problem size: four $1D$-stacks and four $0D$-stacks transporting hierarchical data over grid levels.

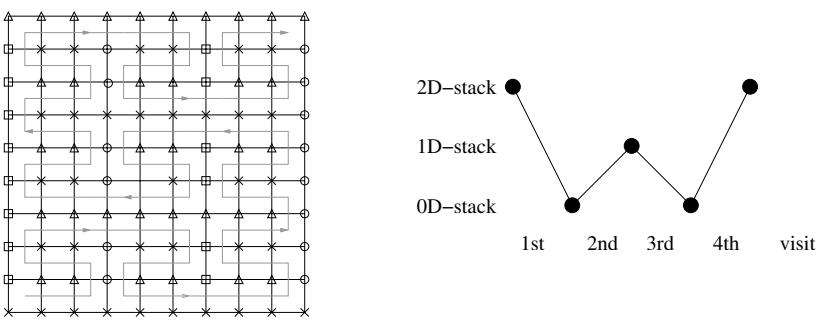


FIG. 3.4. *Left: recursive assignment of vertices to four "colors" and eight stacks; right: "way" of vertex data over the different stack types during one iteration of a two-dimensional solver visiting each vertex four times (corresponding to the four neighboring cells).*

For the formal construction of these stacks, we start with the one-dimensional case with nodal data, extend it to the one-dimensional case with hierarchical data (generating system), and finally generalize the concept to the two-dimensional case.

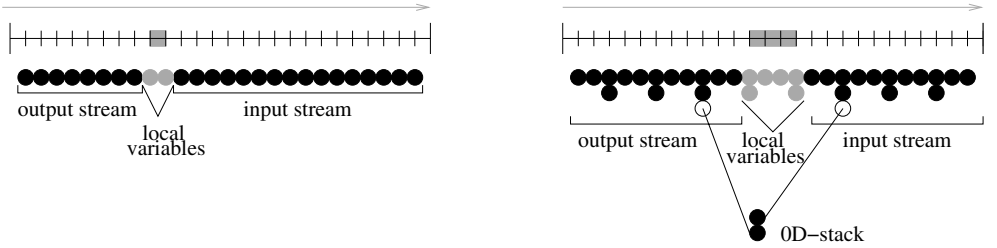


FIG. 3.5. *Illustration of the function and usage of stacks in the one-dimensional case (left: nodal data on a regular grid, right: hierarchical data (generating system)).*

The one-dimensional case with nodal data on a regular grid is trivial to handle: We process the grid cells from the left to the right (or vice versa) and, thus, do not need any stacks for the intermediate storage of points during one solver iteration. If we now introduce hierarchical data in the sense of a generating system, we cannot put every data point directly from the input- to the output-stream, since data points (degrees of freedom) associated with coarser grid levels are used once during the descend phase in the cell tree (to perform the interpolation to finer grids, e.g.) and a second time during the ascend phase (to restrict fine grid data to the respective points, e.g.). In between, we do (and finish) the work on all finer cells that are children (grandchildren, ...) of the respective coarse grid cell. Thus, to be able to access coarse grid points in the ascent phase again, we have to provide an intermediate data structure which we call the $0D$-stack (compare Figure 3.5). In fact, the two stack-operations `put` a datum on top of the stack and `pop` a datum from the top of the stack are sufficient, since at each point in time during a solver iteration, the $0D$-stack contains all vertex data of father (grandfather, ...) cells of the current cells in a top-down order and thus in just the correct order to find the currently needed data on top of the stack during the subsequent ascent of the cell tree.

For the extension to the two-dimensional case, we first reduce the two-dimensional data set to one-dimensional structure, and second reduce the handling of these structures to the one-dimensional case described above. As described in section 3.1, we can identify two lines of vertices processed in one direction during the first pass of the Peano curve and in the opposite direction during the second pass of the curve. If we now decompose those two lines into vertical and horizontal parts each (see Figure 3.4),[2] we end up with lines of vertices completely analogous to the one-dimensional lines considered above. The only difference is that they are not processed once but twice (once in each direction) during one solver iteration. Thus, one iteration of the two-dimensional solver corresponds to two iterations (with alternating processing directions) over the one-dimensional line. Altogether, we end up with eight stacks: one so-called $1D$-stack each for the vertical and the horizontal components of the two vertex groups from section 3.1, and one $0D$-stack for each of the $1D$-stacks. This results in the following "way" over the stacks for each data point associated with the grid vertices. At the beginning of a solver iteration, data are all stored in the $2D$-stack in the order of their first occurrence (like for the simple case of regular grids and nodal data). After their first usage, they are treated like vertices of a one-dimensional grid; that is, we move them to the $0D$-stack corresponding to the associated line (Figure 3.4) and, after the second usage, to the $1D$-stack, replacing the output stack

[2] [13] gives an example showing that the decomposition into vertical and horizontal parts is necessary to avoid a hiding of data between vertical and horizontal lines.

from the one-dimensional case. Subsequently, they are visited yet another two times in the opposite direction; that is, they are again first moved to the $0D$-stack and subsequently to the output stack, and then finally to the output stack (this time the $2D$ output stack; see Figure 3.4).[3]

With the ideas described above, we can now define numerical algorithms suited to adaptively refined grids, hierarchical data, and generating systems in the following section.

4. Numerical algorithms. As mentioned above, our data structures are very well suited for highly efficient numerical methods such as multigrid, adaptive grid refinement, and higher order discretizations. In this section, we will describe the concrete algorithmic realization of these methods.

We assume that we have to solve a system of linear equations

$$L_h u_h = f_h, \tag{4.1}$$

where h indicates the (locally defined) width of the square/cubic grid cells and the discrete operator L_h results from some "local" (see above) discretization of a differential operator.

4.1. Relaxation methods. Before we describe the implementation of a multigrid method, we will briefly mention the realization of relaxation methods with the help of our stacks and the cell-oriented evaluation of operators: The only thing special in comparison to standard implementations is that we do not compute the residual components $r_h^{(i)} = f_h^{(i)} - (L_h u_h)^{(i)}$ in one step per point, but accumulate them while visiting neighboring cells (as described above when we explained the cell-oriented evaluation of operators). For this, we need an appropriate decomposition of the right-hand side into cell-parts, which can easily be achieved, for example, by dividing f_h into four equal parts, which are assigned to the four neighboring cells. As soon as a grid point is visited the last time, we have computed the whole residual, and the respective value is updated according to the iteration scheme used.

4.2. Multigrid. As, in our stack, we store data of all refinement levels of the grid, the algorithm works on a generating system (in the context of finite elements) instead of a basis only. A multigrid cycle corresponds to a single run over all data. Thus, multigrid does not worsen the performance, in terms of runtime efficiency, of our program. In the following, we will briefly describe the basic ingredients of our additive multigrid algorithms, which turn out to perform well for adaptively refined grids in [4].

In contrast to relaxation methods, the multigrid method works on the whole set of hierarchical data, not only on the finest level. As described in the previous section, data are processed in a top-down, depth-first order (see also Figure 3.3 for an example of the processing order of cells). Thus, we cannot finish one step (smoothing, interpolation, or restriction) at all grid points of a level before we proceed to the next level. The cycle function is called recursively for all (nine in the case of two-dimensional Peano curves) subcells of a coarse cell, and thus, whenever we enter a cell, the whole work for one multigrid cycle within this cell has to be finished before we enter the next cell of the same level. Therefore, the natural choice in terms of our algorithm is an additive multigrid method where smoothing is done simultaneously

[3] Vertices visited directly consecutively are, of course, not moved to a stack in between but are kept as local variables.

on all levels.[4] Descending the space-tree[5] within the top-down depth-first run over the grid along the Peano curve, we interpolate coarse grid values to the finer grids in order to achieve nodal values on the finest grid. At the finest level we compute the residual and apply the smoother. Ascending the space-tree, we restrict the residual to the coarse levels and apply the smoother on the coarse levels.

1. We have to take care that residuals are correctly transported from fine to coarse grid points: Since we restrict the residual each time we visit a fine grid point (and only when the whole residual is computed), we may restrict only the cell-part of the respective residual and not the accumulated residual. This complication requires some additional local variables for the cell-parts of the residual.

2. As we use a generating system for u_h instead of nodal values, coarse grid corrections are simply done by smoothing the coarse-grid coefficients of the function representations. The transport to fine grid values is done implicitly during dehierarchization of u_h.

4.3. Adaptively refined grids. If we use the processing order of cells and the construction of stacks described in the previous section, adaptively refined grids can also be handled in a very natural way without any further complications, like, for example, special difference stencils at borders between differently refined areas.

Like for regular grids, we compute all operators in a cell-oriented way. Thus, the stencils used look the same for each cell (up to a potential scaling with the cell width h). Thus, in a first step, we end up with contributions to operators at all vertices of cells at the locally finest level. In particular, hanging nodes also have to carry function values and contributions to the operators. To eliminate hanging nodes from the system of equations (hanging nodes may not be considered as degrees of freedom!) we have to compute interpolated function values at hanging nodes before the operator evaluation and distribute the respective cell-part of the operators to the neighboring coarse grid points by some suitable restriction. With this restriction, we automatically get correct operator values at all coarse grid points. From the algorithmic point of view, this interpolation and restriction works completely analogously to the multigrid interpolation and restriction.

5. Examples. To point out the potential of our algorithm, we show some simple examples and the achieved results and make some remarks on the efficiency of our program concerning storage requirements, processing time, and cache behavior. Note that at this stage we are working with an experimental code which is not yet optimized at all. Thus, absolute values like computing time will be improved further, and our focus here is only on the cache behavior and the qualitative dependencies between the number of unknowns and the performance values like computing times.

5.1. Poisson equation on the unit square. As a first test, we solve the two-dimensional Poisson equation on the unit square with homogeneous Dirichlet boundary conditions:

$$\triangle u(\mathbf{x}) = -2\pi^2 \sin(\pi x)\sin(\pi y) \quad \forall\, \mathbf{x} = (x,y)^T \in \Omega =]0,1[\times]0,1[\,, \tag{5.1}$$

$$u(\mathbf{x}) = 0 \quad \forall\, \mathbf{x} \in \partial\Omega. \tag{5.2}$$

[4]The implementation of a multiplicative multigrid method is possible but requires additional techniques to outsource fine grid data currently not needed on coarser grid levels to separate data structures and to afterwards insert them into the input-stream again. The algorithmic methods needed are similar to those used for dynamical adaptivity [10] and currently in the implementation phase.

[5]In contrast to "descending" in the common context of multigrid methods, descending in our context means proceeding from the coarsest level to the finest.

The exact solution of this problem is given by $u(\mathbf{x}) = \sin(\pi x) \cdot \sin(\pi y)$. To discretize the Laplace operator, we use the common finite element stencil

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}.$$

The resulting system of linear equations is solved by an additive multigrid method with bilinear interpolation and full-weighting as restriction operator. As criteria for termination of the iteration loop we take a value of $r_{max} = 10^{-5}$, where r_{max} is the maximum (in magnitude) of all corrections over all levels. Table 5.1 shows performance values obtained on a dual Intel XEON 2.4 GHz with 4 GB of RAM for regular grids with growing resolution. All measurements of the cache-performance were realized with the help of the tools , or , which are based on [19] by Hewlett-Packard and which allow access to the hardware performance counters of the respective processors.

TABLE 5.1
Performance values for the two-dimensional Poisson equation on the unit square solved on a dual Intel XEON 2.4 GHz with 4 GB of RAM.

Resolution	Variables over all levels	Iterations	MFLOPS	CPU time (s)	L2 cache misses per iteration	#real misses / #minimal misses
243×243	65,708	39	354.55	5.800	134,432	1.09
729×729	595,692	39	360.20	52.130	1,246,949	1.12
2187×2187	5,374,288	39	359.65	472.410	11,278,058	1.12

Note that the number of iterations until convergence is independent of the resolution, which one would have expected for a multigrid method.

Table 5.1 shows very high L2 hit-rates of at least 99.13% measured on an Intel XEON, which is a very high value for "real" algorithms beyond simple programs for testing performance of a given architecture. Even more significant for the efficiency of our algorithm is the result of a comparison of the minimal number of necessary cache misses and actually measured cache misses: With the size s of a stack element, the number n of degrees of freedom, and the size cl of an L2 cache line on the architecture used, we can guess a minimum $cm_{min} = \frac{n \cdot s}{cl}$ of cache misses per iteration, which has to occur if we read each grid point once per iteration, producing $\frac{s}{cl}$ cache misses per point. In fact, grid points are typically used several times (in our case once by each of the four neighboring cells) in our algorithm, as well as in most FEM-algorithms. Thus, even this minimum guess is assumed to be too low. The entries of the last column in Table 5.1 are defined as $\frac{cm_{real}}{cm_{min}}$, where cm_{real} are the L2 cache misses per iteration simulated with `calltree`. As this rate is nearly 1 in our algorithm, we produce hardly more cache misses than if we used every grid point only once per iteration.

5.2. Poisson equation on a disk. The examples above show a very high performance on the L2 cache for full grids, but the major advantage of our method is that this holds also for adaptive grids and for more complicated geometries. To show this at least for a simple example, we consider the two-dimensional Poisson equation with inhomogeneous Dirichlet boundary conditions on a disk with radius one:

$$\triangle u(\mathbf{x}) = -2\pi^2 \sin(\pi x) \sin(\pi y) \quad \forall\, \mathbf{x} = (x, y)^T \text{ with } \|\mathbf{x}\| < 1, \tag{5.3}$$

$$u(\mathbf{x}) = \sin(\pi x) \cdot \sin(\pi y) \quad \forall\, \mathbf{x} \in \partial\Omega. \tag{5.4}$$

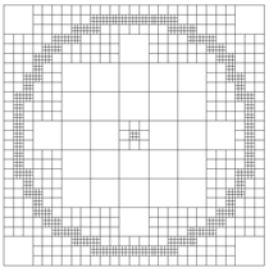


FIG. 5.1. *Representation of the disk with the help of an adaptive grid gained by a geometry-oriented coarsening of a* 81×81 *grid.*

TABLE 5.2
Performance values for the two-dimensional Poisson equation on a disk solved on a dual Intel XEON 2.4 GHz with 4 GB of RAM. "Iter" is the number of iterations.

Variables	% full grid	Computing time (s)	L2 hit-rate	Iter	time per iteration (ms)	time per variable and iteration (ms)
66220	15.094%	39.68	99.28%	287	138.26	0.002088
101146	23.055%	58.95	99.26%	287	205.40	0.002031
175578	40.021%	97.10	99.20%	286	339.51	0.001934
438719	100.00%	250.23	99.16%	281	890.50	0.002030

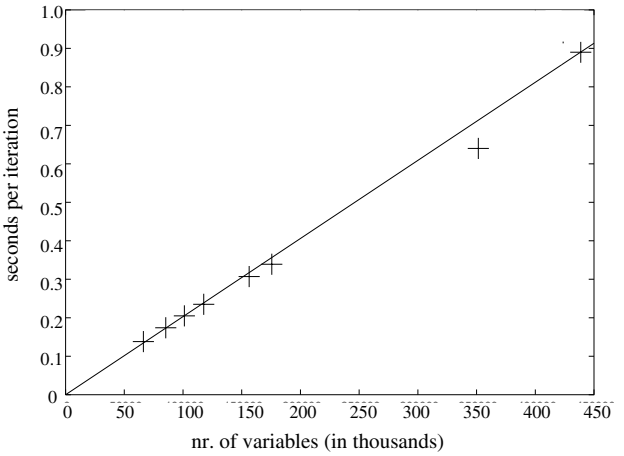


FIG. 5.2. *Linear dependency: number of variables versus computing time.*

Figure 5.1 shows the representation of the disk with the help of an adaptive grid gained by a geometry-oriented coarsening of an initial grid with 81×81 cells.

For our numerical tests, we used different adaptive grids gained by local coarsening strategies starting from an initial grid with 729×729 cells. To get sufficient accuracy near the boundary, we did not allow any coarsening of boundary cells.

The stopping criterion for the iterative solver was again $|r_{max}| < 10^{-5}$. In Table 5.2, the last column shows the amount of time needed per variable and per iteration. These values are nearly constant or even better for some adaptive grids than for the full grid. In Figure 5.2 you see the value "seconds per iteration" plotted against

the number of variables with crosses. The solid line is the corresponding line from the origin through the full grid point $(438719, 768.934)$. We see that we have a linear dependency between the number of variables/grid points and the computing time, no matter whether we have a regular full grid or an adaptively refined grid. This is a remarkable result, as we can conclude from this correlation that in our method variables on coarser grids do not have higher costs than those on the finest level.

6. Extension to three dimensions. In the previous sections we considered only the two-dimensional case. However, as mentioned in the introduction, our concepts can be generalized in a very natural way to three or even more dimensions. In this section, we want to give a few preliminary results on the three-dimensional case. The basic concepts are the same as in the two-dimensional case, but to achieve an efficient implementation, we introduce some changes and/or enhancements in the concrete realization.

The first, and obvious, change is that we need $3D$ in- and output stacks and $2D$-, $1D$-, and $0D$-stacks (corresponding to faces, edges, and vertices of a cube) during our run through the space tree instead of $1D$- and $0D$-stacks only. In addition, we use twelve $0D$- and $1D$-stacks each and six $2D$-stacks. These results can be deduced in a recursive way from the two-dimensional case (similarly as we deduced the two-dimensional case from the one-dimensional case): We decompose the set of vertices into "planes," distinguishing between the three possible orientations of a plane and the side of the curve (left, right) on which they are situated. Each of these six types of planes (corresponding to the six $2D$-stacks) can be handled like a two-dimensional set of vertices, again with the only difference that a solver iteration in three dimensions corresponds to two iterations over the respective planes. Since several planes can share the same $1D$- and $0D$-stacks, we need only twelve types of them instead of 24. The proof for the correctness of this concept can be found in [22]. Figure 6.1 shows the "way" of a data point over the stacks in the three-dimensional way.

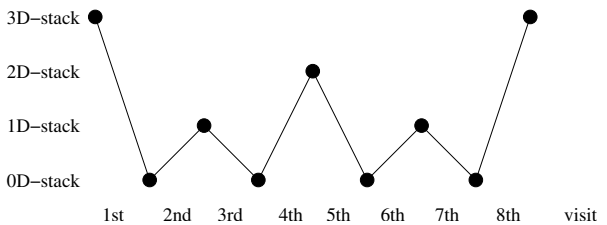


FIG. 6.1. *Way of data points over the different types of stacks in the three-dimensional case.*

Another interesting aspect in three dimensions is that we replace the direct refinement of a cell by the introduction of 27 subcells through a dimension recursive refinement: A cell is cut into three "plates" in a first step, each of these plates is cut into three "bars," and finally, each bar is cut into three "cubes" (see Figure 6.2). This reduces the number of different cases dramatically and can even be generalized to arbitrary dimensions [17]. Detailed descriptions of the three-dimensional implementations, including dynamical adaptivity, higher order discretizations, and full multigrid methods, are given in [30, 22, 10].

As can be seen from the following example, the performance of our algorithm carries over from two to three dimensions. The L2 hit-rates even get better by an order of magnitude (from 99% in the two-dimensional case to 99.99% in the three-

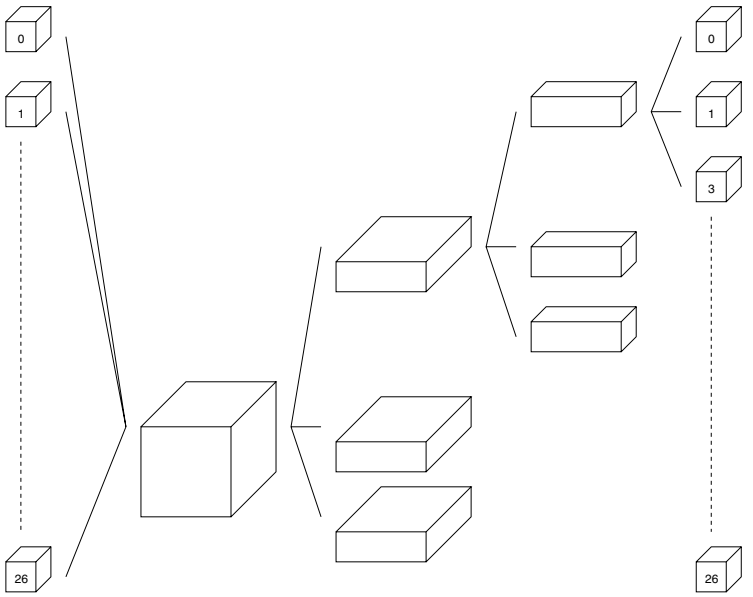


FIG. 6.2. *Dimension recursive refinement of a cell.*

TABLE 6.1
Performance values for the three-dimensional Poisson equation on a star-shaped domain solved on a dual Intel XEON 2.4 GHz with 4 GB of RAM. Abbreviations as in Table 5.2.

Resolution of the full grid	Variables	Storage requirements	Computing time (s)	L2 hit-rate	Iter	time per variable per iteration (ms)
$243 \times 243 \times 243$	508,528	9MB	2,405.86	99.99%	103	0.0459
$729 \times 729 \times 729$	13,775,328	230MB	63,502.14	99.98%	103	0.0448

dimensional case). Only the absolute computing time per variable and per iteration become worse. Both phenomena can easily be explained for the most part by the more complicated discretization stencil, with more nonzero entries which cause bigger computational cost, on the one hand, but also a larger number of operations performed per data point.

We solve the three-dimensional Poisson equation

$$\triangle u((\mathbf{x})) = 1 \tag{6.1}$$

on a star-shaped domain (see Figure 6.3), a sphere, and the unit cube with homogeneous boundary conditions. The Laplace operator is discretized by the common finite difference stencil, and—analogously to the two-dimensional case—we use an additive multigrid method with trilinear interpolation and full weighting as restriction operator. The termination criterion was $|r_{max}| \leq 10^{-5}$. Tables 6.1 and 6.2 show performance values obtained on a dual Intel XEON 2.4 GHz with 4 GB of RAM for adaptive grids (see also Figure 6.3 for a part of the (adaptively coarsened) $243 \times 243 \times 243$ grid for the star-shaped domain). For the star-shaped domain, the grid-coarsening was restricted to the portion of the domain outside the computational domain. Table 6.3 shows another important advantage of our algorithm—that we could handle a huge number of degrees of freedom with the same performance (e.g., over 400,000,000

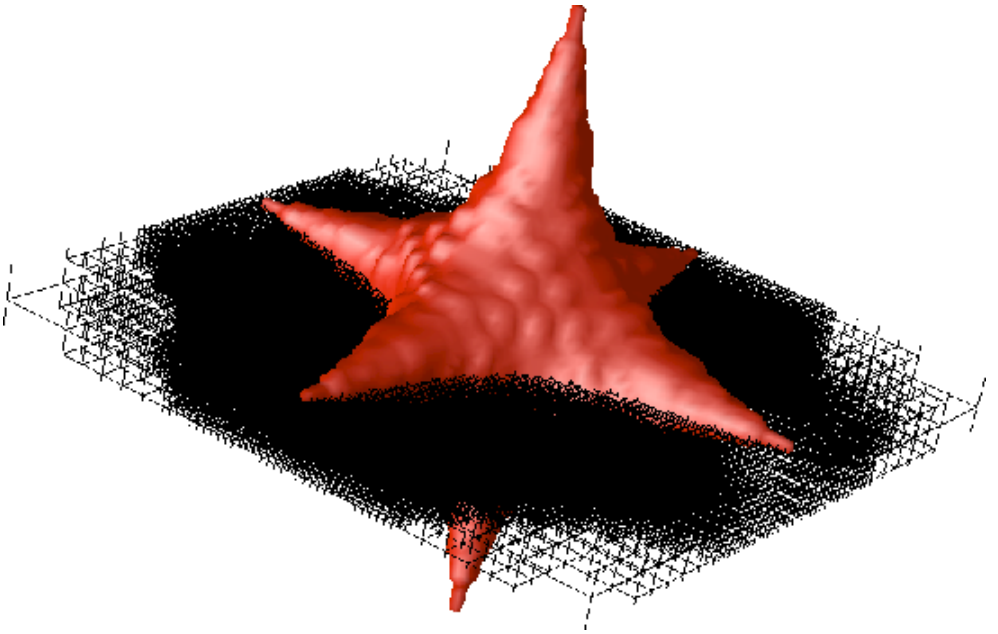


FIG. 6.3. *Representation of the star-shaped domain with the help of an adaptive grid gained by a geometry-oriented coarsening of a* $243 \times 243 \times 243$ *grid.*

TABLE 6.2
Performance values for the three-dimensional Poisson equation on a sphere solved on a dual Intel XEON 2.4 *GHz with* 4 *GB of RAM. Abbreviations as in Table* 5.2.

Resolution of the full grid	Variables	Storage requirements	Computing time (s)	L2 hit-rate	Iter	time per variable per iteration (ms)
$243 \times 243 \times 243$	1,969,872	24MB	4,949.87	99.98%	96	0.0262
$243 \times 243 \times 243$	900,024	6MB	773.21	99.97%	97	0.0089
$729 \times 729 \times 729$	24,231,440	150MB	20,127.97	99.94%	100	0.0083

TABLE 6.3
Performance values for the three-dimensional Poisson equation on the unit cube solved on a dual Intel XEON 2.4 *GHz with* 4 *GB of RAM. Abbreviations as in previous tables.*

Resolution of the full grid	Variables	Storage requirements	Computing time (s)	L2 hit-rate	Iter	time per variable per iteration (ms)
$243 \times 243 \times 243$	14,702,584	76MB	2,641.86	99.91%	42	0.0043
$729 \times 729 \times 729$	400,530,936	2GB	72,307.26	99.90%	42	0.0043

variables) and with a very small amount of memory. The high number of iterations needed for the star-shaped domain, in particular, can be ascribed first to the worse performance of additive multigrid schemes in comparison to multiplicative ones, especially as tuning the relaxation parameter becomes more and more tricky for complicated problems, and second to the relatively bad approximation of complicated geometries on the coarse grid levels. Both problems, the switch to multiplicative multigrid methods and the better approximation of complicated geometries on coarse grids, are subjects of our current work.

7. Conclusion. In this paper we have presented a method combining several features for the solution of PDEs:

- adaptivity,
- efficient multilevel algorithm,
- geometrical flexibility,
- suitability for modern computer architectures (cache-awareness and parallelizability combined with load balancing),
- suitability for general discretized systems of PDEs.

The present implementations include the first three of these aspects and part of the fourth aspect. However, the implementation of the other features is not difficult. The parallelization of the method follows step-by-step the approach of Zumbusch [40], who has already successfully used space-filling curves for the parallelization of PDE-solvers. The generalization to systems of PDEs is also straightforward. [38] implemented a version solving the two-dimensional Navier–Stokes equations, and [29] presents results for problems with nonconstant coefficients.

All codes used here are still in an experimental state, in particular without any runtime optimization. Optimization is one of the main tasks of our current work, where we concentrate on testing the applicability of different methods like exploiting locally regular grid parts for a better vectorization, or methods similar to those used in [11, 20, 21, 39] or with the help of streaming SIMD[6] extensions (SSE).

As a further remark we want to mention that space-filling curves can also be used, e.g., for the computation of the product of a full matrix with a full vector, or a full matrix with a full matrix, where we also obtain a reduction of cache misses [3].

REFERENCES

[1] M. J. Aftosmis, M. J. Berger, and G. Adomavivius, *A Parallel Multilevel Method for Adaptively Refined Cartesian Grids with Embedded Boundaries*, AIAA Paper, 38th AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, 2000.

[2] M. Bader, H.-J. Bungartz, A. Frank, and R. Mundani, *Space tree structures for pde solution*, in Proceedings of ICCS 2002, Part 3, P. M. A. Sloot, C. J. K. Tan, J. J. Dongarra, and A. G. Hoekstra, eds., Lecture Notes in Comput. Sci. 2331, Springer, New York, 2002, pp. 662–671.

[3] M. Bader and C. Zenger, *Cache oblivious matrix multiplication using an element ordering based on the Peano curve*, Linear Algebra Appl., (2006), to appear.

[4] F. A. Bornemann, *An adaptive multilevel approach to parabolic equations* III: 2*D error estimation and multilevel preconditioning*, Impact Comput. Sci. Engrg., 4 (1992), pp. 1–45.

[5] D. Braess, *Finite Elements. Theory, Fast Solvers and Applications in Solid Mechanics*, Cambridge University Press, Cambridge, UK, 2001.

[6] E. D. Demaine, *Cache-oblivious algorithms and data structures*, in Lecture Notes from the EEF Summer School on Massive Data Sets, Lecture Notes in Computer Science, BRICS, University of Aarhus, Denmark, 2002.

[7] S. Chatterjee and S. Sen, *Cache-efficient matrix transposition*, in Proceedings of HPCA-6, Toulouse, France, 2000, IEEE Press, Piscataway, NJ, pp. 195–205.

[8] S. Chatterjee, A. R. Lebeck, P. K. Patnala, and M. Thottethodi, *Recursive array layouts and fast parallel matrix multiplication*, in Proceedings of the Eleventh Annual ACM Symposium on Parallel Algorithms and Architectures, Saint-Malo, France, 1999, ACM, New York, pp. 222–231.

[9] W. Clarke, *Key-Based Parallel Adaptive Refinement for FEM*, bachelor thesis, Department of Engineering, Australian National University, Canberra, Australia, 1996.

[10] N. Dieminger, *Kriterien für die Selbstadaption cache-effizienter Mehrgitteralgorithmen*, diploma thesis, Institut für Informatik, TU München, Munich, Germany, 2005.

[6] SIMD denotes a single instruction multiple data architecture.

[11] C. C. Douglas, J. Hu, M. Kowarschik, U. Rüde, and C. Weiss, *Cache optimization for structured and unstructured grid multigrid*, Electron. Trans. Numer. Anal., 10 (2000), pp. 21–40.

[12] M. Frigo, C. E. Leierson, H. Prokop, and S. Ramchandran, *Cache-oblivious algorithms*, in Proceedings of the 40th Annual Symposium on Foundations of Computer Science, New York, 1999, IEEE Press, Piscataway, NJ, pp. 285–297.

[13] F. Günther, *Eine cache-optimale Implementierung der Finite-Elemente-Methode*, Doctoral thesis, TU München, Munich, Germany, 2004.

[14] M. Griebel, *Multilevelverfahren als Iterationsmethoden über Erzeugendensystemen*, Habilitationsschrift, TU München, Munich, Germany, 1993.

[15] M. Griebel and G. W. Zumbusch, *Parallel multigrid in an adaptive PDE solver based on hashing and space-filling curves*, Parallel Computing, 25 (1999), pp. 827–843.

[16] M. Griebel and G. Zumbusch, *Hash based adaptive parallel multilevel methods with space-filling curves*, in Proceedings of the NIC Symposium 2001, NIC Series 9, H. Rollnik and D. Wolf, eds., John von Neumann Institut für Computing, Forschungszentrum Jülich Germany, 2002, pp. 479–492.

[17] J. Hartmann, *Entwicklung eines cache-optimalen Finite-Element-Verfahrens zur Lösung d-dimensionaler Probleme*, diploma thesis, Institut für Informatik, TU München, Munich, Germany, 2005.

[18] M. Griebel and G. Zumbusch, *Hash-storage techniques for adaptive multilevel solvers and their parallelization*, in Domain Decomposition Methods 10 (Boulder, CO, 1997), Contemp. Math. 218, J. Mandel, C. Farha, and X.-C. Cai, eds., AMS, Providence, RI, 1998, pp. 271–278.

[19] HP invent, *perfmon: Create Powerful Performance Analysis Tools Which Use the IA-54 Performance Monitoring Unit (PMU)*, project website at http://www.hpl.hp.com/research/linux/perfmon/index.php4.

[20] M. Kowarschik, *Data Locality Optimizations for Iterative Numerical Algorithms and Cellular Automata on Hierarchical Memory Architectures*, Ph.D. thesis, Institut für Informatik, Universität Erlangen-Nürnberg, SCS Publishing House, Ghent, Belgium, 2004.

[21] M. Kowarschik and C. Weiss, *An overview of cache optimization techniques and cache-aware numerical algorithms*, in Algorithms for Memory Hierarchies—Advanced Lectures, Lecture Notes in Comput. Sci. 2625, U. Meyer, P. Sanders, and J. Sibeyn, eds., Springer, New York, Berlin, 2003, pp. 213–232.

[22] A. Krahnke, *Adaptive Verfahren höherer Ordnung auf cache-optimalen Datenstrukturen für dreidimensionale Probleme*, Doctoral thesis, Institut für Informatik, TU München, Munich, Germany, 2005.

[23] A. Laszloffy, J. Long, and A. K. Patra, *Simple data management, scheduling and solution strategies for managing the irregularities in parallel adaptive hp finite element simulations*, Parallel Computing, 26 (13–14) (2000), pp. 1765–1788.

[24] J. T. Oden, A. Para, and Y. Feng, *Domain decomposition for adaptive hp finite element methods*, in Domain Decomposition Methods in Scientific and Engineering Computing, Proceedings of the 7th International Conference on Domain Decomposition, Pennsylvania State University, Contemp. Math. 180, D. E. Keyes and J. Xu, eds., AMS, Providence, RI, 1994, pp. 203–214.

[25] A. K. Patra, J. Long, and A. Laszloffy, *Efficient parallel adaptive finite element methods using self-scheduling data and computations*, in HiPC: Proceedings of the 6th International Conference on High Performance Computing, Calcutta, India, P. Banerjee, V. K. Prasanna, and B. S. Sinha, eds., Lecture Notes in Comput. Sci. 1745, Springer, NY, 1999, pp. 359–363.

[26] A. K. Patra, A. Laszloffy, and J. Long, *Data structures and load balancing for parallel adaptive hp finite-element methods*, Comput. Math. Appl., 46 (2003), pp. 105–123.

[27] M. Parashar and J. C. Browne, *On partitioning dynamic adaptive grid hierarchies*, in Proceedings of the 29th Annual Hawaii International Conference on System Sciences, Maui, HI, 1996, pp. 604–613.

[28] M. Parashar, J. C. Browne, C. Edwards, and K. Klimkowski, *A common data management infrastructure for parallel adaptive algorithms for PDE solutions*, in Proceedings of the 1997 ACM/IEEE Conference on Supercomputing, ACM Press, New York, 1997, pp. 1–22.

[29] B. Pentenrieder, *Finite Element Solutions of Heat Conduction Problems in Complicated 3D Geometries Using the Multigrid Method*, Diploma thesis, Fakultät für Informatik, TU München, Munich, Germany, 2005.

[30] M. Pögl, *Entwicklung eines cache-optimalen 3D Finite-Element-Verfahrens für große Probleme*, Doctoral thesis, Institut für Informatik, TU München, Munich, Germany, 2004.

[31] H. Prokop, *Cache-Oblivious Algorithms*, Master Thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, 1999.
[32] S. Roberts, S. Klyanasundaram, M. Cardew-Hall, and W. Clarke, *A key based parallel adaptive refinement technique for finite element methods*, in Proceedings of the Conference on Computational Techniques and Applications: CTAC '97, B. J. Noye, M. D. Teubner, and A. W. Gill, eds., World Scientific, Singapore, 1998, pp. 577–584.
[33] H. Sagan, *Space-Filling Curves*, Springer-Verlag, New York, 1994.
[34] J. Steensland, S. Chandra, and M. Parashar, *An application-centric characterization of domain-based SFC partitioners for parallel SAMR*, IEEE Trans. Parallel Distributed Systems, 13 (2002), pp. 1275–1289.
[35] R. J. Stevens, A. F. Lehar, and F. H. Preston, *Manipulation and presentation of multidimensional image data using the Peano scan*, IEEE Trans. Pattern Anal. Machine Intell., PAMI-5 (1983), pp. 520–526.
[36] O. Temam, *Investigating optimal local memory performance*, in Proceedings of the ACM International Conference on Architectural Support for Programming Languages and Operating Systems, San Diego, CA, 1998, ACM, New York, pp. 218–227.
[37] L. Velho and J. de Miranda Gomes, *Digital halftoning with space-filling curves*, Comput. Graphics, 25 (1991), pp. 81–90.
[38] T. Weinzierl, *Eine cache-optimale Implementierung eines Navier-Stokes Lösers unter besonderer Berücksichtigung physikalischer Erhaltungssätze*, diploma thesis, Institut für Informatik, TU München, Munich, Germany, 2005.
[39] C. Weiss, *Data Locality Optimizations for Multigrid Methods on Structured Grids*, Ph.D. thesis, Institut für Informatik, TU München, Munich, Germany, 2001.
[40] G. Zumbusch, *Adaptive Parallel Multilevel Methods for Partial Differential Equations*, Habilitationsschrift, Mathematisch-Naturwissenschaftliche Fakultät, Universität Bonn, Bonn, Germany, 2001.
[41] G. W. Zumbusch, *On the quality of space-filling curve induced partitions*, Z. Angew. Math. Mech., 81 (suppl. 1) (2001), pp. 25–28.

SIAM J. SCI. COMPUT.
Vol. 28, No. 5, pp. 1651–1674
© 2006 Society for Industrial and Applied Mathematics

RECYCLING KRYLOV SUBSPACES FOR SEQUENCES OF LINEAR SYSTEMS*

MICHAEL L. PARKS†, ERIC DE STURLER‡, GREG MACKEY§, DUANE D. JOHNSON¶, AND SPANDAN MAITI‖

Abstract. Many problems in science and engineering require the solution of a long sequence of slowly changing linear systems. We propose and analyze two methods that significantly reduce the total number of matrix-vector products required to solve all systems. We consider the general case where both the matrix and right-hand side change, and we make no assumptions regarding the change in the right-hand sides. Furthermore, we consider general nonsingular matrices, and we do not assume that all matrices are pairwise close or that the sequence of matrices converges to a particular matrix. Our methods work well under these general assumptions, and hence form a significant advancement with respect to related work in this area. We can reduce the cost of solving subsequent systems in the sequence by recycling selected subspaces generated for previous systems. We consider two approaches that allow for the continuous improvement of the recycled subspace at low cost. We consider both Hermitian and non-Hermitian problems, and we analyze our algorithms both theoretically and numerically to illustrate the effects of subspace recycling. We also demonstrate the effectiveness of our algorithms for a range of applications from computational mechanics, materials science, and computational physics.

Key words. sequence of linear systems, linear solvers, Krylov methods, truncation, restarting, Krylov subspace recycling, iterative methods

AMS subject classifications. 65F10, 65N22

DOI. 10.1137/040607277

1. Introduction.

We consider the solution of a long sequence of general linear systems

$$A^{(i)}x^{(i)} = b^{(i)}, \qquad i = 1, 2, \ldots, \tag{1.1}$$

where the matrix $A^{(i)} \in \mathbb{C}^{n\times n}$ and right-hand side $b^{(i)} \in \mathbb{C}^n$ change from one system to the next, and the systems are typically not available simultaneously. Such sequences arise in many problems.

*Received by the editors April 24, 2004; accepted for publication (in revised form) February 16, 2006; published electronically October 6, 2006.
http://www.siam.org/journals/sisc/28-5/60727.html

†Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL 61801, and Computational Mathematics and Algorithms, Sandia National Laboratories, P.O. Box 5800, MS 1110, Albuquerque, NM 87185 (mlparks@sandia.gov). Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under contract DE-AC04-94-AL85000. The work of this author was supported by the Computational Science and Engineering program at UIUC through two CSE fellowships.

‡Department of Mathematics, Virginia Tech, Blacksburg, VA 24061-0123 (sturler@vt.edu). The work of this author was supported by the Materials Computation Center (UIUC) through grants NSF DMR 99-76550 and DMR-0325939, and by the Center for Simulation of Advanced Rockets (UIUC) through grant DOE LLNL B341494.

§Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL 61801 (currently gemacke@sandia.gov). The work of this author was supported by the Center for Simulation of Advanced Rockets through grant DOE LLNL B341494.

¶Department of Materials Science, University of Illinois at Urbana-Champaign, Urbana, IL 61801 (duanej@uiuc.edu). The work of this author was supported through the Materials Computation Center through grant NSF DMR 99-76550.

‖Department of Aerospace Engineering, University of Illinois at Urbana-Champaign, Urbana, IL 61801 (maiti@uiuc.edu).

One important class of applications that we consider occurs in modeling fatigue and fracture via finite element analysis. These analyses use dynamic loading, requiring many loading steps, and rely on implicit solvers [13]. Generally, several thousand loading increments, each corresponding to a linear system, are required to resolve the fracture progression. The matrix and right-hand side, at each loading step, depend on the previous solution, so that only one linear system is available at a time. Further, the solution vectors are updates to a displacement vector and are therefore uncorrelated. Although the change from one linear system to the next is small, the cumulative change after many loading increments is significant. Rather than discarding the Krylov space generated when solving a linear system, we judiciously select a subspace and use it to reduce the number of iterations for solving the next system. We refer to this process as [illegible]. Clearly, similar sequences of linear systems arise from other (nonlinear) time-dependent applications. Another important source of such sequences of linear systems are Newton or Broyden-type methods for solving nonlinear equations and optimization problems.

We consider methods for the solution of sequences of general matrices, and do not assume that all matrices are pairwise close or that the sequence of matrices converges to a particular matrix. In addition, we make no assumptions on the right-hand-side vectors. A method that is effective under these assumptions must satisfy a number of properties. First, the method must be able to identify and converge to an effective subspace for recycling (the [illegible]) in a reasonable number of iterations, and it must be able to converge to an effective recycle space over the solution of multiple linear systems. Otherwise, a good recycle space may never be found for a sequence of changing matrices. Second, for efficiency a significant convergence improvement for the linear solver must be obtained with a relatively small recycle space. Third, the method must be able to converge quickly to an effective [illegible] recycle space for an updated matrix, and it must provide an inexpensive mechanism for regularly updating the recycle space to reflect the changes in the linear systems. As we will show, our proposed methods satisfy these properties, and they are effective under these general assumptions. As such, they form a significant advancement with respect to related work in this area.

We consider two approaches for the solution of (1.1) which are related to two existing truncated and restarted solvers. These solvers, GMRES-DR [21] and GCROT [7], were developed for solving single linear systems; both recycle a judiciously selected subspace between restarts to maintain good convergence. In the following, we define a [illegible] in the sense used by GCROT, wherein the iteration is restarted with a selected subspace and each new vector generated is orthogonalized against this subspace. We define a [illegible] as the computation between truncations or restarts. The first approach is to recycle an approximate invariant subspace and use it for deflation, following the GMRES-DR method. Since GMRES-DR cannot be adapted for recycling, we propose a more general method, GCRO-DR. The work by Rey and Risler [25, 26] has a similar motivation as that for GMRES-DR; see below. However, our implementation is cheaper, more effective, and more adaptive. An alternative idea is to recycle a subspace that minimizes the loss of orthogonality with the Krylov subspace from the previous system [7]. In either case, the recycled subspace will be utilized by first minimizing the residual over this subspace and then maintaining orthogonality with the image of this space in the Arnoldi recurrence. We will show that subspaces that are useful to retain for a subsequent cycle when solving a single linear system are also useful for the next linear system in a sequence if the matrix does

not change significantly. The proposed methods can be tuned to recycle a variety of subspaces based on computed information from the matrix, background knowledge of the application, and other information; this is discussed in [15]. However, for a first evaluation, it is reasonable to analyze the effectiveness of these existing methods appropriately modified for solving (1.1). The application of these methods to sequences of linear systems where both matrix and right-hand side change is new and provides a significant advance on related methods discussed below. Our proposed methods offer an efficient mechanism for continuous updating of the recycled subspace. As a result, they quickly adapt to gradual changes in matrices. Further, they make few assumptions on the linear systems and are applicable to general matrices. In addition, we provide theoretical motivation and careful experimental analysis of these methods. Our results show that significant convergence improvements are obtained using recycled subspaces of a small dimension. The application of an early version of GCROT to multiple right-hand sides was briefly discussed in [5]. The tuning of subspace recycling for diffuse optical tomography, leading to further convergence improvements, is discussed and relevant theory presented in [15].

We discuss the basic derivation of our methods from Morgan's GMRES-DR and de Sturler's GCROT in section 2. We modify GCROT to recycle subspaces between linear systems. GMRES-DR cannot be modified to do this, so we introduce GCRO-DR, a more general variant of GMRES-DR, capable of Krylov subspace recycling. In section 3, we discuss two important theorems and their consequences to explain why GCRO-DR satisfies the three essential properties for an effective recycling method. A similar convergence theory for GCROT is a topic of future research. This analysis is complemented by careful experimentation in section 4, where we provide an extensive experimental analysis to illustrate the behavior of both proposed methods and how well they satisfy the required properties. Particular examples from section 4.4 suggest that further improvements are possible in the subspace selection process, which is a subject of future research. Finally, we discuss conclusions and future work in section 5.

To conclude this section, we discuss a number of related approaches. For the Hermitian positive definite case, Rey and Risler have proposed reducing the effective condition number by either explicitly retaining all converged Ritz vectors from previous CG iterations or implicitly approximating the dominant invariant subspace by retaining all previously generated complete Krylov spaces [25, 26]. Moreover, both approaches use full recurrences, so the CG iteration is really a FOM (full orthogonalization method) iteration [27]. Clearly, both in memory and floating point operations, these methods are very expensive. This drawback is somewhat alleviated as the methods are presented in the context of the finite element tearing and interconnecting (FETI) method [10], which operates on a reduced-size problem. However, for general problems the approach would appear impractical. Furthermore, both approaches lack the possibility to gradually improve the recycle space. Instead, the recycle space is computed once and periodically replaced by starting from scratch. There is no effective mechanism for adapting this space for a sequence of linear systems that change gradually but substantially over many steps. For this reason, the authors make the explicit assumption that all matrices remain close. Although the approaches are motivated by well-known properties of CG, no theoretical or quantitative analysis is provided of the quality of the approximate invariant subspaces or the convergence rate for CG with recycling. Finally, the reported reductions in the number of iterations are modest at about 20%, even though large recycle spaces are used (e.g., of dimension 100 in [25]). In [26] the authors report about a 20% reduction in CPU-time with respect to CG with a full recurrence.

Another interesting approach for solving a sequence, or better a collection, of linear systems was proposed by Chan and Ng [3] developing two related Galerkin projection methods. Unfortunately, these methods require all systems to be available simultaneously, or at least the right-hand sides. They do not recycle subspaces that arise in the iteration, but instead use all vectors arising in the iteration immediately for all systems. Therefore, they focus on situations where all the matrices are close. However, this is generally not the case for the problems we target. Although the matrices change slowly, the cumulative change over many steps is usually significant.

Solving a sequence of linear systems where the matrix is fixed is a special case of (1.1). When all right-hand sides are available simultaneously, block methods are often suitable, such as block CG [22], block GMRES [37], and the family of block EN-like methods [39]. However, block methods do not generalize to the case (1.1). If only one right-hand side is available at a time, the method of Fischer [11], the deflated CG method [29], or the hybrid method of Simoncini and Gallopoulos [30] may be employed. Fischer's method looks for a starting vector in the space spanned by the previous solution vectors in the sequence, which is helpful if the solution vectors are correlated. The method does not maintain orthogonality to this subspace, and so no further speed-up is obtained. In deflated CG, only a small number of the initial Lanczos vectors for every system is used to update the approximate invariant subspace. This is efficient in computation and memory use, but the convergence to an invariant subspace is slow. Hence, the improvement in iterations is modest. The hybrid method of Simoncini and Gallopoulos is most effective when the right-hand sides share common spectral information.

2. Recycling Krylov methods. Restarting GMRES [28] may lead to poor convergence and even stagnation. Therefore, recent research has focused on truncated methods that improve convergence by retaining a judiciously selected subspace between cycles. A taxonomy of popular choices is given in [9]. In this section, we discuss two choices and solvers implementing them. We then modify those solvers to recycle subspaces between linear systems. Although these linear solvers are not new, their extension to sequences of linear systems has not been discussed and analyzed before.

One strategy for subspace selection was proposed in [7] for the GCROT method. We discuss this approach and its modification for solving (1.1) in section 2.2.

We discuss Morgan's GMRES-DR, which retains an approximate invariant subspace between cycles, in section 2.3. In particular, it focuses on removing the eigenvalues of smallest magnitude by recycling an approximate invariant subspace associated with those eigenvalues. Note that GMRES-DR must use only harmonic Ritz vectors, and that it cannot be modified for Krylov subspace recycling even when the matrix does not change. Therefore, we combine ideas from GCRO [6] and GMRES-DR to produce a new linear solver, GCRO-DR, which is suitable for the solution of individual linear systems as well as sequences of them, and is more flexible than GMRES-DR. We discuss GCRO-DR in section 2.4.

2.1. Definitions. The Arnoldi recurrence in GMRES leads to the following relation, which we denote as the Arnoldi relation:

$$AV_m = V_{m+1}\underline{H}_m, \tag{2.1}$$

where $V_m \in \mathbb{C}^{n\times m}$ and $\underline{H}_m \in \mathbb{C}^{(m+1)\times m}$ is upper Hessenberg. Let $H_m \in \mathbb{C}^{m\times m}$ denote the first m rows of $\underline{H}_m$.

For any subspace $S \subseteq \mathbb{C}^n$, $y \in S$ is a Ritz vector of A with Ritz value θ if

$$Ay - \theta y \perp w \qquad \forall w \in S. \tag{2.2}$$

Frequently, we choose $S = K^{(j)}(A, r)$, the jth Krylov subspace associated with the matrix A and the starting vector r. In this case the eigenvalues of H_m are the Ritz values of A. Ritz values tend to approximate the extremal eigenvalues of A well, but can give poor approximations to the interior eigenvalues. Likewise, the Ritz values of A^{-1} tend to approximate the interior eigenvalues of A. We define harmonic Ritz values as the Ritz values of A^{-1} with respect to the space AS,

$$A^{-1}\widetilde{y} - \widetilde{\mu}\widetilde{y} \perp w \qquad \forall w \in AS, \tag{2.3}$$

where again $S = K^{(j)}(A, r)$ and $\widetilde{y} \in AS$. We call $\widetilde{\theta} = 1/\widetilde{\mu}$ a harmonic Ritz value. In this case, we have approximated the eigenvalues of A^{-1}, but using a Krylov space generated with A. In GCRO-DR, we construct harmonic Ritz vectors using a modified operator rather than A.

2.2. GCROT. GCROT is a truncated minimum residual Krylov method that retains a subspace between cycles such that the loss of orthogonality with respect to the discarded space is minimized [7]. This process is called [illegible]. We discuss optimal truncation in the context of restarted GMRES, although it can be described in more general terms, independently of any specific linear solver [7, 18]. Consider solving $Ax = b$ with initial residual r_0. At the end of the first cycle of GMRES, starting with $v_1 = r_0/\|r_0\|_2$, we have the Arnoldi relation (2.1).

Let r_1 denote the residual vector after m iterations. Consider some iteration $s < m$. For s iterations of GMRES, we have the Arnoldi relation

$$AV_s = V_{s+1}\underline{H}_s. \tag{2.4}$$

Let r denote the residual after s iterations. Now suppose that we had restarted after iteration s, with initial residual r, and made $m - s$ iterations, yielding residual r_2. The optimal residual after m iterations is r_1. At best, we may have $\|r_2\|_2 = \|r_1\|_2$, but in general, $\|r_2\|_2 > \|r_1\|_2$, because GMRES restarted after iteration s ignores orthogonality to the Krylov subspace $AK^{(s)}(A, r_0)$. The deviation from optimality incurred by restarting after iteration s is $e = r_2 - r_1$, which we call the [illegible]. The residual error e depends on the [illegible] [12, pp. 603–604] between the subspaces $AK^{(s)}(A, r_0)$ and $AK^{(m-s)}(A, r)$. Instead of completely discarding the space $AK^{(s)}(A, r_0)$, suppose we had maintained orthogonality to a k-dimensional subspace ($k < s$) of $AK^{(s)}(A, r_0)$ for the remaining $m - s$ iterations to produce a new residual vector r_3. If we chose our k-dimensional subspace of $AK^{(s)}(A, r_0)$ to correspond to the k largest principal angles, we would minimize the norm of the new residual error $\|e'\|_2 = \|r_1 - r_3\|_2$. This process is what is meant by optimal truncation. Since the Krylov space generated with r contained vectors close to the recycled subspace, this is likely to happen again after restarting with r_1. Therefore, we retain the selected k-dimensional subspace for the next cycle.

GCROT maintains matrices $U_k, C_k \in \mathbb{C}^{n\times k}$ satisfying the relations

$$AU_k = C_k, \tag{2.5}$$

$$C_k^H C_k = I_k. \tag{2.6}$$

The minimum residual solution over range(U_k) is known from the previous cycle. In the following cycle, we carry out the Arnoldi recurrence while maintaining orthogonality to C_k. This corresponds to an Arnoldi recurrence with the operator $(I - C_k C_k^H)A$.

Then, we compute the update to the solution as in GMRES, taking the singularity of the operator into account using the relation $A^{-1}(I - C_k C_K^H)A = I - U_k C_k^H A$ [6]. The correction to the solution vector and other vectors selected via optimal truncation of the Krylov subspace are appended to U_k, and then U_k and C_k are updated such that (2.5)–(2.6) again hold. At the end of each cycle, only the matrices U_k and C_k are carried over to the next cycle. Each cycle of GCROT requires approximately $m-k$ matrix-vector products and $O(nm^2+nkm)$ other floating point operations. For details, see [7].

GCROT can be modified to solve (1.1) by carrying over U_k from the ith system to the $(i+1)$st system. We have the relation $A^{(i)}U_k = C_k$. We modify U_k and C_k to satisfy (2.5)–(2.6) with respect to $A^{(i+1)}$ as follows:

$$[Q, R] = \text{reduced QR decomposition of } A^{(i+1)}U_k^{old}, \tag{2.7a}$$

$$C_k^{new} = Q, \tag{2.7b}$$

$$U_k^{new} = U_k^{old} R^{-1}. \tag{2.7c}$$

[1]

Now, $A^{(i+1)}U_k^{new} = C_k^{new}$, and we can proceed with GCROT on the $(i+1)$st linear system. Note that in many cases computing $A^{(i+1)}U_k^{old} = C_k^{old} + \Delta A^{(i)}U_k^{old}$ is much cheaper than k matrix-vector products, because $\Delta A^{(i)}$ is considerably sparser than $A^{(i)}$ or has a special structure. See our example problem in section 4.1. Moreover, even if we were to compute $A^{(i+1)}U_k^{old}$ directly, this can be faster than k separate matrix-vector multiplications [8]. So long as $A^{(i+1)}$ has not changed significantly from $A^{(i)}$, the use of U_k^{new} should accelerate the solution of the $(i+1)$st linear system.

2.3. GMRES-DR. GMRES-DR [21] recycles an approximate invariant subspace to deflate eigenvalues of smallest magnitude. Deflating these eigenvalues can greatly improve convergence in certain circumstances.

In each cycle, GMRES-DR carries forward k harmonic Ritz vectors $\widetilde{Y}_k \in \mathbb{C}^{n\times k}$ computed at the end of the previous cycle. For the first cycle, the harmonic Ritz vectors can be computed from H_m in (2.1). It can be shown that these harmonic Ritz vectors fit naturally into a Krylov subspace [20]. In each cycle, GMRES-DR proceeds by first orthogonalizing $\widetilde{Y}_k$ to give $\widetilde{\Upsilon}_k$. GMRES-DR then carries out the Arnoldi recurrence for $m-k$ iterations while maintaining orthogonality to $\widetilde{\Upsilon}_k$. This gives the Arnoldi-like relation

$$A[\widetilde{\Upsilon}_k \;\; V_{m-k}] = [\widetilde{\Upsilon}_k \;\; V_{m-k+1}]\underline{H}_m, \tag{2.8}$$

where $\underline{H}_m$ is upper Hessenburg, except for a leading dense $(k+1)\times(k+1)$ submatrix. GMRES-DR updates the solution and residual as in GMRES. It then computes the harmonic Ritz vectors associated with the k smallest harmonic Ritz values using (2.8), and finally restarts with those vectors.

GMRES-DR cannot be used to solve (1.1) directly, even if the matrix is fixed. The harmonic Ritz vectors of A in $\widetilde{Y}_k$ do not form a Krylov subspace for another matrix or even just another starting vector. These and other reasons lead us to develop GCRO-DR, a generalization of GMRES-DR capable of solving (1.1).

2.4. GCRO-DR. We introduce a new Krylov method that uses recycling. We call this method GCRO-DR because it uses deflated restarting within the framework of GCRO [6]. The method is a generalization of GMRES-DR to solve (1.1). GCRO-DR is more flexible because any subspace may be recycled for subsequent cycles or

[1] For efficiency, U_k^{new} need not be computed explicitly.

linear systems. In the pseudocode given in the appendix, the harmonic Ritz vectors corresponding to the harmonic Ritz values of smallest magnitude have been chosen. However, any combination of k vectors may be selected. An interesting possibility would be to select a few [illegible] corresponding to the harmonic Ritz values of smallest magnitude, and a few [illegible] corresponding to the Ritz values of largest magnitude. This would allow simultaneous deflation of eigenvalues of both smallest and largest magnitude using the better approximation for each.

When solving a single linear system, GCRO-DR and GMRES-DR are algebraically equivalent. The primary advantage of GCRO-DR is its capability for solving sequences of linear systems.

GCRO-DR is a combination of GMRES-DR and GCRO. Suppose that we have solved the ith system of (1.1) with GCRO-DR, and we retain k approximate eigenvectors, $\widetilde{Y}_k = [\widetilde{y}_1, \widetilde{y}_2, \ldots, \widetilde{y}_k]$. Then, GCRO-DR computes matrices $U_k, C_k \in \mathbb{C}^{n\times k}$ from $\widetilde{Y}_k$ and $A^{(i+1)}$ such that $A^{(i+1)}U_k = C_k$ and $C_k^H C_k = I_k$, in the same manner as in (2.7). In the remainder of this section we drop the superscript in $A^{(i+1)}$ for notational convenience.

We find the optimal solution over the subspace range(U_k) as $x = x_0 + U_k C_k^H r_0$, and set $r = r_0 - C_k C_k^H r_0$ and $v_1 = r/\|r\|_2$. Next, we generate a Krylov space of dimension $m-k+1$ with $(I - C_k C_k^H)A$, which produces the Arnoldi relation

$$(I - C_k C_k^H)AV_{m-k} = V_{m-k+1}\underline{H}_{m-k}. \tag{2.9}$$

Since $V_{m-k+1} \perp C_k$, we have

$$A[U_k \;\; V_{m-k}] = [C_k \;\; V_{m-k+1}] \begin{bmatrix} I_k & B_k \\ 0 & \underline{H}_{m-k} \end{bmatrix}, \tag{2.10}$$

where $B_k = C_k^H A V_{m-k}$. To reduce unnecessary ill-conditioning of the rightmost matrix in (2.10) we proceed as follows. We compute the diagonal matrix D_k such that $\widetilde{U}_k = U_k D_k$ has unit columns, and we define

$$\widehat{V}_m = [\widetilde{U}_k \;\; V_{m-k}], \quad \widehat{W}_{m+1} = [C_k \;\; V_{m-k+1}], \quad \underline{G}_m = \begin{bmatrix} D_k & B_k \\ 0 & \underline{H}_{m-k} \end{bmatrix}.$$

We rewrite (2.10) as

$$A\widehat{V}_m = \widehat{W}_{m+1}\underline{G}_m, \tag{2.11}$$

where the columns of $\widehat{V}_M$ and $\widehat{W}_{m+1}$ have unit norm. Note that $\underline{G}_m = \widehat{W}_{m+1}^H A\widehat{V}_m$ is upper Hessenberg, with D_k diagonal. The columns of $\widehat{W}_{m+1}$ are orthogonal, but this is not true for the columns of $\widehat{V}_m$.

At the end of each cycle, GCRO-DR solves the minimization problem

$$t = \arg \min_{z \in \text{ range}(\widehat{V}_m)} \|r - Az\|_2, \tag{2.12}$$

which reduces to the $(m+1)\times m$ least squares problem

$$\underline{G}_m y \approx \widehat{W}_{m+1}^H r = \|r\|_2 e_{k+1}, \tag{2.13}$$

with $t = \widehat{V}_m y$. The residual and solution are given by

$$r = r - A\widehat{V}_m y = r - \widehat{W}_{m+1}\underline{G}_m y, \tag{2.14}$$

$$x = x + \widehat{V}_m y. \tag{2.15}$$

Next, the method solves the generalized eigenvalue problem

$$\underline{G}_m^H \underline{G}_m \widetilde{z}_i = \widetilde{\theta}_i \underline{G}_m^H \widehat{W}_{m+1}^H \widehat{V}_m \widetilde{z}_i, \tag{2.16}$$

derived from (2.3), and recovers the harmonic Ritz vectors as $\widetilde{y}_i = \widehat{V}_m \widetilde{z}_i$. In general, (2.16) will have complex eigenvalues. When storing the harmonic Ritz vectors in GCRO-DR, we use a real representation of a complex conjugate pair. To ensure that we retain both complex conjugate harmonic Ritz vectors associated with a selected eigenvalue, it is sometimes necessary to store $k+1$ vectors rather than k vectors per cycle.

GCRO-DR and GMRES-DR have about the same computational cost per cycle. In particular, they have the same number of matrix-vector products and orthogonalizations per cycle. GCRO-DR stores k additional vectors. If the new U_k is computed explicitly (which is not necessary), GCRO-DR has a modest additional computational cost of about $nk^2/2$. Given U_k and C_k, generating (2.11) with GCRO-DR(m,k) requires approximately $2kn\,(1+k)$ floating point operations than generating (2.1) with GMRES(m), although GMRES(m) stores k fewer vectors. The number of dot-products and vector updates per cycle is of the same order for GCRO-DR(m,k) and GMRES(m); the cost savings in GCRO-DR(m,k) arise because (2.5) and (2.6) are already satisfied.

3. Convergence analysis for deflation-based Krylov subspace recycling. Recent work on the convergence of GMRES [31] together with the theory on invariant subspaces and their perturbations [34] provides a good framework for analyzing the GCRO-DR method. Unfortunately, a similar convergence theory for the GCROT method is still lacking. However, in section 4 we show by numerical experiment that, regarding recycling, GCROT shares many of the properties of GCRO-DR. A full theoretical analysis of GCRO-DR is beyond the scope of the present paper; instead we discuss two main theoretical results and their implications and demonstrate these numerically in section 4. For more details on these theoretical results we refer to [15, 23, 24].

The first result concerns the convergence of GCRO-DR; see [23, 24]. We show that the recycle space need not approximate an invariant subspace accurately to improve the rate of convergence significantly.

Let $\mathcal{Q}$ be an ℓ-dimensional invariant subspace of A, and let $\mathcal{C} = \text{range}(C_k)$ be a k-dimensional space ($k \geq \ell$) selected to approximate $\mathcal{Q}$, where C_k is defined in section 2.4. We define $\Pi_{\mathcal{C}}$ to be the orthogonal projector onto $\mathcal{C}$, and we define $\Pi_{\mathcal{Q}}$ similarly. Furthermore, we define $P_{\mathcal{Q}}$ to be the spectral projector onto $\mathcal{Q}$. Finally, we define the from the subspace $\mathcal{Q}$ to the subspace $\mathcal{C}$ as

$$\delta(\mathcal{Q},\mathcal{C}) \equiv \|(I-\Pi_{\mathcal{C}})\,\Pi_{\mathcal{Q}}\|_2, \tag{3.1}$$

which is equal to the sine of the largest principal angle between $\mathcal{Q}$ and $\mathcal{C}$ [1]. This means that any unit vector in $\mathcal{Q}$ has a component of at most length δ orthogonal to $\mathcal{C}$.

THEOREM 3.1. $\mathcal{C}$ $\mathcal{V} = \text{range}\left(V_{m-k+1}\underline{H}_{m-k}\right)$ $(m-k)$ (2.9) $r_0 \in \mathbb{C}^n$ $r_1 = (I-\Pi_{\mathcal{C}})\,r_0$ $\mathcal{Q}$ $\delta(\mathcal{Q},\mathcal{C}) < 1$

$$\min_{d_1 \in \mathcal{V}\oplus\mathcal{C}} \|r_0 - d_1\|_2 \leq \min_{d_2 \in (I-P_{\mathcal{Q}})\mathcal{V}} \|(I-P_{\mathcal{Q}})\,r_1 - d_2\|_2 + \frac{\gamma}{1-\delta}\|P_{\mathcal{Q}}\|_2 \cdot \|(I-\Pi_{\mathcal{V}})\,r_1\|_2,$$

$\gamma = \|(I-\Pi_{\mathcal{C}})P_{\mathcal{Q}}\|_2$

A

$$\min_{d_1 \in \mathcal{V} \oplus \mathcal{C}} \|r_0 - d_1\|_2 \leq \min_{d_2 \in (I - \Pi_{\mathcal{Q}})\mathcal{V}} \|\,(I - \Pi_{\mathcal{Q}})\, r_1 - d_2\|_2 + \frac{\delta}{1-\delta} \cdot \|\,(I - \Pi_{\mathcal{V}})\, r_1\|_2.$$

. For the proof, see [23, Chapter 3] or [24]. □

Theorem 3.1 was inspired by a related theorem in [31] used to explain superlinear convergence in GMRES. In the two bounds above, the left-hand side represents the residual norm after $m-k$ iterations of GCRO-DR with the recycled subspace $\mathcal{C}$. On the right-hand sides, the first term represents the convergence of a deflated problem where all components in the subspace $\mathcal{Q}$ have been removed, which typically leads to an improved rate of convergence [21, 31, 36]. The second term in the right-hand sides represents a constant times the residual of $m-k$ iterations of GCRO-DR, solving for r_1. If the recycle space $\mathcal{C}$ contains an invariant subspace $\mathcal{Q}$, then $\delta = \gamma = 0$ for this $\mathcal{Q}$, and GCRO-DR converges at least as fast as the deflated problem. In our numerical experiments we demonstrate that the method fairly quickly gets to values of $\delta = O(10^{-2})$. In that case, we still obtain the convergence rate of the deflated problem, so long as $\|P_{\mathcal{Q}}\|_2$ is not large in the non-Hermitian case. Notice that for $\delta = O(10^{-2})$ the invariant subspace $\mathcal{Q}$ is not approximated very accurately, and that such values of δ are relatively easily obtained. Finally, we point out that for a given subspace $\mathcal{C}$, Theorem 3.1 is applicable to any invariant subspace $\mathcal{Q}$ such that $\delta(\mathcal{Q}, \mathcal{C}) < 1$. Hence, the sharpest bound for any $\mathcal{Q}$ applies, and the result appears to be fairly insensitive to conditioning issues of invariant subspaces for non-Hermitian A.

The second result concerns the perturbation of invariant subspaces associated with the smallest eigenvalues when the change in the matrix is concentrated in an invariant subspace corresponding to large eigenvalues. When the magnitude of the change is smaller than the gap between smallest and large eigenvalues, then the invariant subspace associated with the smallest eigenvalues is not significantly altered. As we aim to recycle exactly this subspace, this is a desirable property.

For simplicity we deal specifically with a Hermitian positive definite matrix A and a corresponding Hermitian perturbation E, as in our main numerical example in section 4. Following the discussion in [15], let A have the eigendecomposition

$$A = [Q_1\ Q_2\ Q_3]\, \mathrm{diag}(\Lambda_1, \Lambda_2, \Lambda_3)\, [Q_1\ Q_2\ Q_3]^H, \tag{3.2}$$

where $Q = [Q_1\ Q_2\ Q_3]$ is an orthogonal matrix, $\Lambda_1 = \mathrm{diag}(\lambda_1^{(1)}, \ldots, \lambda_{j1}^{(1)})$, and Λ_2 and Λ_3 are defined analogously. Furthermore, let

$$\lambda_1^{(1)} \leq \cdots \leq \lambda_{j1}^{(1)} < \lambda_1^{(2)} \leq \cdots \leq \lambda_{j2}^{(2)} < \lambda_1^{(3)} \leq \cdots \leq \lambda_{j3}^{(3)}.$$

Now we consider the change in the invariant subspace range(Q_1) under a symmetric perturbation E of A. Let $\theta_1(.,.)$ denote the largest canonical angle between two spaces. We do not require that $\|E\|_F$ be small, but we assume that the projection of E onto the subspace range$([Q_1\ Q_2])$ is small. We assume that $\|[Q_1\ Q_2]^H E\|_F \leq \varepsilon$ and that ε is small relative to $\lambda_1^{(2)} - \lambda_{j1}^{(1)}$. We also assume that $\eta \equiv \|Q_3^H E\|_F$ is small relative to $\lambda_1^{(3)} - \lambda_{j1}^{(1)}$. Note that we do not need to assume that $\lambda_1^{(2)} - \lambda_{j1}^{(1)}$ is large.

Also, let

$$\mu \equiv \min(\lambda_1^{(2)} - \varepsilon, \lambda_1^{(3)} - \eta) - 2\varepsilon - (\lambda_{j1}^{(1)} + \varepsilon) > 2\varepsilon,$$

$$\hat{\mu} \equiv \mu \left(1 - \frac{2\varepsilon^2}{\mu^2}\right) + \lambda_{j1}^{(1)} + \varepsilon.$$

THEOREM 3.2. A [illegible] (3.2), [illegible] E ε η μ $\hat{\mu}$ [illegible] $\hat{Q}_1$ [illegible] Q_1 [illegible] range$(\hat{Q}_1)$ [illegible] $A + E$

$$\tan\theta_1\left(\text{range}(Q_1), \text{range}(\hat{Q}_1)\right) \leq \frac{\varepsilon}{\hat{\mu}}.$$

[illegible]. For the proof, see [15]. □

A similar bound holds for the perturbation of the eigenvalues associated with Q_1.

In relation to Theorem 3.1 and GCRO-DR, Q_1 corresponds to $\mathcal{Q}$, whereas Q_2 and Q_3 can be chosen to fit the theorem. We need a specialized perturbation result of this kind, because in general the changes in the matrices are too large to show by standard perturbation theory that the invariant subspaces of interest remain intact. This perturbation theorem is of general importance, as there are many applications that involve a sequence of problems that undergo small local changes. In a problem like crack propagation we do not expect the smooth global modes associated with the smallest eigenvalues to change much from one system to the next, but only gradually over many systems as the crack propagates over some distance. We show experimentally that this is the case in section 4. The continual adaptation of the recycle space needs to track only these gradual changes. Another example is given in [15], where we optimize a parameterized medium in a tomography application to fit measured data.

Finally, we note that GCRO-DR uses more or less the Arnoldi method with dense restarting for approximating an invariant subspace [33, 38]. This method generally offers fast convergence for the exterior components of the spectrum. This fast convergence together with Theorems 3.1 and 3.2 indicates that GCRO-DR satisfies the three important properties mentioned in the introduction, especially for problems such as crack propagation.

The perturbation result indicates that if the recycle space provides a reasonable approximation to an invariant subspace, then it will also provide a reasonable approximation to the slightly perturbed invariant subspace of the updated matrix. The convergence result indicates that GCRO-DR will have fast convergence as the method does not require an accurate approximation to the invariant subspace. Moreover, Arnoldi's method with dense restarting will quickly improve the approximation to the invariant subspace of the perturbed matrix. In general, it is not hard to get a reasonable approximation to an invariant subspace corresponding to the outermost eigenvalues [34]. We will demonstrate this behavior of GCRO-DR using numerical experiments in the next section.

4. Test problems and numerical results. We discuss our main example in section 4.1, a problem from fracture mechanics that produces a long sequence of linear systems. The matrices are symmetric positive definite (SPD), and both the matrix and right-hand side change from one system to the next. For this problem we also provide a more detailed experimental analysis of the GCRO-DR and GCROT methods following the theory described in section 3. In addition, we provide results for

three problems that involve real nonsymmetric matrices and complex non-Hermitian matrices. We consider two examples from physics to illustrate the effectiveness of our approach for the case of a fixed matrix. We discuss electronic structure calculations in section 4.2, and a problem from lattice quantum chromodynamics in section 4.3. Finally, in section 4.4, we apply GCROT and GCRO-DR to a simple convection-diffusion problem to evaluate the effects of subspace recycling in the nonsymmetric case, independent from perturbations in the matrix or right-hand side.

In the following sections, GMRES(m) indicates restarted GMRES with a maximum subspace of dimension m, and GMRES(∞) indicates full GMRES. CG refers to the conjugate gradient method. For GMRES-DR(m,k) and GCRO-DR(m,k), m is the maximum subspace size, and k is the dimension of the recycle space. For GCROT(m,k_{max},k_{min},s,p_1,p_2), m is the maximum subspace size over which we optimize. The maximum number of column vectors stored in U_k and C_k (as described in section 2.2) is k_{max}. The argument k_{min} indicates the number of column vectors retained in U_k and C_k after truncation to make room for new vectors. The argument s indicates the dimension of the Krylov subspace from which we select p_1 vectors to place in U_k. We also include in U_k the last p_2 orthogonal basis vectors generated in the Arnoldi process. See [7, 18] for more discussion regarding the choice of parameters. At each restart for GCROT, the GMRES part is run for $m - k_{min}$ steps.

In comparing restarted GMRES, GCROT, GMRES-DR, and GCRO-DR, we decided to make the solvers minimize over a subspace of the same dimension. An alternative choice would be to provide the same amount of memory to each solver, but we felt that our choice would provide a more informative comparison.

4.1. Fatigue and fracture of engineering components. Research on failure mechanisms (e.g., fatigue and fracture) of engineering components often focuses on modeling complex, nonlinear response. Finite element methods for quasi-static and transient responses over longer time scales generally adopt an implicit formulation. Together with a Newton scheme for the nonlinear equations, such implicit formulations require the solution of linear systems, thousands of times, to accomplish a realistic analysis [13].

We study a sequence of linear systems taken from a finite element code developed by Philippe Geubelle and Spandan Maiti (both Aerospace Engineering, University of Illinois at Urbana-Champaign (UIUC)). In our example, the code simulates crack propagation in a metal plate using so-called [illegible]. The plate mesh is shown in Figure 4.1. The problem is symmetric about the x-axis, and the crack propagates exactly along this symmetry axis. The cohesive elements act as nonlinear springs connecting the surfaces that will define the crack location. As the crack propagates, the cohesive elements deform following a nonlinear yield curve and eventually break. The element stiffness is set to zero for a broken cohesive element. These elements are usually inserted dynamically, but that is not the case here. This simulation results in a sequence of sparse, symmetric, positive definite stiffness matrices that change slowly from one system to the next. Each stiffness matrix can be expressed as $A^{(i+1)} = A^{(i)} + \Delta A^{(i)}$. Although $\Delta A^{(i)}$ is considerably more sparse than $A^{(i)}$, it is not low-rank, as the terms in the update $\Delta A^{(i)}$ come from all nonbroken cohesive elements. The other finite elements model linear elasticity and have constant stiffness matrices. The matrices produced in our examples are 3988×3988, and have a condition number on the order of 10^4 before preconditioning. They have an average of 13.4 nonzero entries per row. Over 2000 linear systems must be solved to capture the fracture progression.

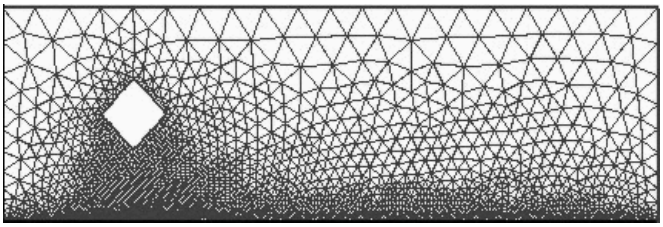


FIG. 4.1. *Two-dimensional plate mesh for the crack propagation problem.*

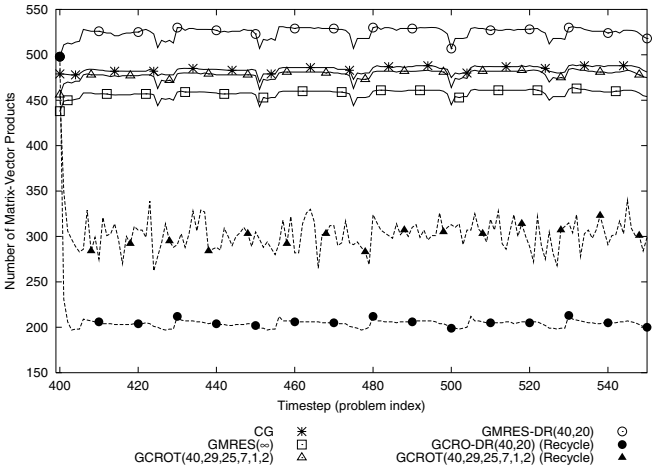


FIG. 4.2. *Number of matrix-vector products versus timestep for various solvers for the crack propagation problem without preconditioning. All solvers use a recurrence of at most* 40 *vectors, except GMRES*(∞) *and CG.*

We examine the 151 linear systems 400–550, representing a typical subset of the fracture progression in which many cohesive elements break. We start with a straightforward comparison of GCRO-DR and GCROT with subspace recycling with CG, restarted GMRES, GMRES(∞), GMRES-DR, and GCROT without subspace recycling. We always start with a zero initial guess, since we solve for the incremental displacement associated with the loading increment. We give both preconditioned and nonpreconditioned convergence results. All solvers are required to reduce the relative residual to 1.0×10^{-10}. In Figure 4.2 we give the number of matrix-vector products needed to solve each of these systems without preconditioning for GMRES(∞), CG, GMRES-DR(40,20), GCRO-DR(40,20), and GCROT(40,34,30,5,1,2), both with and without subspace recycling. Except for GMRES(∞) and CG, all methods in Figure 4.2 minimize over a subspace of dimension 40 in each cycle. GMRES(40) is not shown, because it required too many matrix-vector products. In Figure 4.3 we give results for the same sequence of problems and the same methods but with incomplete Cholesky (IC(0)) preconditioning. A new preconditioner was computed for each matrix, which is not the most efficient approach. The total number of matrix-vector products needed to solve all 151 preconditioned linear systems is given in Table 4.1.

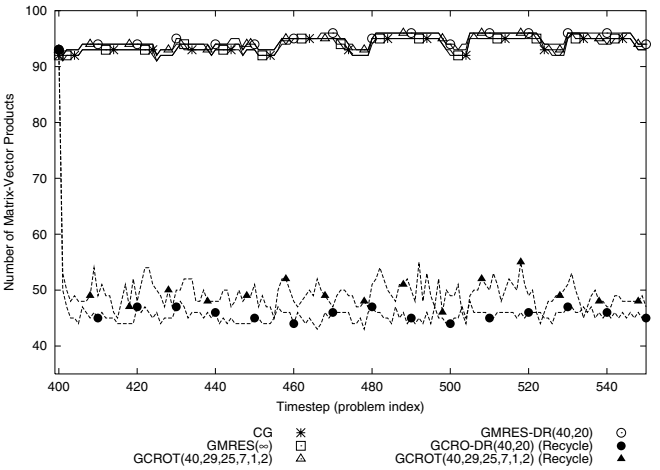


FIG. 4.3. *Number of matrix-vector products versus timestep for various solvers for the crack propagation problem with incomplete Cholesky preconditioning. All solvers use a recurrence of at most* 40 *vectors, except GMRES*(∞) *and CG.*

TABLE 4.1
The total number of iterations required to solve 151 *consecutive IC*(0) *preconditioned linear systems. Only GCRO-DR and GCROT(recycle) exploit subspace recycling.*

Method	Matrix-vector products
GMRES(40)	27188
GMRES-DR(40,20)	14305
GCROT(40,34,30,5,1,2)	14277
CG	14162
GMRES(∞)	14142
GCROT(40,34,30,5,1,2) (recycle)	**7482**
GCRO-DR(40,20) (recycle)	**6901**

We see in Figure 4.2 that GCRO-DR requires the fewest matrix-vector products, except for the first system in the sequence when no recycle space is available. For the first system, GCROT outperforms GCRO-DR. For all but the first system, GCRO-DR and GCROT with recycling require significantly fewer matrix-vector products than the solvers without subspace recycling. Furthermore, GCROT without recycling and CG show about the same convergence. GMRES(∞) outperforms CG, indicating that the convergence of CG is delayed due to effects of finite-precision arithmetic.

For the preconditioned case shown in Figure 4.3, GCRO-DR performs best, with GCROT with subspace recycling a close second. All the other solvers cluster near GMRES(∞).

Comparing GMRES-DR and GCRO-DR, we see a significant difference in convergence, even though both methods aim to retain the same approximate invariant subspace. This difference is due solely to subspace recycling. Without the space recycled from previous linear systems, GCRO-DR is algebraically equivalent to GMRES-DR. The results indicate that the invariant subspace associated with the smallest eigenvalues is hard to estimate accurately. GCRO-DR exhibits superior performance, because it does not have to compute that space from scratch for each new linear system. Indeed, we show below that after solving only three linear systems, GCRO-DR has computed a reasonable approximation to the invariant subspace associated with

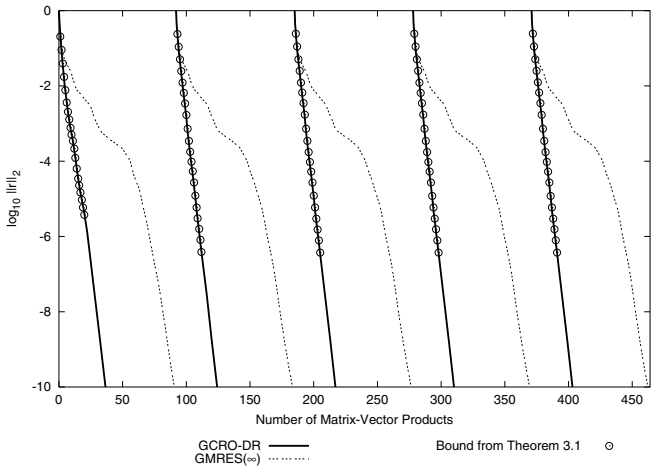


FIG. 4.4. *Convergence curves for GCRO-DR on five consecutive linear systems from the preconditioned crack propagation problem, along with the bound described in Theorem* 3.1. *The bound was computed with* $\mathcal{Q}$ *the eigenspace corresponding to the* 20 *smallest eigenvalues of the preconditioned operator. Since the bound is tight and* δ *is small, GCRO-DR achieved the convergence rate of the deflated problem in Theorem* 3.1.

TABLE 4.2

Table of $\delta(\mathcal{Q}, \mathcal{C})$ *for GCRO-DR*$(40, 20)$ *on the preconditioned crack propagation problem, where* $\mathcal{Q}$ *has been selected as the invariant subspace associated with the* 15 *eigenvalues of smallest magnitude. The decrease of* δ *within a column indicates the improvement of the approximation to* $\mathcal{Q}$ *for a particular system. The small perturbation of the invariant subspace is reflected in the minor increase in* δ *from the last cycle of one system to before the first of the next system. At each cycle, GCRO-DR updates its approximate invariant subspace.*

Linear system	401	402	403	404	405
Before cycle 1	0.9994	0.2380	0.0464	0.0607	0.0794
After cycle 1	0.9180	0.1448	0.0463	0.0606	0.0787
After cycle 2	0.2446	0.0302	0.0416	0.0568	0.0690
After cycle 3	0.2331	0.0302	0.0415	0.0567	0.0684

the fifteen smallest eigenvalues; GMRES-DR cannot do this.

Next, we evaluate some of the properties of GCRO-DR related to the two theoretical results in section 3 and the three properties mentioned in the introduction.

As GCRO-DR approximates an invariant subspace better, it gets closer to the convergence rate of a deflated problem, as described in Theorem 3.1. The convergence curves for five consecutive systems from the preconditioned crack propagation problem are shown in Figure 4.4, along with the bound described in Theorem 3.1. The bound was computed with the invariant subspace, $\mathcal{Q}$, corresponding to the 20 smallest eigenvalues of the preconditioned matrix. We see that the bound is sharp. Additionally, note that for each problem the initial convergence rate for GCRO-DR is approximately the same as the final convergence rate of GMRES(∞).

In Table 4.2, for the preconditioned example we show how the one-sided distance δ, defined in (3.1), from the invariant subspace associated with the 15 smallest eigenvalues to the recycle space evolves over multiple cycles and multiple linear systems. This table illustrates a number of important issues.

The first system solved in this experiment is system 400, and so system 401 is the first system that starts with a recycle space. For the chosen invariant subspace δ is

quite poor at the start, but δ improves quickly over the next two linear systems, and after cycle 2 for system 402, δ has taken a reasonably small value. So, GCRO-DR is able to compute from scratch increasingly good approximations to invariant subspaces that change slightly from system to system. Note that GMRES-DR starts from scratch for every new system. Therefore, it never obtains a sufficiently good approximation, and hence requires many additional iterations per linear system. Any other method that cannot continually update the approximation to the invariant subspace will suffer from the same problem.

Next, note how δ improves during the solution of a single linear system, whereas the small perturbation of the invariant subspace for the next linear system is reflected in the minor increase in δ from the last cycle of one system to before the first cycle of the next system. This confirms our earlier statement, based on Theorem 3.2, that the invariant subspace associated with the smallest eigenvalues and smooth global modes changes little under small local changes in the model. We have also verified this explicitly by computing the principal angles between invariant subspaces of successive matrices. However, in the next experiment we show that the cumulative change over a larger number of loading steps requires the continual or at least periodic updating of the recycle space to keep the number of iterations small.

Finally, this particular $\mathcal{Q}$ was chosen to illustrate the role and behavior of δ, not to get the best bound from Theorem 3.1. Initially, a smaller invariant subspace might have a much smaller δ and may lead to a tighter bound. The bound holds for any invariant subspace with $\delta < 1$, and so the [illegible] bound is the effective one.

Next, we provide a number of experiments that illustrate how quickly GCRO-DR and GCROT learn and adapt to an updated linear system, and how the convergence rate deteriorates if we stop updating the recycle space.

Figure 4.5 shows the results for three variants of GCRO-DR applied to the non-preconditioned problem. Standard GCRO-DR, as discussed so far, updates the recycle space every cycle, as this update is generally quite cheap. We also show a GCRO-DR variant that runs standard GCRO-DR for the first five linear systems in the sequence, but after that never updates the approximate invariant subspace. This also means that it makes no updates while solving each single system. In this case, convergence deteriorates slowly for the first few linear systems but then more rapidly as the gradual changes in the matrices accumulate. Eventually, the number of iterations per linear system exceeds that of GMRES-DR (cf. Figure 4.2). The third variant of GCRO-DR runs normal GCRO-DR for the first five linear systems in the sequence, then stops updating, except that it runs a standard GCRO-DR every 250th linear system. This frequency was chosen solely for the purpose of demonstration. Clearly, for this problem, the recycled subspace must be updated more frequently for good convergence, say every 25 or 50 systems. Note how, despite the significant deterioration in the number of iterations per system, GCRO-DR recovers the rate of convergence of standard GCRO-DR over the solution of a single system. This shows how quickly the method is able to recover (learn) a sufficiently accurate approximation to the invariant subspace. This example also suggests that the convergence is quite sensitive to the δ values for the relevant invariant subspaces. For this example, the average value of $\|A^{(i)}\|_2$ is 3.1357×10^{10}, and the average value of $\|\Delta A^{(i)}\|_2$ is 2.9076×10^8. In a relative sense, the change from one matrix to the next is small, but the cumulative change over many linear systems can be quite large. As mentioned before, $\|\Delta A^{(i)}\|_2$ is larger than the smallest eigenvalues. However, the values of δ in Table 4.2 show that the perturbation of the corresponding invariant subspace is quite small, as suggested

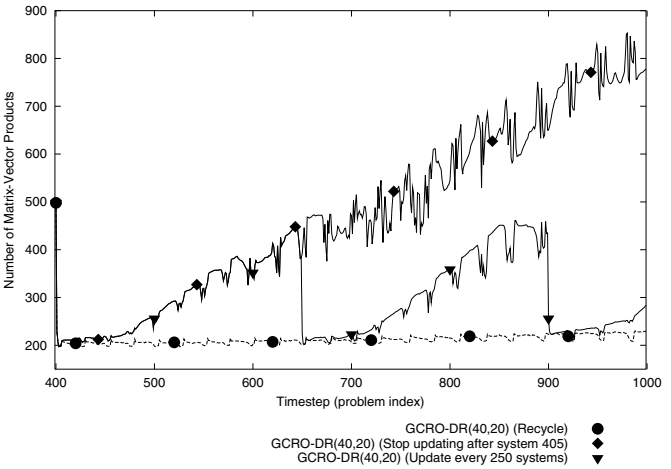


FIG. 4.5. *Number of matrix-vector products versus timestep for GCRO-DR*$(40, 20)$ *with subspace recycling on the nonpreconditioned crack propagation problem. We also show a modified version of GCRO-DR*$(40, 20)$ *that stops updating the recycled subspace after linear system* 405*, and a version that stops updating after linear system* 405 *but does update every* 250 *systems thereafter. Convergence quickly deteriorates in the latter two cases, showing the importance of updating the approximate invariant subspace regularly.*

by the discussion in section 3 regarding small localized changes in the problem and by Theorem 3.2. Figure 4.5 shows that GCRO-DR can adapt to the slow change of the invariant subspace. Finally, it would not be hard to dynamically balance the cost of updating the recycle space with the improved rate of convergence.

In Figure 4.6, we give the results for a similar experiment with GCROT. We show results for standard GCROT with recycling, a GCROT variant that stops updating the recycle space after the first five linear systems, and for a variant that stops updating after the first five linear systems except that it runs a standard GCROT every 100th system. Without the continual updating of the recycle space, the convergence of GCROT degrades more severely than that of GCRO-DR. After each linear system solved by standard GCROT, the number of iterations per system remains small for four or five subsequent systems, but then increases rapidly. This is likely caused by two factors. Since GCROT recycles a subspace that is important for maintaining orthogonality for the current iterations, the recycled space may have a more temporary relevance than an invariant subspace. If we stop updating, then GCROT more or less reduces to restarted GMRES after a number of updates to the matrix when the recycle space is no longer relevant. In addition, after solving a linear system, standard GCROT recycles a subspace that was relevant at the end of that iteration, which may not be the best recycle space for the start of the iteration for the next system. Note that solving a single linear system with standard GCROT restores the convergence rate to that for using standard GCROT for each system. This shows that the method is able to , , , quickly.

These experiments demonstrate that GCRO-DR and GCROT are able to make the small changes to the recycle space necessitated by small localized changes in the problem. For GCRO-DR this is explained by the perturbation result from Theorem 3.2 in combination with the convergence predicted by Theorem 3.1 for small enough δ (say, $\delta = O(10^{-2})$).

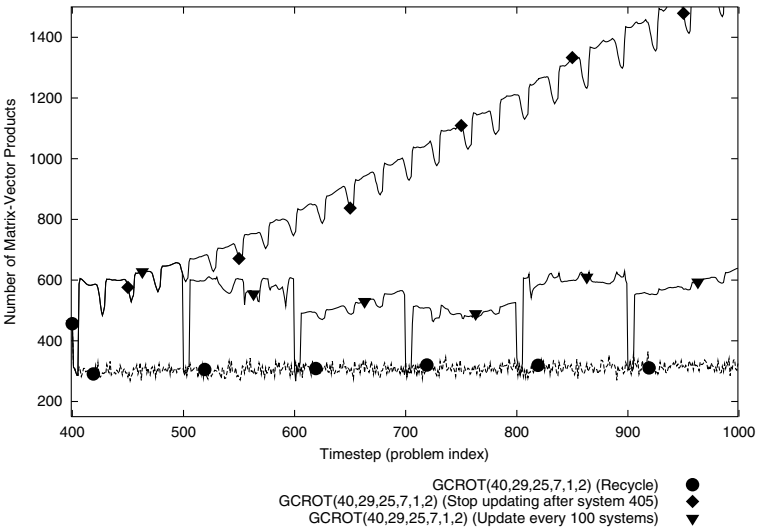


FIG. 4.6. *Number of matrix-vector products versus timestep for* $GCROT(40, 29, 25, 7, 1, 2)$ *with subspace recycling on the nonpreconditioned crack propagation problem. We also show a modified version of* $GCROT(40, 29, 25, 7, 1, 2)$ *that stops updating the recycled subspace after linear system* 405*, and a version that stops updating after linear system* 405 *but runs standard GCROT every* 100*th system thereafter. In the latter case, convergence remains small for about four or five systems after using standard GCROT, but then quickly deteriorates, showing the importance of updating the approximate invariant subspace regularly. Note that GCROT, too, is able to restore rapid convergence in the course of a single linear system.*

4.2. Electronic structure. First-principles, electronic-structure calculations based on the Schrödinger equation are used to predict key physical properties of materials systems with a large number of atoms. We consider systems arising in the KKR (Korringa–Kohn–Rostoker) method [16, 17] and seek to compute entries in the inverse of the matrix

$$G = (I - (t - t_{\text{ref}})G_{\text{ref}})^{-1}(t - t_{\text{ref}}),$$

where G_{ref} is a sparse and easily invertible matrix and t and t_{ref} are block-diagonal matrices. G is a sparse, complex, non-Hermitian matrix whose relative number of nonzeros decreases with the number of atoms [14, 40, 32].

Only the block-diagonal elements of G^{-1} are needed to calculate physical properties, such as charge densities, total energy, force, and formation and defect energies. As such, we solve $GX = I$, column-by-column. Iterative methods offer the advantage of storing only those components of the inverse that we need. Standard direct inversion methods are infeasible for large numbers of atoms ($N \geq 500$) on regular workstations, because the memory and computational costs grow as $O(N^2)$ and $O(N^3)$, respectively.

We consider a small model problem provided by Duane Johnson (Materials Science and Engineering, UIUC) and Andrei Smirnov (Oak Ridge National Laboratory). The problem involves the simulation of a cubic lattice of 54 copper atoms (treated as inequivalent) for a complex energy point close to the real axis. This is the key physical regime for metals and leads to problems that converge poorly. The matrix is 864×864 and has about 300,000 nonzeros. However, for increasingly larger systems the matrix

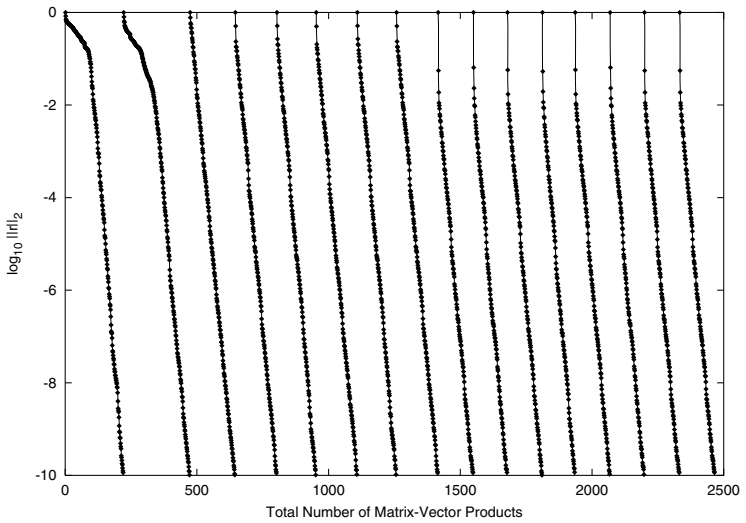


FIG. 4.7. *Convergence for* 16 *consecutive right-hand sides for a small electronic structure problem. Each distinct curve gives the convergence for a subsequent right-hand side, plotted against the total number of matrix-vector products. The first two right-hand sides together take about* 500 *iterations, while the remaining right-hand sides take about* 140 *iterations each, a reduction of almost* 50*%.*

becomes more sparse; the number of nonzeros grows roughly linearly with the size of the matrix. We solved this problem using GCRO-DR(50,25) with subspace recycling for 32 consecutive right-hand sides (the first 32 unit Cartesian basis vectors). We give the convergence history for the first atom in Figure 4.7. Note that the first two right-hand sides together take about 500 iterations; the remaining right-hand sides take approximately 140 iterations each, a reduction of almost 50%. Each right-hand side for the second atom (not shown) also takes approximately 140 iterations. Although for problems of this size iterative methods are not competitive with direct solvers, we have observed this convergence behavior for larger problems, in particular the immediate acceleration in convergence for subsequent right-hand sides.

4.3. QCD. Quantum chromodynamics (QCD) is the fundamental theory describing the strong interaction between quarks and gluons. Numerical simulations of QCD on a four-dimensional space-time lattice are considered the only way to solve QCD ab initio [4, 35]. As the problem has a 12×12 block structure, we are often interested in solving for 12 right-hand sides related to a single lattice site. The linear system to be solved is $(I - \kappa D)x = b$ with $0 \leq \kappa < \kappa_c$, where D is a sparse, complex, non-Hermitian matrix representing periodic nearest neighbor coupling on the four-dimensional space-time lattice [19]. For $\kappa = \kappa_c$ the system becomes singular. The physically interesting case is for κ slightly smaller than κ_c; κ_c depends on D.

As a model problem we use the matrix `conf5.0_0014x4.1000.mtx` downloaded from the Matrix Market website at NIST [2]. The model problems were submitted by Medeke [19]. For this problem we have $\kappa_c = 0.20611$, and we used $\kappa = 0.202$.

We solve for 12 consecutive right-hand sides (the first 12 Cartesian basis vectors) using the GCROT method with subspace recycling. The results are presented in Figure 4.8.

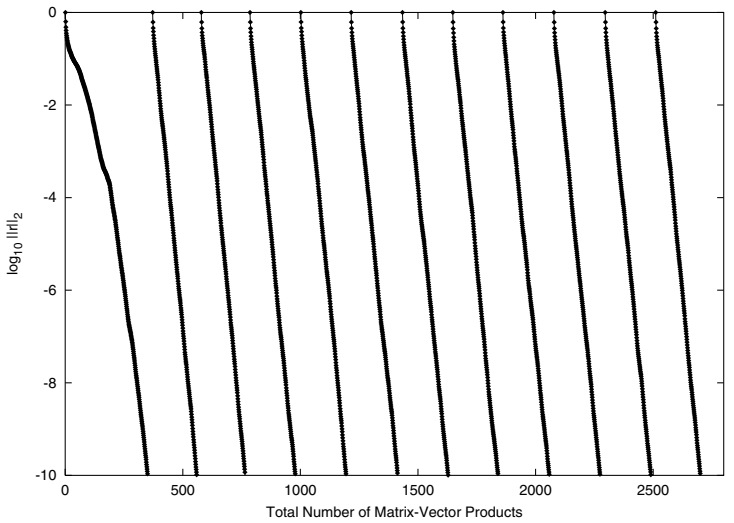


FIG. 4.8. *Convergence for* 12 *consecutive right-hand sides for a model QCD problem from the NIST Matrix Market. Each distinct curve gives the convergence for a subsequent right-hand side, plotted against the total number of matrix-vector products. Note that the final convergence rate for the first system is the starting convergence rate for the rest of the systems.*

4.4. Convection-diffusion. We consider the finite difference discretization of the partial differential equation

$$u_{xx} + u_{yy} + cu_x = 0$$

on $(0,1) \times (0,1)$ with boundary conditions

$$u(x,0) = u(0,y) = 0,$$
$$u(x,1) = u(1,y) = 1.$$

Central differences are used, and we set the mesh width to be $h = 1/41$ in both directions, which results in a 1600×1600 matrix. We consider the symmetric case, $c = 0$, and a nonsymmetric case for $c = 40$. The eigenvalues are real for both these examples. In order to study how a recycled subspace affects convergence, we will consider the "ideal" situation for subspace recycling by solving a linear system , , with GCRO-DR and GCROT, recycling the subspace generated from the first run.

In this example, we consider GMRES(∞), GMRES(25), GMRES-DR(25,10), GCRO-DR(25,10), and GCROT(25,18,15,5,1,1). To explore the effects of subspace recycling on this example problem, we , , GCRO-DR and GCROT on the same linear system, and recycle the subspace from the first run. We do this to exclude the effects of right-hand sides having slightly different eigenvector decompositions. In a sense, this is the ideal case for subspace recycling. The first run for GCRO-DR is the same as GMRES-DR. The results for the $c = 40$ (nonsymmetric) case are quite interesting and counterintuitive. In particular, this example suggests that there are better choices for a recycle space than approximate invariant subspaces. This topic is discussed further in [23]. We give the results for the symmetric case, $c = 0$, in Figure 4.9 and for the nonsymmetric case with $c = 40$ in Figure 4.10. In the legend for each of these figures, , , , denotes the second run of a solver that was run twice. All solvers were required to reduce the relative residual to 1.0×10^{-10}.

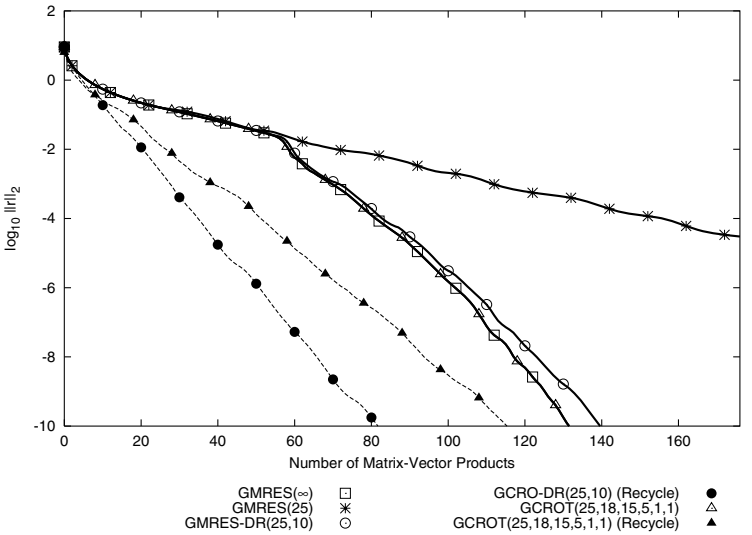


FIG. 4.9. *Residual norm versus number of matrix-vector products for various solvers for the convection-diffusion problem with* $c = 0$.

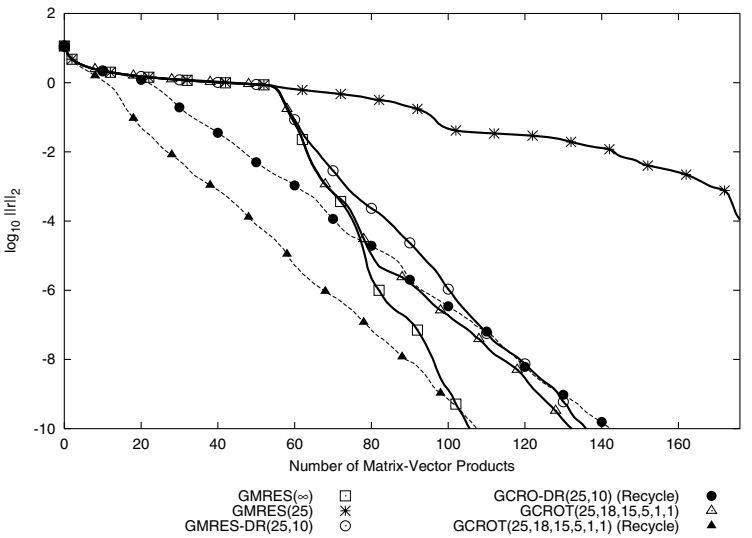


FIG. 4.10. *Residual norm versus number of matrix-vector products for various solvers for the convection-diffusion problem with* $c = 40$.

For the $c = 0$ case, we see that the second runs of GCRO-DR and GCROT both converged faster than GMRES(∞). Note that on its first run GCROT converges as fast as full GMRES and slightly faster than GCRO-DR/GMRES-DR. On the rerun, however, GCRO-DR is faster than GCROT, suggesting a more effective recycle space. Clearly, GCRO-DR and GCROT recycled a small subspace from their first run that improved convergence significantly. For the $c = 40$ case, GMRES(∞) and the second run of GCROT terminate in about the same number of iterations, but the second run of GCROT had a significantly smaller residual for almost the entire run. Only near

TABLE 4.3
Cosines of principal angles between the recycled subspace and the invariant subspaces spanned by the 10 *and* 21 *eigenvectors associated with the eigenvalues of smallest magnitude, respectively, for the* $c = 0$ *and* $c = 40$ *cases.*

Cosines of principal angles between recycled subspace and subspace associated with 10 smallest magnitude eigenvalues		Cosines of principal angles between recycled subspace and subspace associated with 21 smallest magnitude eigenvalues	
$c = 0$	$c = 40$	$c = 0$	$c = 40$
1.00000000000000	1.00000000000000	1.00000000000000	1.00000000000000
1.00000000000000	0.99999999999997	1.00000000000000	1.00000000000000
1.00000000000000	0.99999999839942	1.00000000000000	1.00000000000000
1.00000000000000	0.99999970490203	1.00000000000000	0.99999999999937
0.99999999999703	0.99990149788562	1.00000000000000	0.99999999545394
0.00000000593309	0.98844658524616	1.00000000000000	0.99999681064565
0.00000000003840	0.89957454665058	0.99999999999988	0.99983896006215
0.00000000000003	0.54237185670110	0.99999999316379	0.99393007943547
0.00000000000000	0.06426938073642	0.99993817690380	0.94584519976471
0.00000000000000	0.02603228754605	0.99792215267787	0.20867650942988

the end, with a much larger search space, does GMRES(∞) catch up. The second run of GCROT also does better than its first run, indicating that it recycled a subspace useful for convergence. However, GCRO-DR performed initially somewhat better on the second run than the first, but the overall iteration count was approximately the same for both runs. This means that the subspace it recycled failed to improve convergence. For more analysis on the selection of good recycle spaces, see [23, 15, 24].

Table 4.3 shows the cosines of the principal angles between the subspace recycled by GCRO-DR and the invariant subspaces associated with the 10 and 21 eigenvalues of smallest magnitude, respectively, for the $c = 0$ and $c = 40$ cases. For the comparison with 10 eigenvectors, we see that the recycle space for the $c = 0$ case captures only 5 eigenvectors. We compare with the space spanned by 21 eigenvectors because it captures the entire recycled subspace for the $c = 0$ case. This means that GCRO-DR does not select the invariant subspace spanned by the eigenvectors for the 10 smallest eigenvalues, but rather selects some subspace of the space spanned by the 21 smallest. The table also shows that the approximation of an invariant subspace for the $c = 40$ case is nearly as good as for $c = 0$. However, this does not lead to similar convergence.

5. Conclusions and future work. We have presented an overview of Krylov subspace recycling for sequences of linear systems, where both the matrix and right-hand side change. Different choices for subspace selection and recycling have been shown, as well as methods implementing those choices. We propose the new solver GCRO-DR to implement Krylov subspace recycling of approximate invariant subspaces for Hermitian and non-Hermitian systems. We provide two important theoretical results complemented by a set of experiments to analyze the convergence of GCRO-DR for a typical application generating a long sequence of linear systems. GCROT has convergence behavior similar to GCRO-DR regarding recycling, but theoretical results for this method are a topic of future research. When solving a sequence of linear systems, methods employing Krylov subspace recycling frequently outperformed GMRES(∞) while recycling a subspace of only small dimension and minimizing over a small subspace. However, as particular examples in section 4.4 show, it is not completely clear how subspace selection affects convergence, so further theory is needed. Short recurrence methods for the Hermitian case have been developed [15].

Appendix. GCRO with deflated restarting (GCRO-DR).

Choose m, the maximum size of the subspace, and k, the desired number of approximate eigenvectors. Let ε_{tol} be the convergence tolerance. Choose an initial guess x_0. Compute $r_0 = b - Ax_0$, and set $i = 1$.
if $\widetilde{Y}_k$ is defined (from solving a previous linear system) **then**
  Let $[Q, R]$ be the reduced QR-factorization of $A\widetilde{Y}_k$.
  $C_k = Q$
  $U_k = \widetilde{Y}_k R^{-1}$
  $x_1 = x_0 + U_k C_k^H r_0$
  $r_1 = r_0 - C_k C_k^H r_0$
else
  $v_1 = r_0/\|r_0\|_2$
  $c = \|r_0\|_2 e_1$
  Perform m steps of GMRES, solving $\min \|c - \underline{H}_m y\|_2$ for y and generating V_{m+1} and $\underline{H}_m$.
  $x_1 = x_0 + V_m y$
  $r_1 = V_{m+1}(c - \underline{H}_m y)$
  Compute the k eigenvectors $\widetilde{z}_j$ of $(H_m + h_{m+1,m}^2 H_m^{-H} e_m e_m^H)\widetilde{z}_j = \widetilde{\theta}_j \widetilde{z}_j$ associated with the smallest magnitude eigenvalues $\widetilde{\theta}_j$ and store in P_k.
  $\widetilde{Y}_k = V_m P_k$
  Let $[Q, R]$ be the reduced QR-factorization of $\underline{H}_m P_k$.
  $C_k = V_{m+1} Q$
  $U_k = \widetilde{Y}_k R^{-1}$
end if
while $\|r_i\|_2 > tol$ **do**
  $i = i + 1$
  Perform $m - k$ Arnoldi steps with the linear operator $(I - C_k C_k^H)A$, letting $v_1 = r_{i-1}/\|r_{i-1}\|_2$ and generating V_{m-k+1}, $\underline{H}_{m-k}$, and B_{m-k}.
  Let D_k be a diagonal scaling matrix such that $\widetilde{U}_k = U_k D_k$, where the columns of $\widetilde{U}_k$ have unit norm.
  $\widehat{V}_m = [\widetilde{U}_k \;\; V_{m-k}]$
  $\widehat{W}_{m+1} = [C_k \;\; V_{m-k+1}]$
  $\underline{G}_m = \begin{bmatrix} D_k & B_{m-k} \\ 0 & \underline{H}_{m-k} \end{bmatrix}$
  Solve $\min \|\widehat{W}_{m+1}^H r_{i-1} - \underline{G}_m y\|_2$ for y.
  $x_i = x_{i-1} + \widehat{V}_m y$
  $r_i = r_{i-1} - \widehat{W}_{m+1} \underline{G}_m y$
  Compute the k eigenvectors $\widetilde{z}_i$ of $\underline{G}_m^H \underline{G}_m \widetilde{z}_i = \widetilde{\theta}_i \underline{G}_m^H \widehat{W}_{m+1}^H \widehat{V}_m \widetilde{z}_i$ associated with smallest magnitude eigenvalues $\widetilde{\theta}_i$ and store in P_k.
  $\widetilde{Y}_k = \widehat{V}_m P_k$
  Let $[Q, R]$ be the reduced QR-factorization of $\underline{G}_m P_k$.
  $C_k = \widehat{W}_{m+1} Q$
  $U_k = \widetilde{Y}_k R^{-1}$
end while
Let $\widetilde{Y}_k = U_k$ (for the next system).

Acknowledgments. We thank one of the anonymous referees and the editor for their helpful comments and for making some very useful suggestions regarding the experimental analysis.

REFERENCES

[1] C. Beattie, M. Embree, and J. Rossi, *Convergence of restarted Krylov subspaces to invariant subspaces*, SIAM J. Matrix Anal. Appl., 25 (2004), pp. 1074–1109.

[2] R. F. Boisvert, R. Pozo, K. Remington, R. F. Barrett, and J. J. Dongarra, *Matrix market: A Web resource for test matrix collections*, in Quality of Numerical Software: Assessment and Enhancement, Ronald F. Boisvert, ed., Chapman and Hall, London, 1997, pp. 125–136.

[3] T. F. Chan and M. K. Ng, *Galerkin projection methods for solving multiple linear systems*, SIAM J. Sci. Comput., 21 (1999), pp. 836–850.

[4] M. Creutz, *Quarks, Gluons, and Lattices*, Cambridge University Press, Cambridge, UK, 1986.

[5] E. de Sturler, *Inner-outer methods with deflation for linear systems with multiple right-hand sides*, in Householder Symposium XIII, Proceedings of the Householder Symposium on Numerical Algebra, 1996, pp. 193–196.

[6] E. de Sturler, *Nested Krylov methods based on GCR*, J. Comput. Appl. Math., 67 (1996), pp. 15–41.

[7] E. de Sturler, *Truncation strategies for optimal Krylov subspace methods*, SIAM J. Numer. Anal., 36 (1999), pp. 864–889.

[8] J. J. Dongarra, I. S. Duff, D. C. Sorensen, and H. A. van der Vorst, *Numerical Linear Algebra for High-Performance Computers*, Software Environ. Tools 7, SIAM, Philadelphia, PA, 1998.

[9] M. Eiermann, O. G. Ernst, and O. Schneider, *Analysis of acceleration strategies for restarted minimal residual methods*, J. Comput. Appl. Math., 123 (2000), pp. 261–292.

[10] C. Farhat and F.-X. Roux, *Implicit parallel processing in structural mechanics*, in Computational Mechanics Advances, J. Tinsley Oden, ed., North–Holland, Amsterdam, 1994, Vol. 2, pp. 1–124.

[11] P. F. Fischer, *Projection techniques for iterative solution of $A\underline{x} = \underline{b}$ with successive right-hand sides*, Comput. Methods Appl. Mech. Engrg., 163 (1998), pp. 193–204.

[12] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed., Johns Hopkins University Press, Baltimore, MD, 1996.

[13] A. Gullerud and R. H. Dodds, *MPI-based implementation of a PCG solver using an EBE architecture and preconditioner for implicit,* 3*-D finite element analyses*, Comput. & Structures, 79 (2001), pp. 553–575.

[14] D. D. Johnson, D. M. Nicholson, F. J. Pinski, B. L. Gyorffy, and G. M. Stocks, *Energy and pressure calculations for random substitutional alloys*, Phys. Rev. B, 41 (1990), pp. 9701–9716.

[15] M. E. Kilmer and E. de Sturler, *Recycling subspace information for diffuse optical tomography*, SIAM J. Sci. Comput., 27 (2006), pp. 2140–2166.

[16] W. Kohn and N. Rostoker, *Solution of the Schrödinger equation in periodic lattices with an application to metallic lithium*, Phys. Rev., 94 (1954), pp. 1111–1120.

[17] J. Korringa, *On the calculation of the energy of a Bloch wave in a metal*, Phys., 8 (1947), pp. 392–400.

[18] G. Mackey, *Reusing Krylov Subspaces for Sequences of Linear Systems*, Master's thesis, Department of Computer Science, University of Illinois at Urbana-Champaign, 2003.

[19] B. Medeke, *Set QCD: Quantum Chromodynamics*, description of matrix set on NIST Matrix Market, http://math.nist.gov/MatrixMarket.

[20] R. B. Morgan, *Implicitly restarted GMRES and Arnoldi methods for nonsymmetric systems of equations*, SIAM J. Matrix Anal. Appl., 21 (2000), pp. 1112–1135.

[21] R. B. Morgan, *GMRES with deflated restarting*, SIAM J. Sci. Comput., 24 (2002), pp. 20–37.

[22] D. P. O'Leary, *The block conjugate gradient algorithm and related methods*, Linear Algebra Appl., 29 (1980), pp. 293–322.

[23] M. L. Parks, *The Iterative Solution of a Sequence of Linear Systems Arising from Nonlinear Finite Element Analysis*, Ph.D. thesis, Department of Computer Science, University of Illinois at Urbana-Champaign, 2005.

[24] M. L. Parks and E. de Sturler, *Analysis of Krylov subspace recycling for sequences of linear systems*, 2006, to appear.

[25] C. Rey and F. Risler, *A Rayleigh-Ritz preconditioner for the iterative solution to large-scale nonlinear problems*, Numer. Algorithms, 17 (1998), pp. 279–311.

[26] F. Risler and C. Rey, *Iterative accelerating algorithms with Krylov subspaces for the solution to large-scale nonlinear problems*, Numer. Algorithms, 23 (2000), pp. 1–30.
[27] Y. Saad, *Iterative Methods for Sparse Linear Systems*, 2nd ed., SIAM, Philadelphia, PA, 2003.
[28] Y. Saad and M. H. Schultz, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 856–869.
[29] Y. Saad, M. Yeung, J. Erhel, and F. Guyomarc'h, *A deflated version of the conjugate gradient algorithm*, SIAM J. Sci. Comput., 21 (2000), pp. 1909–1926.
[30] V. Simoncini and E. Gallopoulos, *An iterative method for nonsymmetric systems with multiple right-hand sides*, SIAM J. Sci. Comput., 16 (1995), pp. 917–933.
[31] V. Simoncini and D. B. Szyld, *On the occurrence of superlinear convergence of exact and inexact Krylov subspace methods*, SIAM Rev., 47 (2005), pp. 247–272.
[32] A. V. Smirnov and D. D. Johnson, *Accuracy and limitations of localized Green's function methods for materials science applications*, Phys. Rev. B, 64 (2001), paper 235129.
[33] A. Stathopoulos, Y. Saad, and K. Wu, *Dynamic thick restarting of the Davidson, and the implicitly restarted Arnoldi methods*, SIAM J. Sci. Comput., 19 (1998), pp. 227–245.
[34] G. W. Stewart, *Matrix Algorithms. Volume* II: *Eigensystems*, SIAM, Philadelphia, PA, 2001.
[35] J. van den Eshof, A. Frommer, Th. Lippert, K. Schilling, and H. A. van der Vorst, *Numerical methods for the QCD overlap operator:* I *Sign function and error bounds*, Comput. Phys. Comm., 146 (2002), pp. 203–224.
[36] H. A. van der Vorst and C. Vuik, *The superlinear convergence behaviour of GMRES*, J. Comput. Appl. Math., 48 (1993), pp. 327–341.
[37] B. Vital, *Etude de quelques méthodes de résolution de problèmes linéaires de grande taille sur multiprocessor*, Ph.D. thesis, Université de Rennes I, Rennes, France, 1990.
[38] K. Wu and H. Simon, *Thick-restart Lanczos method for large symmetric eigenvalue problems*, SIAM J. Matrix Anal. Appl., 22 (2000), pp. 602–616.
[39] U. Meier Yang and K. A. Gallivan, *A new family of block methods*, Appl. Numer. Math., 30 (1999), pp. 155–173.
[40] R. Zeller, P. H. Dederichs, B. Ujfalussy, L. Szunyogh, and P. Weinberger, *Theory and convergence properties of the screened Korringa-Kohn-Rostoker method*, Phys. Rev. B, 52 (1995), pp. 8807–8812.

SIAM J. SCI. COMPUT.
Vol. 28, No. 5, pp. 1675–1693
© 2006 Society for Industrial and Applied Mathematics

ON MASS-CONSERVING LEAST-SQUARES METHODS*

J. J. HEYS†, E. LEE‡, T. A. MANTEUFFEL‡, AND S. F. MCCORMICK‡

Abstract. Least-squares variational methods have several practical and theoretical advantages for solving elliptic partial differential equations, including symmetric positive definite discrete operators and a sharp error measure. One of the potential drawbacks, especially in three dimensions, is that mass conservation is achieved only in a least-squares sense, and underresolved solutions are especially vulnerable to poor conservation. For the stationary Navier–Stokes equations, which are typically rewritten as a larger system of first-order equations, the loss of mass in the approximate solution is strongly dependent upon the boundary conditions used. A new first-order system reformulation of the Navier–Stokes equations is presented that admits a wider range of mass-conserving boundary conditions. This new formulation is shown to provide both excellent mass conservation and excellent algebraic multigrid performance for three different problems, a square channel, backward-facing step, and branching tubes with two generations of bifurcations.

Key words. Navier–Stokes, conservation, finite elements, least squares, multigrid

AMS subject classifications. 76D05, 65F10, 65N55

DOI. 10.1137/050640928

1. Introduction.

Least-squares variational principles have been repeatedly recognized as a new tool to solve elliptic partial differential equations [6]. For the Stokes and Navier–Stokes equations in particular, a number of advantages are present [3, 4, 24], including the following:

- the weak formulations are associated with a minimization problem, resulting in several computational advantages;
- the finite element spaces for each variable may be chosen independently because the inf-sup (or LBB) stability condition does not apply; and
- the functional provides a sharp, local error measure at no additional computational cost.

However, these benefits do not come without some drawbacks. In our experience, the following two are the most significant:

- when popular C^0 finite elements are used, the number of dependent variables typically increases over the original formulation (by a factor of 2 to 3 in many cases); and
- a lack of conservation of mass resulting in "ugly" solutions.

The computational advantages associated with least-squares finite element methods (LSFEM) are usually sufficient to overcome the first drawback, especially as problems become larger due to the scalability of multigrid solvers [9], which are often used in

*Received by the editors September 22, 2005; accepted for publication (in revised form) May 31, 2006; published electronically October 6, 2006. This work was sponsored by the Department of Energy under grants DE-FC02-01ER25479 and DE-FG03-99ER25217, Lawrence Livermore National Laboratory under contract B541045, Sandia National Laboratory under contract 15268, and the National Science Foundation under grants DMS-0410318. Computer time was provided by NSF ARI grant CDA-9601817, NSF MRI grant CNS-0420873, NASA AIST grant NAG2-1646, DOE SciDAC grant DE-FG02-04ER63870, NSF sponsorship of the National Center for Atmospheric Research, and a grant from the IBM Shared University Research (SUR) program.

http://www.siam.org/journals/sisc/28-5/64092.html

†Chemical and Materials Engineering, Arizona State University, Tempe, AZ 85287 (heys@asu.edu).

‡Department of Applied Mathematics, University of Colorado at Boulder, Boulder, CO 80309 (eunjung@colorado.edu, tmanteuf@colorado.edu, steven@boulder.colorado.edu).

conjunction with least-squares methods [12]. Further, in many cases, the additional variables are physically important quantities that must be calculated at some point, regardless of the discretization method being used. It is the second drawback, the lack of conservation of mass, that is the focus of this paper.

Mass conservation between any inflow surface, Γ_i, and an outflow surface, Γ_o, which are possibly connect with impermeable boundaries, is typically measured in terms of the fractional change of mass flow, defined as

$$\frac{\int_{\Gamma_i}(\mathbf{n}\cdot\mathbf{v})d\Gamma_i - \int_{\Gamma_o}(\mathbf{n}\cdot\mathbf{v})d\Gamma_o}{\int_{\Gamma_i}(\mathbf{n}\cdot\mathbf{v})d\Gamma_i},$$

where $\mathbf{v}$ is the velocity of an incompressible fluid. The mass conservation can be measured either globally over the entire domain or locally over smaller subdomains. If the velocity is completely specified over all boundaries, it must be set to satisfy the global mass-conservation constraint, and only local mass conservation will not be exactly satisfied. Published computational results [16] and our own experience show that when global mass conservation is enforced through boundary conditions, LSFEM give approximate solutions with acceptable local mass conservation. However, a common flow problem involves the velocity being completely specified only along the inflow and wall regions but not completely specified along the outflow regions. In this case, the error in global mass conservation is approximately the accumulation of errors in local mass conservation between the inlet and outlet. It is these types of flow problems, where the outlet velocity is not fully specified, that are the subject of this paper. The goal is to improve global mass conservation while maintaining the strengths of LSFEM, specifically the multigrid convergence rates. This is in contrast to many of the existing methods to improve mass conservation, which result in slower convergence rates for standard multigrid.

Mass conservation in Galerkin formulations of the Navier–Stokes equations is enforced by the divergence-free constraint on the velocity field, which essentially uses pressure as a Lagrange multiplier. With least-squares formulations, the objective is to minimize, in a least-squares sense, the value of a functional. For example, the stationary Navier–Stokes equations in dimensionless form are

$$-Re\,(\mathbf{v}\cdot\nabla\mathbf{v}) - \nabla p + \Delta\mathbf{v} = 0 \quad \text{in } \Omega, \tag{1.1}$$

$$\nabla\cdot\mathbf{v} = 0 \quad \text{in } \Omega, \tag{1.2}$$

where p is the pressure normalized by viscosity, Re is the Reynolds number, and $\mathbf{v} = (v_x, v_y, v_z)$ is the dimensionless velocity. Equations (1.1) and (1.2) may be combined in a least-squares sense into a single functional given by

$$G(\mathbf{v},p) := \alpha\| - Re\,(\mathbf{v}\cdot\nabla\mathbf{v}) - \nabla p + \Delta\mathbf{v}\|^2_{0,\Omega} + \beta\|\nabla\cdot\mathbf{v}\|^2_{0,\Omega}\,. \tag{1.3}$$

In practice, $G(\mathbf{v},p)$ would never be minimized because it exhibits behavior similar to a biharmonic, but it is shown to illustrate that there is always a balance between equations. The first term in G represents conservation of momentum, and the second represents conservation of mass. One of the potential strengths of least-squares methods is that, by adjusting the values of constants α and β, the error can be distributed between mass and momentum conservation in a controlled manner. However, changing the values of α and β almost always affects solver performance. For example, making β larger (i.e., less error in conservation of mass) can result in faster or slower multigrid convergence, depending on the Reynolds number, Re, and the actual

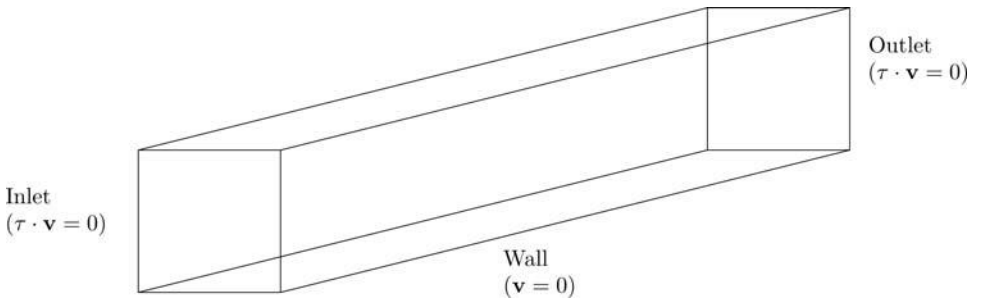


FIG. 1.1. *The test problem domain consisting of a brick with an aspect ratio of* 4:1:1.

functional used (G is not a computationally practical functional and is rarely used). The effects of weighting the functional are demonstrated later in this paper, but, even though this can be beneficial in some situations, it should not be considered a universal solution [8].

The focus in this paper is on stationary or steady-state Navier–Stokes equations, although some general statements can be made about the extension to nonstationary Navier–Stokes using implicit time stepping. The additional terms in the functional due to the discrete approximation of the time derivative makes the diagonal of the operator larger and, hence, it is usually more efficiently solved by a multigrid algorithm. Further acceleration is realized by the fact that the previous time step solution provides a good initial guess to the new time step. As a result, the methods proposed in this paper are also good methods for solving the nonstationary problem if implicit time stepping is desired.

1.1. Model problem and loss of mass conservation. We introduce a simple test problem to illustrate the mass-conservation issue associated with existing least-squares methods. The test problem was chosen to be as simple as possible while still capturing the key mass-conservation issues. Initially, the Reynolds number is set to zero because the important mass conservation issues exist in Stokes flow as well as laminar Navier–Stokes flow. Later, results using nonzero Reynolds numbers are given. While it is possible to demonstrate some of the mass-conservation issues in two dimensions, Pontaza and Reddy [24] recently showed that using higher-order finite elements largely alleviates the problems in two dimensions. Our own experiences agree with this finding, but, as we show, moving to higher-order elements is not as effective in addressing the mass-conservation issues in three dimensions. Therefore, the domain for the test problem is the simple "brick" shown in Figure 1.1 with an aspect ratio of 4:1:1. The long sides of the brick are walls upon which the no-slip boundary condition is imposed. Along the inlet and outlet, the tangential velocity is set to zero, and an additional boundary condition to be chosen later must also be set.

The functional, G, given by (1.3) has a higher regularity requirement and cannot be used with practical C^0 finite element spaces. Typically, the governing equations are transformed into an equivalent system of first-order equations. A number of different methods exist for generating a first-order system for the Navier–Stokes equations [3, 4]. One approach is based on defining a matrix of new variables, $\mathbf{V}$, equal to the gradient of the velocity vector. The first-order system for Navier–Stokes equations is then written as

$$\begin{aligned} &\mathbf{V} - \nabla \mathbf{v} = 0 && \text{in } \Omega, \\ &-Re(\mathbf{v} \cdot \mathbf{V}) - \nabla p + \nabla \cdot \mathbf{V} = 0 && \text{in } \Omega, \\ &\nabla \cdot \mathbf{v} = 0 && \text{in } \Omega. \end{aligned} \tag{1.4}$$

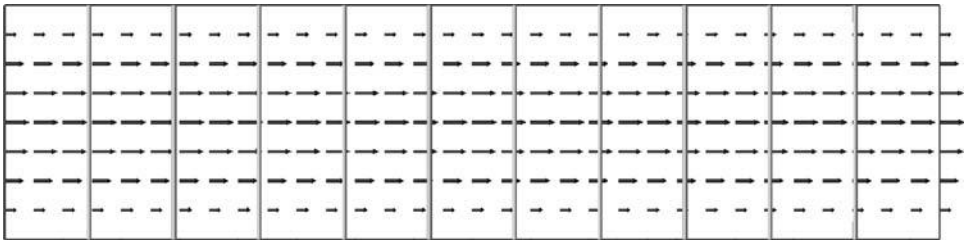


FIG. 1.2. *Cross-section of flow through the brick domain using the velocity-flux formulation of the Navier–Stokes equations. The pressure is set to* 1 *at the inlet and* 0 *at the outlet, and the contour bars represent evenly spaced pressure lines.*

Because $\mathbf{V}$ is the gradient of $\mathbf{v}$ and we are solving a minimization problem, we can augment the first-order system of equations by the following consistent equations:

$$\begin{aligned} \nabla(\mathrm{tr}(\mathbf{V})) &= 0 \quad \text{in } \Omega, \\ \nabla \times \mathbf{V} &= 0 \quad \text{in } \Omega, \end{aligned} \tag{1.5}$$

where $\mathrm{tr}(\mathbf{V}) = V_{11} + V_{22} + V_{33}$. This first-order formulation of the Navier–Stokes equations is often called the "velocity-flux" formulation, and it has a number of theoretical and practical advantages, including optimal error estimates in the H^1 norm for each variable and optimal multigrid convergence estimates [7, 4]. The corresponding functional for the velocity-flux formulation is

$$\begin{aligned} G_{v.f.}(\mathbf{V},\mathbf{v},p) := \|\mathbf{V} - \nabla\mathbf{v}\|^2_{0,\Omega} + \| - Re\,(\mathbf{v}\cdot\mathbf{V}) - \nabla p + \nabla\cdot\mathbf{V}\|^2_{0,\Omega} \\ + \|\nabla\cdot\mathbf{v}\|^2_{0,\Omega} + \|\nabla(\mathrm{tr}(\mathbf{V}))\|^2_{0,\Omega} + \|\nabla\times\mathbf{V}\|^2_{0,\Omega}. \end{aligned} \tag{1.6}$$

To minimize the functional $G_{v.f.}$ over the test domain shown in Figure 1.1, additional boundary conditions must be defined along the inlet and outlet. A common boundary condition is to set the normal stress on the normal face (i.e., the natural boundary condition) equal to a constant. For this domain with no tangential velocity at the inlet and outlet, setting the normal stress to a constant is equivalent to setting the pressure to a constant. The results of setting the inlet pressure (i.e., the normal stress) to 1 along the inlet and 0 along the outlet are shown in Figure 1.2. To generate this result, a hexahedral (cubic) mesh of $32 \times 8 \times 8$ elements was used ($h = 1/8$), the Reynolds number was set to zero (i.e., Stokes flow), and a trilinear finite element basis was used. Even though Figure 1.2 only shows the central cross-section of the brick, it is clear that there is good mass conservation based on a visual inspection. In fact, integrating the velocity field along the inlet and outlet (or any other cross-section orthogonal to the flow) shows perfect mass conservation up to 5 digits. The value of the functional at the approximate solution is 4.0×10^{-3}. If we look at the size of different terms in the functional, the divergence term, $\nabla\cdot\mathbf{v}$, is very small at 5.8×10^{-9}, and the error is largely in the momentum term, $-Re\,(\mathbf{v}\cdot\mathbf{V}) - \nabla p + \nabla\cdot\mathbf{V}$, with a value of 1.6×10^{-3}.

At this point, the least-squares formulation appears to provide excellent mass conservation. However, a change in the inlet boundary condition can have a dramatic effect. For example, the result of replacing the natural inlet condition, $p = 1$, with an essential one, in which the normal velocity is set to $v_x = y(1-y)z(1-z)$, is shown in Figure 1.3. It is important to note that the inlet condition is not set to the steady flow profile, but it is a valid boundary condition. The total value of the functional

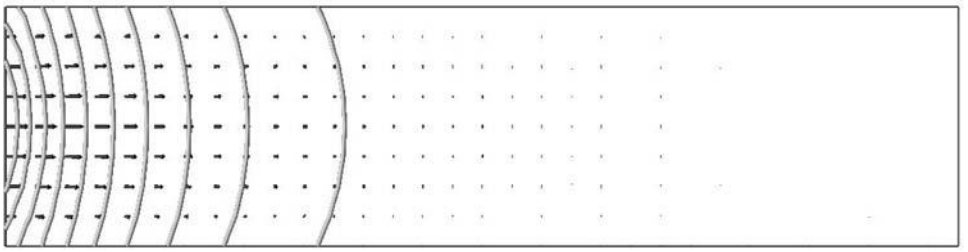


FIG. 1.3. *Cross-section of flow through the brick domain using the velocity-flux formulation of the Navier–Stokes equations. The normal velocity is set to* $v_x = y(1-y)z(1-z)$ *at the inlet and the pressure to* 0 *at the outlet, and the contour bars represent evenly spaced pressure lines.*

at the minimum is 6.0×10^{-3}, and the divergence term in the functional (5×10^{-4}), representing conservation of mass, is now nearly the same size as the momentum term (1×10^{-3}). The total flow rate into the domain is 2.7×10^{-2} and the outflow rate is 5.0×10^{-5}, a 99.8% loss of mass. The key to understanding this mass loss is to look at the pressure, which should be decreasing nearly linearly except for some small entrance effects near the inlet. Instead, the functional is a minimum when both the pressure and the flow rate are approximately zero for two-thirds of the domain lying downstream. Clearly, the functional is approximately zero for the downstream portion of the domain, which helps keep the functional small. The larger error at the entrance is not sufficient to make the overall functional value large. Remember, this method finds the solution that [illegible] the functional over all possible solutions in the finite element space. In other words, if we do not like the solution, we asked the method to minimize a functional that does not capture our requirements for the solution.

1.2. Possible remedies. Several approaches can be taken to "fix" the conservation of mass problem illustrated in the previous example (i.e., Figure 1.3), including the following:

- reducing the mesh size, h;
- weighting the velocity-divergence term in the functional;
- using higher-order elements or some other finite element space (e.g., [10]); and
- using a different first-order system of equations that is still equivalent to the Navier–Stokes equations.

Other, more complex remedies can be used, including norms other than L^2 norms or using a FOSLL* approach [11], but only the approaches listed above are explored here. Below, each of these options is individually explored using the previous test problem. In most cases, mass conservation is improved at the cost of multigrid performance, making the remedy unacceptable.

Reducing the mesh size has the advantage that it should improve mass conservation and not make the problem worse. As the mesh size approaches zero, the approximate solution approaches the true solution at the expected rate, which depends on the finite element basis. However, because conservation of mass is currently far from being acceptable, significant refinement is required before the approximation becomes acceptable. For example, if the previous mesh ($32 \times 8 \times 8$) is refined to ($64 \times 16 \times 16$), the loss of mass is only reduced from 99.8% to 98.8%.

As mentioned previously, weighting the divergence of velocity term in the functional can also improve conservation [16]. If the divergence term in the velocity-flux

formulation (i.e., $G_{v.f.}$) is weighted by 100, the mass loss is reduced from 99.8% to 0.7%. However, the multigrid convergence factor, defined as

$$\rho \approx \sqrt[m]{\frac{\|res^{(m)}\|_2}{\|res^{(0)}\|_2}}, \tag{1.7}$$

where $res^{(m)}$ is the residual (or defect) after the mth V-cycle, is increased from 0.74, with a weight of 1.0, to 0.87, with a weight of 100.0. To put that change into perspective, reducing the residual norm by a factor of 1.0×10^{-8} requires 62 V-cycles when the convergence factor is 0.74, and 132 V-cycles when the convergence factor is 0.87, a doubling of the total computational cost. Further, in our experience, larger weights are required as the domain becomes more complex, especially domains with high aspect ratios. Multigrid methods that perform well on this class of problems exist, but require special elements and special relaxation techniques (cf. [2]). A similar idea to weighting the divergence term is to enforce it through a Lagrange multiplier strategy [13]. In this case, the operator is still symmetric, but the now indefinite operator is not effectively solved using a standard multigrid method.

It has been shown by both Pontaza and Reddy [24] and Deang and Gunzburger [16] that higher-order (or spectral) elements can produce very accurate results with least-squares formulations. This has been shown extensively in two dimensions, but it has not been demonstrated in three dimensions. Further, the computational costs associated with least-squares formulations in three dimensions with high-order elements can become prohibitive. For example, in the velocity-flux formulation there are 13 variables per node in three dimensions. Taking the tensor product of 3 eighth-order nodal basis functions results in an element with 729 nodes. The cost of storing a single element stiffness matrix with double precision variables approaches 1 gigabyte. If eighth-order basis functions are too expensive to store, one may ask what is the benefit of simply refining from trilinear elements to triquadratic elements. For the test problem above with the essential condition on velocity along the inlet, the mass loss is decreased from 99.8% to 64%, a significant improvement. However, even the improved mass conservation is still not acceptable, and it came at a increased computational cost (e.g., if the one-dimensional polynomial order is p, then computational costs scale as $O(p^6)$ if exact integration is used).

Related to the idea of using alternative finite element spaces is the idea of using reduced integration methods [21, 26]. Often, the use of reduced integration results in a collocation solution [6, 24]. The net effect of using reduced integration on most problems, including the test problem, is excellent conservation of mass, loss of the functional as an error measure because it is zero at the minimum, and poor multigrid performance. For example, when using reduced integration on the brick test problem, mass was conserved up to at least 5 digits of accuracy, but thousands of algebraic multigrid V-cycles were required for marginal convergence. In this case, a direct solver would have been much more effective, but our desire is to find a solution to the mass conservation problem that maintains the optimal computational scalability of standard multigrid.

The final approach we mention is to change the form of the functional (i.e., to use a different first-order system of equations). This approach is attractive due to our previous observation that if we do not like the approximate solution that minimizes the functional, then we are using the wrong functional. The most popular first-order system that is equivalent to the Navier–Stokes equations is the vorticity system, given

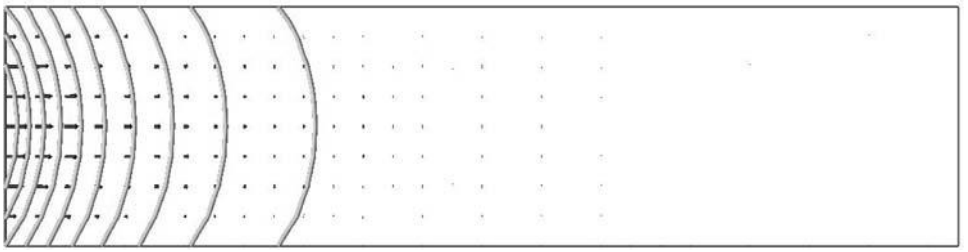


FIG. 1.4. *Cross-section of flow through the brick domain using the vorticity formulation of the Navier–Stokes equations. The normal velocity is set to* $v_x = y(1-y)z(1-z)$ *at the inlet and the pressure to* 0 *at the outlet, and the contour bars represent evenly spaced pressure lines.*

by

$$\begin{aligned} &\omega + \nabla \times \mathbf{v} = 0 && \text{in } \Omega, \\ &-Re(\mathbf{v}\cdot\nabla)\mathbf{v} - \nabla p + \nabla \times \omega = 0 && \text{in } \Omega, \\ &\nabla \cdot \mathbf{v} = 0 && \text{in } \Omega. \end{aligned} \tag{1.8}$$

Because ω, the vorticity, is the negative curl of $\mathbf{v}$ and we are solving a minimization problem, we can augment this first-order system by the following consistent equation:

$$\nabla \cdot \omega = 0 \quad \text{in } \Omega. \tag{1.9}$$

The corresponding functional for the vorticity formulation is

$$\begin{aligned} G_\omega(\omega, \mathbf{v}, p) := \|\omega + \nabla \times \mathbf{v}\|^2_{0,\Omega} + \| - Re\,(\mathbf{v}\cdot\nabla)\,\mathbf{v} - \nabla\, p + \nabla \times \omega\|^2_{0,\Omega} \\ + \|\nabla \cdot \mathbf{v}\|^2_{0,\Omega} + \|\nabla \cdot \omega\|^2_{0,\Omega}. \end{aligned} \tag{1.10}$$

One advantage of this formulation is that it contains only 7 dependent variables in three dimensions, almost half the number of the velocity-flux formulation; however, since it is not a fully H^1-coercive system [6], optimal multigrid convergence cannot be proven. Nevertheless, since the focus here is on mass conservation, we examine the effects of this formulation on mass loss through the brick domain using the same boundary conditions as Figure 1.2 (i.e., $v_x = y(1-y)z(1-z)$ along the inlet). The results are summarized by the cross-section shown in Figure 1.4, and they demonstrate extensive mass loss similar to the velocity-flux formulation. In this case, the mass loss is 99.9%, even worse than the 99.8% previously observed, and the multigrid convergence factor was significantly worse, requiring approximately twice the number of V-cycles. The final value of the functional, G_ω, was 2.7×10^{-3}, but most of the error was in the divergence-of-velocity term (1.1×10^{-3}). Clearly, the two first-order systems for the stationary Navier–Stokes equations commonly found in the literature do not provide acceptable mass conservation for this set of boundary conditions in three dimensions.

In summary, even though some of the remedies tested improved mass conservation, they did so at the cost of multigrid performance. Since the goal is to improve mass conservation without an appreciable decline in multigrid performance, none of these remedies is acceptable. The only solution shown to improve mass conservation without a decline in multigrid performance is setting boundary conditions on pressure for both the inlet and outlet. This observation first provoked an effort to find a new formulation that allows pressure boundary conditions to be set in a more flexible way.

During this effort, we developed a formulation that not only allowed new boundary conditions to be set that incorporated pressure, but the new formulation also achieves better computational performance (i.e., multigrid convergence rates) at the same. In this way, it satisfies our goal of improving mass conservation while maintaining the strengths of the LSFEM methodology.

2. New formulation. Before defining the new formulation, we need the identity

$$\nabla \times (\mathbf{v} \cdot \nabla)\mathbf{v} = (\mathbf{v} \cdot \nabla)\omega - (\omega \cdot \nabla)\mathbf{v}, \tag{2.1}$$

and we define a new variable $\mathbf{r}$ as

$$\mathbf{r} = \nabla p + Re(\mathbf{v} \cdot \nabla)\mathbf{v} \ . \tag{2.2}$$

The new variable is equal to the pressure gradient plus the convective term. We now introduce the following first-order system to replace the Navier–Stokes equations:

$$\begin{array}{ll} \nabla \times \mathbf{v} + \omega = 0 & \text{in } \Omega, \\ \nabla \cdot \mathbf{v} = 0 & \text{in } \Omega, \\ \nabla \times \omega - \mathbf{r} = 0 & \text{in } \Omega, \\ \nabla \cdot \omega = 0 & \text{in } \Omega, \\ \nabla \times \mathbf{r} - Re(\mathbf{v} \cdot \nabla\omega - \omega \cdot \nabla\mathbf{v}) = 0 & \text{in } \Omega, \\ \nabla \cdot \mathbf{r} = 0 & \text{in } \Omega. \end{array} \tag{2.3}$$

The first two equations in system (2.3) are the same as those previously used in the vorticity formulation. The third equation is the momentum balance, rewritten in terms of $\mathbf{r}$ and the curl of vorticity. The fourth equation is derived by taking the divergence of the first equation. The fifth equation comes from taking the curl of (2.2), and the final equation is the result of taking the divergence of the momentum balance. The corresponding functional for the new formulation is

$$\begin{aligned} G_{new}(\omega, \mathbf{v}, \mathbf{r}) := & \|\omega + \nabla \times \mathbf{v}\|^2_{0,\Omega} + \|\nabla \cdot \mathbf{v}\|^2_{0,\Omega} \\ & + \|\mathbf{r} + \nabla \times \omega\|^2_{0,\Omega} + \|\nabla \cdot \omega\|^2_{0,\Omega} \\ & + \|\nabla \times \mathbf{r} - Re(\mathbf{v} \cdot \nabla\omega - \omega \cdot \nabla\mathbf{v})\|^2_{0,\Omega} + \|\nabla \cdot \mathbf{r}\|^2_{0,\Omega} \ . \end{aligned} \tag{2.4}$$

To further analyze this system of equations, it is useful to rewrite the system using slack variables, α, β, and δ, and moving subprincipal terms to the right-hand side, giving:

$$\begin{array}{cccccccc} \nabla \times \mathbf{v} & -\nabla\alpha & & & & & = & -\omega, \\ \nabla \cdot \mathbf{v} & & & & & & = & 0, \\ & & \nabla \times \omega & -\nabla\beta & & & = & \mathbf{r}, \\ & & \nabla \cdot \omega & & & & = & 0, \\ & & & & \nabla \times \mathbf{r} & -\nabla\delta & = & Re(\mathbf{v} \cdot \nabla\omega - \omega \cdot \nabla\mathbf{v}), \\ & & & & \nabla \cdot \mathbf{r} & & = & 0. \end{array} \tag{2.5}$$

Here, the slack variables are included to help analyze the new formulation, but the boundary conditions are set so that the slack variables, α, β, and δ, are equal to zero everywhere and, hence, are not necessary in the computation. The system of equations given by (2.5) contain a block diagonal operator with three blocks. Each block consists of a curl and divergence of a variable we are interested in approximating. With the proper boundary conditions on each block (specifically, one condition on the

slack variable and one on the original variable in the normal direction or, alternatively, conditions on the original variable in the two tangential directions), it is easy to show that the functional associated with (2.5) and $Re = 0$ is continuous and coercive in the $(H(\text{div}) \cap H(\text{curl})) \times H^1$ norm for each block of variables. The results are obtained by using a standard compactness argument. For the lowest block, containing the curl and divergence of $\mathbf{r}$, boundary conditions are typically set on $\mathbf{n} \cdot \mathbf{r}$, where $\mathbf{n}$ is the normal vector, and on slack variable δ. For the central block, boundary conditions are set on the normal vorticity and on slack variable β. For the upper block, boundary conditions only need to be set on the normal velocity and α because the tangential velocity is related to the normal vorticity, but, in practice, boundary conditions are set on two or all three of the velocity components.

If the Reynolds number is set to zero, the system of equations (2.5) can be solved by inverting the individual blocks of the matrix. First, the bottom block is inverted to calculate $\mathbf{r}$, then $\mathbf{r}$ is used on the right-hand side of the central block, which is inverted to calculate ω. Finally, ω is used to calculate the velocity, $\mathbf{v}$. In this way, storage costs can be reduced by requiring only that the blocks of the operator be stored. Further, these div/curl blocks are solved very efficiently by multigrid solvers. If the Reynolds number is not equal to zero, the block strategy can still be used, but an initial guess for $\mathbf{v}$ and ω is required followed by multiple iterations. In practice, we assemble the full operator because it is efficiently handled by multigrid and makes the implementation more straightforward.

The boundary conditions for the new formulation are similar to those typically used for the traditional vorticity formulation except for those on $\mathbf{r}$. An excellent description and classification of possible boundary conditions on $\mathbf{v}$ and ω can be found in the work of Bochev [3]. Fully specifying the velocity along all boundaries is sufficient to obtain a well-posed problem having a unique solution. However, the inclusion of boundary conditions on $\mathbf{n}\cdot\omega$ and $\mathbf{n}\cdot\mathbf{r}$ results in the operator being coercive in H^1 and optimal multigrid performance on the linear system. Before the boundary conditions are described in detail, it is important to note that all boundary conditions can be imposed either strongly on the finite element space or weakly by including the boundary conditions in the least-squares functional. This flexibility is useful since we want to strongly impose some boundary conditions, like the no-slip condition on velocity, while others are typically imposed weakly.

Along the walls or the no-slip boundaries, all velocity components are set to zero, the normal vorticity is set to zero, and $\mathbf{n} \cdot \mathbf{r}$ is set to zero (see Table 2.1). The last boundary condition, which is set on $\mathbf{r}$, can be explained by the equation

$$\mathbf{n} \cdot \mathbf{r} = \mathbf{n} \cdot \nabla p + \mathbf{n} \cdot (Re(\mathbf{v} \cdot \nabla)\mathbf{v}) = \mathbf{n} \cdot \Delta\mathbf{v}, \tag{2.6}$$

where $\Delta\mathbf{v} \equiv (\nabla \cdot \nabla)\mathbf{v}$. Therefore, setting $\mathbf{n} \cdot \mathbf{r}$ to zero along the wall can be viewed physically in two different ways. First, it is equivalent to setting the second derivative of the normal velocity in the normal direction to zero (i.e., the normal velocity must be zero near the wall as well as at the wall). Second, since the normal velocity is zero at the wall, it is equivalent to setting the normal derivative on pressure to zero, and this is consistent with the basic assumptions of Prandtl's boundary layer theory [28].

Inflow and outflow boundary conditions are a well-known difficulty in computational fluid dynamics [18]. For the new formulation, boundary conditions are typically set on two or three of the velocity components (see Table 2.1), although it is possible to set a condition on just the normal velocity. A condition should be set only on the normal vorticity because fully specifying the vorticity usually results in a problem

TABLE 2.1
Boundary conditions used for the the new formulation. τ_1 and τ_2 are the two tangential vectors along the inlet and c is an unknown constant that approximates $\frac{\partial^2 \mathbf{v}_n}{\partial \mathbf{n}^2}$.

Boundary	Condition	Enforcement
Wall	$\mathbf{v} = 0$	strong
	$\mathbf{n} \cdot \omega = 0$	weak
	$\mathbf{n} \cdot \mathbf{r} = 0$	weak
Inflow	$\mathbf{n} \cdot \mathbf{v} = g$	strong
	$\mathbf{n} \times \mathbf{v} = 0$	strong
	$\mathbf{n} \cdot \omega = 0$	weak
	$\mathbf{n} \cdot \mathbf{r} = \frac{\partial^2 \mathbf{v}_n}{\partial \tau_1^2} + \frac{\partial^2 \mathbf{v}_n}{\partial \tau_2^2} + c$	weak
Outflow	$\mathbf{n} \times \omega = 0$	strong
	$\mathbf{n} \cdot \omega = 0$	weak
	$\mathbf{n} \times \mathbf{r} = 0$	weak

that is not well posed [3]. The final boundary condition to be set is on $\mathbf{r}$. In the test problem, for example, a boundary condition setting the pressure equal to zero was used. An equivalent boundary condition here would be to set the value of $\mathbf{r}$ in the tangential direction(s) to zero, assuming that the tangential velocity is zero. If a steady flow boundary condition is desired, then $\mathbf{n} \cdot \mathbf{r}$ is set to $\mathbf{n} \cdot \Delta\mathbf{v}$, which is fully known since the second derivative in the normal direction is zero under steady flow conditions. The most difficult boundary condition to handle is when the velocity is fully specified, but the velocity profile is not equal to the steady profile, i.e., $\mathbf{n} \cdot \Delta\mathbf{v}$ is not fully known. Clearly, we know the second derivatives of the normal velocity in the tangential directions, but the second derivative in the normal direction of the normal velocity is not known and it is generally nonzero. One possible mechanism for solving for this unknown is to (1) assume the unknown is zero, (2) impose the boundary condition weakly, (3) solve the minimization problem, (4) check the value of the unknown quantity, and (5) repeat using the new value until convergence. This method has potentially slow convergence, and we have developed an alternative approach. Recalling that one of our primary goals is global mass conservation, we propose setting the second derivative of $\mathbf{n} \cdot \mathbf{v}$ in the normal direction equal to an unknown constant and solving for the constant using the constraint of global mass conservation.

Solving for the unknown constant in the $\mathbf{n} \cdot \mathbf{r}$ boundary condition can be accomplished easily if the Reynolds number is zero. In this case, two different guesses of the value of the constant are made, and the problem is solved with each guess. Unless one of the guesses is exact, the inflow rate will not exactly equal the outflow rate. However, based on the different mass conservation results, linear interpolation can be used to determine the correct value of the unknown constant, and a linear combination of the two solutions gives the final solution. For the case of the nonlinear Navier–Stokes equations, the same basic procedure of guessing different values of the constant may be used. In this case, however, more than two solutions may be required to get acceptably close to global conservation. This process often does not add any additional computational cost, in the case of a nonzero Reynolds number, because multiple iterations are already required by the nonlinear operator.

To solve the least-squares problem, the equations that are used in the functional (G_{new}) are first linearized so that the solution can be found using a Gauss–Newton approach. The value of the nonlinear functional is calculated after each Gauss–Newton step to ensure that the nonlinear functional is decreasing to a minimum. A line search in the direction of the Gauss–Newton step could be used to ensure that the nonlinear

functional decreases at each step. However, for all the problems that we tested, no line search was necessary. The functional for the linearized equations is minimized using standard techniques from the calculus of variations to obtain the weak form (e.g., [5, 3]). A finite element basis is then chosen so that the weak form generates a matrix problem. An alternative approach is to linearize the weak form instead of linearizing the original equations [14, 15], but we found better convergence by first linearizing the equations. All of the simulations presented in the results section utilized a trilinear finite element basis for all of the variables unless otherwise noted. The FOSLS formulation allows the solution spaces for the variables to be chosen independently, and there is no restrictive stability condition (i.e., inf-sup condition) to satisfy. As a result, both the pressure and velocity in the Navier–Stokes equations can be approximated with the same basis.

All simulations, including those in the introduction section, were performed using the ParaFOS code, written by the authors. The code imports hexahedral meshes from the Cubit mesh generation package (Sandia National Laboratory). The finite element meshes are then partitioned using the Metis graph partitioning library [23]. The software is designed to run on distributed memory clusters using the MPI library for communication. The linear matrix problem generated during each Gauss–Newton step is solved using the .,. library of solvers ([17], Center for Applied Scientific Computing, Lawrence Livermore National Laboratory). Specifically, the Boomer-AMG parallel algebraic multigrid (AMG) solver is used as a preconditioner for a conjugate gradient solver. Most of the solver parameters used were the default settings except for the coarse grid selection algorithm, which was set to the parallel modified independent set algorithm. This algorithm was chosen because it chooses coarse grids with fewer nodes, thus reducing storage requirements.

3. Results. The brick test problem with all components of the inlet velocity specified ($v_x = y(1-y)z(1-z)$, $v_y = 0$, $v_z = 0$) is complicated because the value of $\mathbf{n} \cdot \Delta\mathbf{v}$ is not completely known over the inlet. Specifically, the value of $\frac{\partial^2 v_x}{\partial x^2}$, which we refer to as C_1, is unknown. While this value may be a function of (y, z) along the inlet, we assume it is a constant and solve for it using the constraint of global mass conservation. Since the Reynolds number is also set to zero, this constant is equal to the deviation of the normal pressure gradient from the steady flow value. In other words, if the inlet velocity profile was the steady velocity profile, the pressure gradient would be a constant (i.e., a linear pressure drop), and the value of C_1 would be zero. The appropriateness of our assumption that C_1 is a constant and not a function of (x, y) is going to depend upon how close the specified inlet velocity profile is to the steady velocity profile. If the specified profile is close in some regions and deviates greatly in other regions, the assumption will not be appropriate. If the specified profile is close to the steady profile, the assumption is not a considerable source of error. A more detailed analysis of the error resulting from this assumption is the subject of a future paper. For now, we just show numerically that setting C_1 to a constant is acceptable for the chosen test problems.

We begin by setting $C_1 = 0$ and minimizing the functional, G_{new}, over the brick-shaped domain. Clearly, $C_1 \neq 0$ and the result is that the inflow rate is 2.7×10^{-2} and the outflow rate is 1.9×10^{-2}. It is well known from computational and experimental fluid dynamics that the pressure drop in the normal direction at the inlet must be larger than the steady-state pressure drop due to entrance effects. In other words, $C_1 > 0$ due to the extra pressure drop to rearrange the flow profile into the steady-state profile. For the second run, we set $C_1 = 1$ and minimize the functional again.

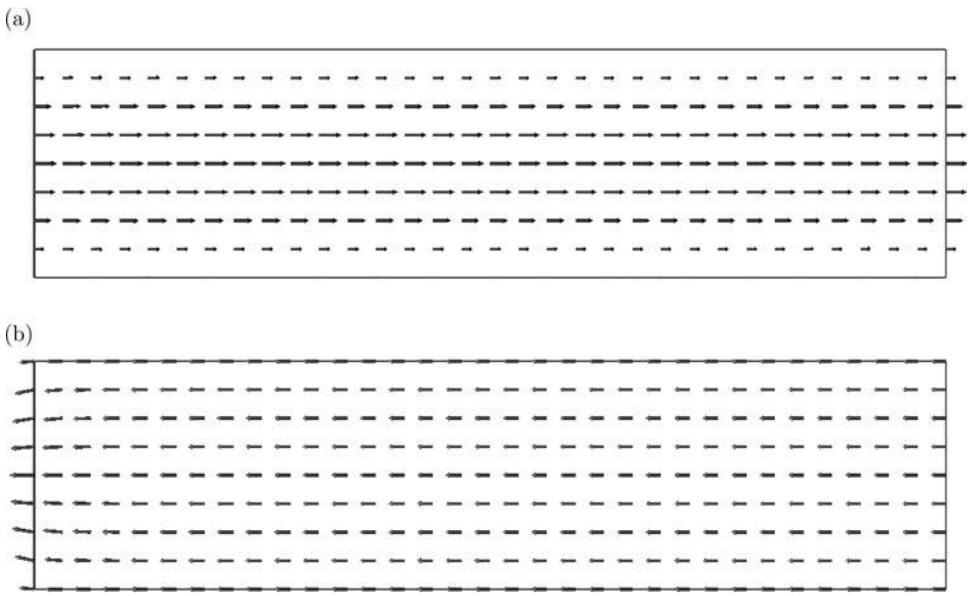


FIG. 3.1. *Central cross-section of flow through the brick domain using the new formulation of the Navier–Stokes equations. The normal velocity is set to* $v_x = y(1-y)z(1-z)$ *at the inlet and the pressure to* 0 *at the outlet. Velocity vectors are shown in* (a) *and the pressure gradient is shown in* (b).

The inflow rate remains unchanged at 2.7×10^{-2} (this essential boundary condition was strongly enforced), but now the outflow rate is 4.8×10^{-2}. Instead of losing mass, we are now "creating" mass. The true value for C_1 can be determined by linear interpolation to be 0.28 to give discrete mass conservation. To get the final solution shown in Figure 3.1, we can either use the appropriate linear combination of previous solutions or simply resolve the problem using $C_1 = 0.28$. Both the global and local conservation of mass using the final value of C_1 are excellent as mass is conserved exactly to the 5 digits calculated for any cross-section. Further, the average AMG convergence factor is also excellent at 0.3, indicating that only 15 V-cycles are required to reduce the residual by a factor of 1×10^{-8}. A comparison of the 3 different functionals, $G_{v.f.}$, G_ω, and G_{new}, is given in Table 3.1. The new formulation provides excellent global mass conservation and significantly better computational performance while maintaining computational scalability. It is not appropriate to compare the minimums of the different functionals, and they are only given for reference. Under steady flow conditions, the pressure drop in the normal direction is 1.0 and, now, for this inlet flow profile, the pressure drop at the center of the entrance is approximately 1.28. Figure 3.1(b) shows the pressure gradient throughout the domain. Note that it is constant at the outlet, but there are some entrance effects near the inlet.

Similar results can be achieved for problems with nonzero Reynolds number. Repeating the previous problem with $Re = 40$ results in $C_1 = 0.46$ being required for global mass conservation. In this case, the average AMG convergence factor deteriorates slightly to between 0.36 and 0.5, depending on the Gauss–Newton step, but it remains much improved over the factors of the other two formulations, which are between 0.9 and 0.95. As the Reynolds number becomes larger and larger, the entrance effects extend further downstream. For laminar flow into a round pipe, the

TABLE 3.1
Comparison of the different functionals for solving Stokes flow through the brick-shaped domain with the velocity fully specified along the inlet.

Method	Mesh	Mass loss	Functional	Avg. C.F.	CPU time
Velocity-flux	$32 \times 8 \times 8$	99.8%	6×10^{-3}	0.74	60 sec.
Vorticity	$32 \times 8 \times 8$	99.9%	2.7×10^{-3}	0.88	16 sec.
New	$16 \times 4 \times 4$	$< 0.01\%$	2.3×10^{-1}	0.25	1.6 sec.
New	$32 \times 8 \times 8$	$< 0.01\%$	9.6×10^{-2}	0.3	15 sec.
New	$64 \times 16 \times 16$	$< 0.01\%$	3.0×10^{-2}	0.36	150 sec.

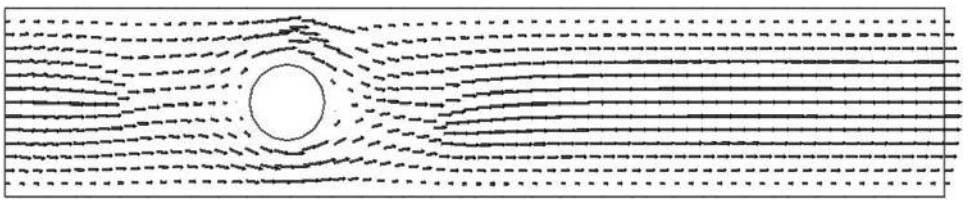


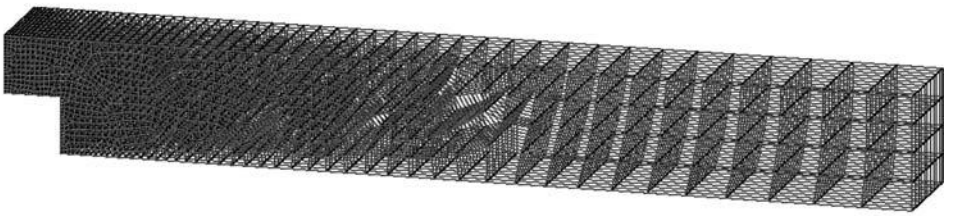
FIG. 3.2. *Central cross-section of flow through the brick domain* $(5 \times 1 \times 1)$ *with a cylindrical obstruction with a diameter of* 0.4. *The normal velocity is set to* $v_x = y(1-y)z(1-z)$ *at the inlet, and the nodal velocity vectors are shown.*

accepted correlation [27] is $L \approx 0.06 \cdot Re \cdot d$, where d is the diameter and L is the length of the entrance effects. The current assumption of steady flow at the outlet is probably appropriate since the inlet profile is close to the steady profile, but the assumption will break down at higher Reynolds numbers.

An obstruction can be included in this first test problem by removing a cylindrical portion of the domain. The results for $Re = 100$ and an obstruction with a diameter of 0.4 are shown in Figure 3.2. The boundary conditions along the walls of the obstruction are the same as those along the outer walls of the domain (i.e., $\mathbf{v} = 0$, $\mathbf{n} \cdot \omega = 0$, and $\mathbf{n} \cdot \mathbf{r} = 0$), and, in this case, a triquadratic finite element basis is used to more accurately capture the cylindrical geometry. Interestingly, to achieve global mass conservation C_1 is set to 0.20, a smaller value than the unobstructed case, but this is likely due to the more accurate triquadratic basis. The finite element mesh for this problem contains 1290 hexahedral elements and is unstructured, which contributes to some of the deterioration in the average AMG convergence factor to 0.74. For this problem, the domain was extended to a dimensionless length of 5 to ensure that the outlet profile is the steady profile.

The geometry and boundary conditions for the second test problem are somewhat similar to the first. In this case, however, the domain is longer, having an aspect ratio of 8:1:1, and there is a 0.5:0.5:1 notch removed from the lower half of the inlet. In other words, this is the classic flow over a backward-facing step in three dimensions. The inlet velocity is once again set to a tensor product of one-dimensional parabolas, $v_x = y(0.5-y)(0.5+z)(0.5-z), v_y = 0, v_z = 0$, and the tangential velocity is set to zero along the outlet. Along the inlet, $\mathbf{n} \cdot \mathbf{r} = -2(0.5+z)(0.5-z) - 2y(0.5-y) - C_2$, where C_2 is a constant that is set so that global conservation of mass is achieved, and $\tau \cdot \mathbf{r}$ is set to zero along the outlet. Finally, the no-slip boundary conditions along the wall are set the same as before, with the notable exception of the condition on $\mathbf{n} \cdot \mathbf{r} = \mathbf{n} \cdot \Delta\mathbf{v}$, which is set to zero near the reentrant corner of the step. This is because the physical assumption that allowed us to set this to zero along a smooth wall

(a)



(b)

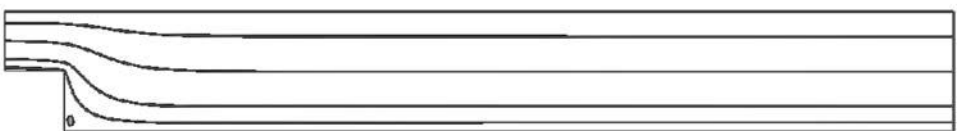


FIG. 3.3. *Central cross-section of flow through the backward-facing step domain using the new formulation of the Navier–Stokes equations at* $Re = 50$. *The finite element mesh are shown in* (a) *and streamlines are shown in* (b).

no longer holds. In this problem, there is a boundary layer separation at the corner, and there is an infinite number of normal vectors at the corner. Numerical experiments were performed on different methods for removing this boundary condition from the corner. First, the $\mathbf{n}\cdot\mathbf{r}$ condition, which is typically enforced weakly, can be completely removed from the functional along the two walls that contact at the corner. Second, the $\mathbf{n}\cdot\mathbf{r}$ term can be multiplied by a weight function that is zero at the corner and one at some radius away from the corner [19]. A typical weight function has the form $w(r) = (r/R)^n$, where r is the distance from the corner, R is the radius of the weight function (typically $O(0.1)$), and n is the polynomial order of the weight function (typically 3 or 4). Acceptable (and indistinguishable) results were obtained with either method as long as the radius of the weight function was sufficiently large.

The finite element mesh used for the backward-facing step geometry is shown in Figure 3.3(a). The mesh is unstructured and consists of 6510 elements. A triquadratic finite element basis was used to improve the accuracy of the method. An analysis of the accuracy of LSFEM for the two-dimensional backward-facing step problem can be found in Röhrle [25] and for the three-dimensional problem in Jiang et al. [22]. A Reynolds number of 50 was used to obtain the streamlines in a central cross-section of the approximate solution shown in Figure 3.3(b). At this low Reynolds number, only a small recirculation is found downstream of the step; this is consistent with the two-dimensional experimental results of Armaly et al. [1]. The global conservation of mass was exact, since it was enforced by setting C_2, and the behavior of the pressure field was also consistent with experimental predictions in that the pressure gradient in the flow direction was greater for the small channel, upstream from the step, than for the larger channel downstream from the step. The AMG convergence factor was 0.49 for a trilinear basis and 0.72 for a triquadratic basis. Both factors are acceptable considering the higher-order finite element basis, the partitioning of the problem among 16 processors, and absence of a boundary condition on $\mathbf{r}$ along the step.

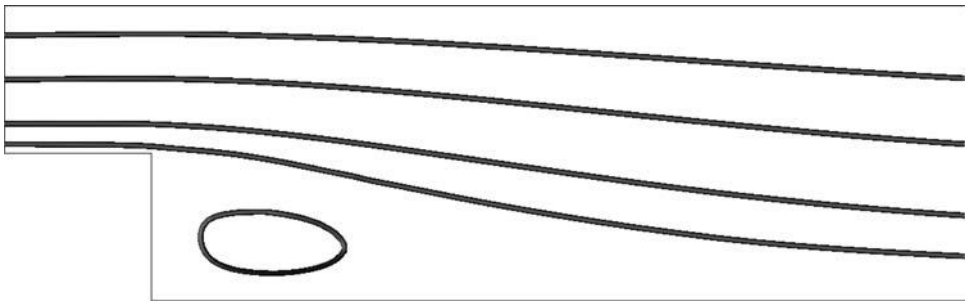


FIG. 3.4. *Central cross-section of flow through the backward-facing step domain using the functional $G_{new,b}$ at $Re = 200$. Only the upstream half of the total domain is shown for clarity.*

For greater values of Re, it is potentially helpful to rescale the momentum equation by dividing by Re, giving a new functional

$$
\begin{aligned}
G_{new,b}(\omega, \mathbf{v}, \mathbf{r}) := {} & \|\omega + \nabla \times \mathbf{v}\|_{0,\Omega}^2 + \|\nabla \cdot \mathbf{v}\|_{0,\Omega}^2 \\
& + \|\mathbf{r} + \frac{1}{Re}\nabla \times \omega\|_{0,\Omega}^2 + \|\nabla \cdot \omega\|_{0,\Omega}^2 \\
& + \|\nabla \times \mathbf{r} - (\mathbf{v} \cdot \nabla\omega - \omega \cdot \nabla\mathbf{v})\|_{0,\Omega}^2 + \|\nabla \cdot \mathbf{r}\|_{0,\Omega}^2 \ ,
\end{aligned} \tag{3.1}
$$

in which the diffusive term becomes small instead of the convective term becoming large and dominating the other terms in the functional. Figure 3.4 shows streamlines in the upstream half of the domain for flow over the backward-facing step at a Reynolds number of 200. Interestingly, when triquartic elements were used, excellent conservation ($> 99\%$ of mass was conserved) is achieved even though the boundary conditions on $\mathbf{n} \cdot \mathbf{r}$ were not set along the inflow. Consequently, in this case, we did not have to solve for the value of an unknown constant. Unfortunately, the lack of a boundary condition on $\mathbf{r}$ means that the operator is no longer elliptic, and the convergence rate of the linear solver is slower. The excellent conservation without a boundary condition on $\mathbf{r}$ along the inlet and outlet is not observed if a trilinear basis is used or if the Reynolds number is small (i.e., less than approximately 100 in this case). Regardless, this is an encouraging result because the requirement of an inflow boundary condition on $\mathbf{n} \cdot \mathbf{r}$ is one of the few drawbacks of the new formulation.

The final test problem is flow with $Re = 40$ through a series of bifurcations, as shown in Figure 3.5. The inlet is the single tube at the top, and the flow into the initial tube is the steady flow profile, which has a known analytic form. Boundary conditions along the inlet are very straightforward as a result of the steady profile. Velocity is set to $v_x = 0$, $v_y = 0$, and $v_z = -1 + (x^2 + y^2)$, normal vorticity is set to zero, and $\mathbf{n} \cdot \mathbf{r} = \frac{\partial^2 v_z}{\partial x^2} + \frac{\partial^2 v_z}{\partial y^2} = 4$. Because we wish to accurately capture the inflow boundary condition on velocity, a [illegible] finite element basis is used for this problem. The no-slip conditions, specified previously, are set along the walls. The main inlet tube branches into two tubes, and each of these tubes branch, giving a total of four outlet surfaces. The outflow boundary condition is, as always in computational fluid dynamics, difficult to determine. We begin by setting the tangential velocity to zero, which implies that the normal vorticity is also zero. Another potential boundary condition is setting the pressure to a constant by setting $\tau \cdot \mathbf{r} = 0$. However, better accuracy is usually found by setting a boundary condition on $\mathbf{n} \cdot \mathbf{r}$. If the flow is steady (i.e., fully developed), then $\mathbf{n} \cdot \mathbf{r} = \textit{constant}$, and the value of the *constant* can

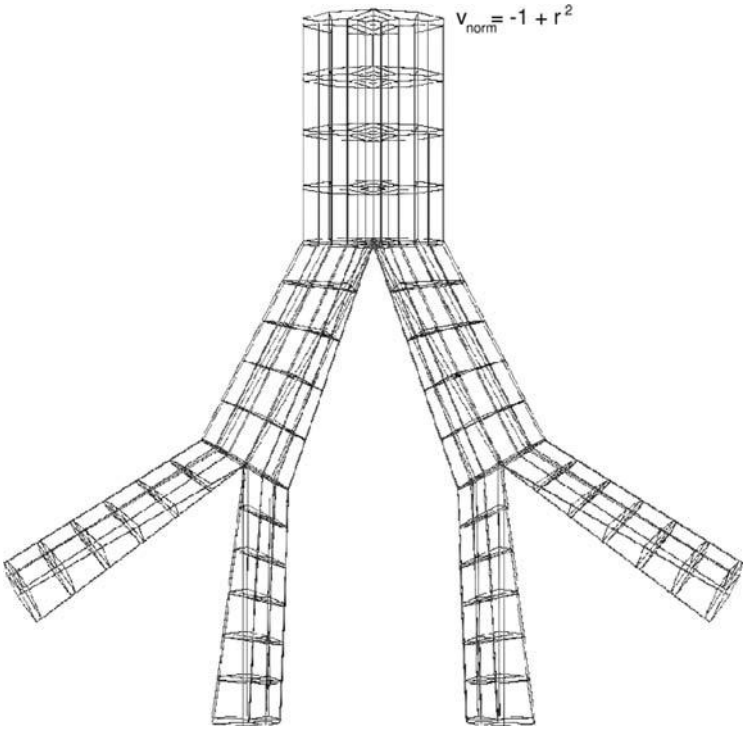


FIG. 3.5. *The hexahedral mesh used for the bifurcating flow domain. The inlet is the single tube at the top, and the four tubes at the bottom contain the outflow surfaces.*

be determined based on the geometry and setting the inflow rate equal to the outflow rate. However, because the tubes are not exact cylinders due to the bifurcations, it is unlikely that the outflow is fully developed. A third possibility is to assume that the flow has not developed fully (i.e., $\frac{\partial^2 v_n}{\partial n^2} \neq 0$), but well enough that it is a paraboloid at the outlet, albeit an unsteady one due to small changes in tube geometry. In this case, $\mathbf{n} \cdot \mathbf{r}$ is equal to an unknown constant, C_3. We solve for this unknown constant using the constraint that mass be conserved globally.

Figure 3.6 shows streamlines for the central cross-section of the bifurcating tubes domain. Also shown in the inset images are velocity vectors for specific branches. Because the triquartic finite element basis was based on Chebyshev–Lobatto node positions, which have been previously shown to work well with AMG solvers [20], the nodes are not evenly distributed but are somewhat clustered near the corners of elements. The global conservation of mass is approximately exact because C_3 was set to enforce this condition, but it is useful to examine the pressure gradient to determine the accuracy of the solution. For laminar flow in a straight, cylindrical tube, the relationship between the pressure gradient, flow rate (Q), and tube radius (R) is given by [27]

$$\frac{\partial p}{\partial n} \propto \frac{Q}{R^4}. \tag{3.2}$$

If the flow rate into the domain is Q, the flow rate through each of the outlets is approximately $Q/4$ due to symmetry. Further, for this geometry, the radius of the inlet is R, and the radius of each outlet tube is $R/2$. Combining these relationships

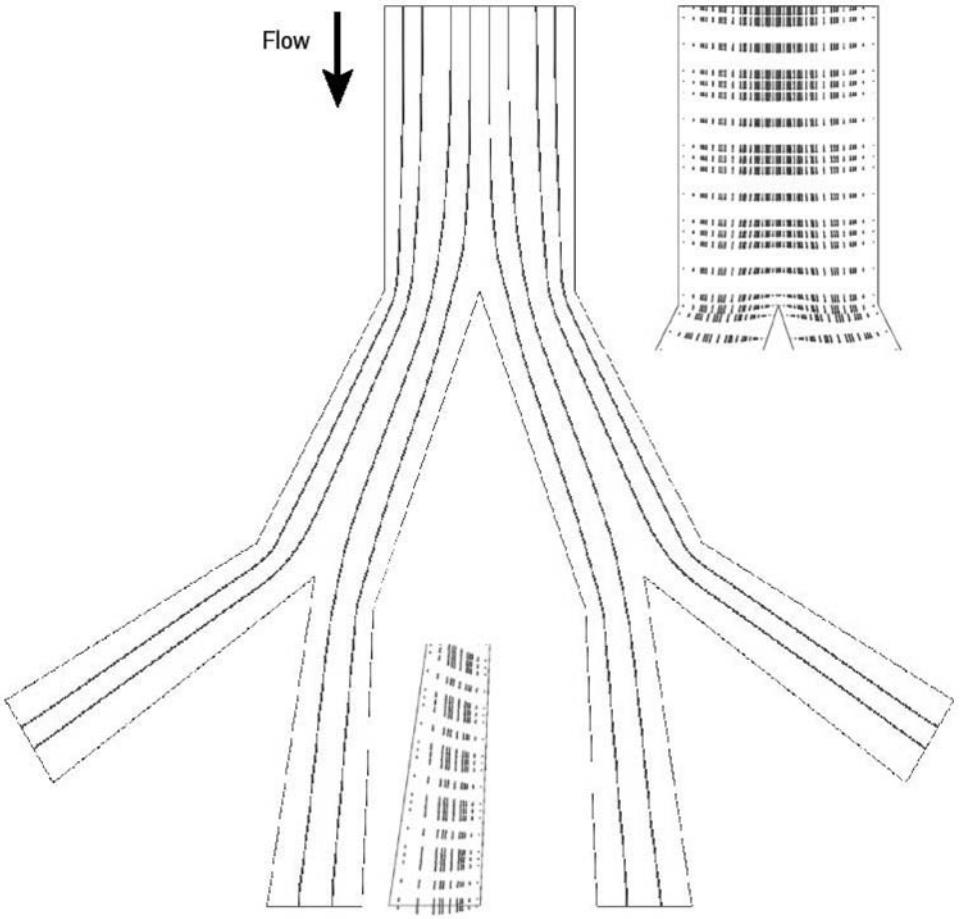


FIG. 3.6. *The central cross-section of the bifurcating tube domain showing streamlines through the bifurcations. The inset images show velocity vectors for the nearby tubes. The inlet is the single tube at the top, and the four tubes at the bottom contain the outflow surfaces.*

and using (3.2) results in the prediction that the normal pressure gradient at the outlet is roughly 4 times the normal pressure gradient at the inlet. Since the normal pressure gradient at the inlet is equal to 4, we expect that the normal pressure gradient at the outlet is roughly 16, but probably slightly higher due to the multiple bifurcations and entrance effects. For the result shown in Figure 3.6, the normal pressure gradient at the outlet was 17.4, indicating good agreement with our straight tube heuristic. Further, the approximation just presented can be used to obtain a good initial guess for the value of C_3.

Conclusions. Poor mass conservation can be observed when reformulating the Navier–Stokes equations as a first-order system using either the velocity-flux or vorticity formulations. The problem occurs primarily when the pressure (or normal stress) is not specified on both ends of the domain. To help address this problem, we are proposing a new first-order system of equations that allows boundary conditions to be set on the pressure gradient. In many cases, it is easier to specify boundary conditions on the pressure gradient because much of the information can be obtained from the

boundary conditions on velocity. Further, when using the new formulation in conjunction with AMG, significant computational improvements can also be observed. In this way, the new formulation achieves the objective of improved mass conservation on coarse grids while maintaining AMG convergence rates. The new formulation achieves only local mass conservation asymptotically as the mesh size goes to zero, like other least-squares formulations, but overall conservation is much improved for coarser meshes due to the boundary condition on $\mathbf{n}\cdot\mathbf{r}$. For each of the three test problems presented here—the square channel, the backward-facing step, and the series of bifurcations—excellent mass conservation was possible for both Stokes flow and low to moderate Reynolds numbers. Traditionally, high Reynolds number flows have been a strength of least-squares methods because of their excellent stability properties. However, high Reynolds numbers also imply that flow through the outflow boundary is unlikely to be steady, a difficult condition to handle with the new formulation. However, we are confident that as experience is gained with the new formulation, the proper boundary conditions can be found for a much wider set of problems.

REFERENCES

[1] B. F. Armaly, F. Durst, J. C. F. Pereira, and B. Schönung, *Experimental and theoretical investigation of backward-facing step flow*, J. Fluid Mech., 127 (1983), pp. 473–496.

[2] T. Austin, T. Manteuffel, and S. McCormick, *A robust approach to minimizing h(div) dominated functionals in an h^1-conforming finite element space*, J. Numer. Linear Algebra Appl., 11 (2004), pp. 115–140.

[3] P. B. Bochev, *Analysis of least-squares finite element methods for the Navier–Stokes equations*, SIAM J. Numer. Anal., 34 (1997), pp. 1817–1844.

[4] P. B. Bochev, Z. Cai, T. M. Manteuffel, and S. F. McCormick, *Analysis of velocity-flux first-order system least-squares principles for the Navier–Stokes equations: Part* I, SIAM J. Numer. Anal., 35 (1998), pp. 990–1009.

[5] P. B. Bochev and M. D. Gunzburger, *Accuracy of least-squares methods for the Navier-Stokes equations*, Comput. Fluids, 22 (1993), pp. 549–563.

[6] P. B. Bochev and M. D. Gunzburger, *Finite element methods of least-squares type*, SIAM Rev., 40 (1998), pp. 789–837.

[7] P. B. Bochev, T. A. Manteuffel, and S. F. McCormick, *Analysis of velocity-flux least-squares principles for the Navier–Stokes equations: Part* II, SIAM J. Numer. Anal., 36 (1999), pp. 1125–1144.

[8] P. Bolton and R. W. Thatcher, *On mass conservation in least-squares methods*, J. Comput. Phys., 201 (2004), pp. 110–118.

[9] W. L. Briggs, V. E. Henson, and S. F. McCormick, *A Multigrid Tutorial*, 2nd ed., SIAM, Philadelphia, 2000.

[10] Z. Cai, B. Lee, and P. Wang, *Least-squares methods for incompressible Newtonian fluid flow: Linear stationary problems*, SIAM J. Numer. Anal., 42 (2004), pp. 843–859.

[11] Z. Cai, T. A. Manteuffel, S. F. McCormick, and J. Ruge, *First-order system $\mathcal{LL}^*$ (FOSLL*): Scalar elliptic partial differential equations*, SIAM J. Numer. Anal., 39 (2001), pp. 1418–1445.

[12] Z. Cai, T. A. Mantueffel, and S. F. McCormick, *First-order system least squares for second-order partial differential equations: Part* II, SIAM J. Numer. Anal., 34 (1997), pp. 425–545.

[13] C. L. Chang and J. J. Nelson, *Least-squares finite element method for the Stokes problem with zero residual of mass conservation*, SIAM J. Numer. Anal., 34 (1997), pp. 480–489.

[14] A. L. Codd, T. A. Manteuffel, and S. F. McCormick, *Multilevel first-order system least squares for nonlinear partial differential equations*, SIAM J. Numer. Anal., 41 (2003), pp. 2197–2209.

[15] A. L. Codd, T. A. Manteuffel, S. F. McCormick, and J. W. Ruge, *Multilevel first-order system least squares for elliptic grid generation*, SIAM J. Numer. Anal., 41 (2003), pp. 2210–2232.

[16] J. M. Deang and M. D. Gunzburger, *Issues related to least-squares finite element methods for the Stokes equations*, SIAM J. Sci. Comput., 20 (1998), pp. 878–906.

[17] R. D. Falgout and U. M. Yang, *hypre: A library of high performance preconditioners*, in Computational Science—ICCS 2002, Part III, Lecture Notes in Comput. Sci. 2331, Springer-Verlag, Berlin, 2002, pp. 632–641.
[18] P. M. Gresho and R. L. Sani, *Introducing four benchmark solutions*, Internat. J. Numer. Methods Fluids, 11 (1990), pp. 951–952.
[19] J. J. Heys, T. A. Manteuffel, S. F. McCormick, and J. W. Ruge, *First-order systems least squares (FOSLS) for coupled fluid-elastic problems*, J. Comput. Phys, 195 (2004), pp. 560–575.
[20] J. J. Heys, T. A. Manteuffel, S. F. McCormick, and L. N. Olson, *Algebraic multigrid for higher-order finite elements*, J. Comput. Phys, 204 (2005), pp. 520–532.
[21] B. N. Jiang, *The Least-Squares Finite Element Method*, Springer-Verlag, Berlin, 1998.
[22] B. N. Jiang, L. J. Hou, T. L. Lin, and L. A. Povinelli, *Least-squares finite element solutions for three-dimensional backward-facing step flow*, Internat. J. Comput. Fluid Dynamics, 4 (1995), pp. 1–19.
[23] G. Karypis and V. Kumar, *A fast and high quality multilevel scheme for partitioning irregular graphs*, SIAM J. Sci. Comput., 20 (1998), pp. 359–392.
[24] J. P. Pontaza and J. N. Reddy, *Space-time coupled spectral/hp least-squares finite element formulations for the incompressible Navier-Stokes equations*, J. Comput. Phys., 197 (2004), pp. 418–459.
[25] O. Röhrle, *Multilevel First-Order System Least Squares for Quasilinear Elliptic Partial Differential Equations*, Ph.D. thesis, Department of Applied Mathematics, University of Colorado at Boulder, Boulder, CO, 2004.
[26] L. Q. Tang and T. H. Tsang, *A least-squares finite element method for time-dependent incompressible flows with thermal convections*, Internat. J. Numer. Methods Fluids, 17 (1993), pp. 271–289.
[27] F. M. White, *Viscous Fluid Flow*, 2nd ed., McGraw-Hill, New York, 1991.
[28] F. M. White, *Fluid Mechanics*, 5th ed., McGraw-Hill, Boston, MA, 2003.

SIAM J. SCI. COMPUT.
Vol. 28, No. 5, pp. 1694–1715
© 2006 Society for Industrial and Applied Mathematics

A VARIATIONAL APPROACH TO MODELING AND SIMULATION OF GRAIN GROWTH*

DAVID KINDERLEHRER†, IRENE LIVSHITS‡, AND SHLOMO TA'ASAN†

Abstract. Most technologically useful materials arise as polycrystalline microstructures, composed of a myriad of small crystallites, called grains, separated by their interfaces, called grain boundaries. The orientations and arrangements of the grains and their network of boundaries are implicated in many properties across wide scales, for example, functional properties, like conductivity in microprocessors, and lifetime properties, like fracture toughness in structures. Simulation is becoming an important tool for understanding both materials properties and their processing requirements. Here we offer a consistent variational approach to the mesoscale simulation of these systems subject to the Mullins equation of curvature-driven growth in a two-dimensional setting. The main objective is to provide a calibration for future two-dimensional and three-dimensional efforts. We discuss several novel features of our approach, which we anticipate will render it a flexible, scalable, and robust tool to aid in microstructural prediction. Simulation results offer compelling evidence of the predictability and robustness of statistical properties of large systems, such as grain size distribution and texture, that are of immediate interest in materials science.

Key words. polycrystalline materials, grain growth, simulation, Mullins equation

AMS subject classifications. 15A15, 15A09, 15A23

DOI. 10.1137/030601971

1. Introduction.

A central problem in materials science is the understanding and control of microstructure, the ensemble of grains that comprise polycrystalline materials. The orientations and arrangements of the grains and their network of boundaries are implicated in many properties across wide scales, for example, functional properties, like conductivity in microprocessors, and lifetime properties, like fracture toughness in structures. As discussed below, these mesoscale properties are necessarily interpreted by means of statistical measures, especially grain size statistics and texture, or grain orientation, statistics. Most tantalizing is the grain boundary character distribution (GBCD), a five-dimensional distribution function of grain misorientation and interface normals. Modeling and simulation of the grain evolution at the mesoscale plays an increasingly important role in resolving these central problems. Indeed, the grain boundary character distribution owes much of its discovery to the very simulation we are presenting here.

With the implementation of automated data acquisition, based on orientation imaging microscopy, in the Mesoscale Interface Mapping Project, a strategy has emerged for determining interfacial energy and mobility in polycrystals [1]. Simulation with such energy and mobility functions offers the promise of understanding these properties and satisfying processing requirements. Here we exhibit a consistent variational approach to the mesoscale simulation of large systems of grain boundaries sub-

*Received by the editors November 25, 2003; accepted for publication (in revised form) May 1, 2006; published electronically October 12, 2006.
http://www.siam.org/journals/sisc/28-5/60197.html

†Center for Nonlinear Analysis and Department of Mathematical Sciences, Carnegie Mellon University, Pittsburgh, PA 15213 (davidk@andrew.cmu.edu, shlomo@andrew.cmu.edu). The work of the third author was supported by the MRSEC program of the NSF under award DMR-0520425, the Center for Nonlinear Analysis under awards NSF DMS 0405343 and NSF DMS 0305794, and the DoE Computational Materials Science Network.

‡Department of Mathematical Sciences, Ball State University, Muncie, IN 47306 (ilivsits@bsu.edu).

ject to the Mullins equations of curvature-driven growth [31], [32] in a two-dimensional setting. We illustrate this with the simulation of a polycrystalline system with constant energy and mobility. This serves as a calibration for future two-dimensional and three-dimensional efforts. This paradigm system is the most studied in theory and corresponds in experiment to a columnar "fiber texture." We discuss several novel features of our approach, which we anticipate will render it a flexible, scalable, and robust tool to aid in microstructural prediction. We offer compelling evidence of the predictability and robustness of statistical properties of large systems, such as grain size distribution and texture, that are of immediate interest in materials science. As a consequence, the simulation technique enters the arena of being useful. For example: How important is anisotropy? How prominent are low angle boundaries in various textures? Or (see below) are there general laws of evolution for ensembles of grains? Such questions become accessible now that we are in possession of a reliable and accurate approach to simulation.

The Mullins equations, discussed below, are a system of evolution parabolic partial differential equations for each grain boundary curve. Grain boundaries meet, typically, at triple junctions, where an additional condition is required. We enforce the Herring force balance condition, which is the natural boundary condition for equilibrium, at these triple junctions [17], [18]. The resulting system is then dissipative for the energy [22]. The evolution of grain boundaries using the above model can be viewed as a steepest descent method for the total grain boundary energy, where natural boundary conditions are imposed. This observation serves as the basis for our discretization method and leads to stable schemes. Our method begins by discretizing the grain boundaries, which are curves in two dimensions, using linear elements. We then form the discrete energy by numerical integration (using the midpoint rule) of the grain boundary energy density along the network of grains. The gradient of this functional gives the velocity of the curves, and the discretized Herring condition is given as the discrete natural boundary condition for that functional. The resulting semidiscrete system (continuous time) satisfies a dissipation estimate analogous to that of the original problem. This guarantees the stability for small time steps of the whole network. The Herring condition at the discrete level provides us with a nonlinear set of equations that the triple junctions must satisfy. These are solved approximately using a fixed point method. Time stepping uses a first order discretization. The scheme here is explicit, so there is the usual time stepping restriction of $\triangle t = O(h^2)$, where h is a parameter describing the grain boundary discretization. One of the features of this approach is that the data structure consists of curves only and does not require the discretization of the grains. In this way we deal with discretized systems that are of lower dimension than the physical space. This permits us to have very large configurations: initial configurations typically consist of 25,000 grains.

In addition to using the gradient of the above functional to determine the grain boundary evolution, we had to implement treatment of certain critical events. During evolution, a grain boundary or a grain may shrink and disappear. This creates multiple junctions that are unstable. Such multiple junctions split into triple junctions in a way that is consistent with energy reduction. Thus, we introduce, one at a time, new edges, in a way that the energy will result in maximum reduction.

Convergence of approximation schemes, of which this is one, is a standard problem in numerical analysis. Usually there is a natural topology associated with a problem; for example, when we deal with solution of the Poisson equation, we examine convergence in H^1. The situation here is more complex since it is not clear that the network as a whole has convergence of the same type as the curves (network links). In any

event, we may not have convergence of the configuration of the network, but only of certain statistical properties. As an analogy consider a large system of interacting particles, say with a Lennard–Jones interaction force. Individual trajectories are not predictable for large times, but statistical properties, such as temperature, density, and average velocity, are predictable. In the present context many theoretical questions remain open. We are faced with a vast number of solutions of time dependent partial differential equations which satisfy relations among themselves, like being boundaries of the same grain. There need not be any characterizing statistical properties. If there are, we are faced with understanding the meaning of "convergence" for such properties.

The numerical approach described here aims at large networks of grains, with emphasis toward distribution functions such as grain size, numbers of edges, and orientations. We succeed in showing that these distribution functions are robust from a computational standpoint. We verify some elementary diagnostics, for example, the von Neumann–Mullins $(n-6)$ rule for area evolution of individual grains [31], [38], as explained later.

Essentially, we are faced with a large metastable system. Ideally, one would like to have an evolution equation for such distribution functions, so that the simulation of the quantities of interest for scientific and engineering purposes will be as efficient as possible. This would constitute a coarse graining of the system in a dynamic fashion analogous to the way in which the equations of fluid dynamics describe the interacting particles mentioned above when large temporal and spatial scales are considered. We discuss the use of this simulation to derive stochastic models for grain growth that are valid for large time scales (cf. [5] for a preliminary version). In [24], a first attempt to understand the evolution of grain growth statistics derived from this simulation is discussed.

There is an extensive and distinguished literature in the simulation of grain growth. These models can be divided into the following major groups: molecular dynamics (MD) [8], [16], [33], [34], [35], [19], [37], [39], [40], [42], Monte-Carlo (MC) [2], [3], [28], [29], [41], [42], front tracking [14], [15], vertex models [43], [44], [45], phase field models [6], [7], [25], [26], [42], partial differential equations (PDE) [9], [24], [27], and distribution function models [10], [11], [12], [13]. Each modeling approach is appropriate in studying grain growth on a particular length or time scale. For instance, understanding local topological changes occurring during grain growth can be best achieved using MD models, while studying abnormal grain growth is more appropriate by MC or PDE models.

2. Theory. In preparation for discussing our grain growth algorithm, we briefly review the elements of the theory of curvature-driven growth developed by Herring [17], [18] and Mullins [32]. This theory is based on thermodynamic considerations. In this paper we consider two-dimensional grain growth; an extension of the theory to three dimensions for most parts is rather straightforward [20], [21]. (This is not true for the simulation, of course!) Different viewpoints are possible for the description of the Mullins curvature-driven growth theory; here we present it in a form a best suited for subsequent discretization purposes.

Let us consider a collection of grain boundaries, which are smooth curves, meeting at triple junctions. The dynamics are governed by a given energy density function σ and a given mobility function μ. Consider a set of smooth curves, i.e., the grain boundaries

$$\mathbf{\Gamma} = \mathbf{\Gamma}(t) = \{\Gamma^1, \Gamma^2, \ldots, \Gamma^K\}, \quad K = K(t), \tag{2.1}$$

where each curve is given in a parametrized form

$$\Gamma^k(t) = \{x = \xi^k(s,t), 0 \le s \le 1\}. \tag{2.2}$$

The curves are assumed to form a network, meeting at triple junctions or fixed points, for example the points on the outside border of the configuration.

We consider the set of triple junctions

$$\mathbf{\Lambda} = \mathbf{\Lambda}(t) = \{z^1, z^2, \dots, z^M\}, \quad M = M(t), \tag{2.3}$$

consisting of all points that belong to three grain boundaries simultaneously; namely,

$$z^m(t) = \xi^{k_1}(s_{k_1}, t) = \xi^{k_2}(s_{k_2}, t) = \xi^{k_3}(s_{k_3}, t)$$

for $s_{k_i} = 0$ or $s_{k_i} = 1$; k_1, k_2, k_3 being distinct integers in $\{1, \dots, K(t)\}$.

We assume that the grain boundary energy density function, σ, is given in terms of misorientation angle α and θ, the angle that the normal makes with the x-axis. Each grain boundary Γ^k is associated with a grain boundary energy, σ^k. The total grain boundary energy of the network can then be written as

$$E(t) = \sum_{k=1}^{K} \int_0^1 \sigma^k(\theta^k(s,t), \alpha^k)|l^k(s,t)|ds, \qquad l^k = \frac{d\xi^k}{ds}. \tag{2.4}$$

This energy is equal to the amount of work required to create an infinitesimal amount of new surface. The next step is to describe grain boundary motion that leads to energy dissipation. For an arbitrary motion of the grain boundaries, specified by $\xi^k(s,t)$, $0 \le s \le 1$, the rate of change in energy is

$$\frac{d}{dt}E(t) = \sum_{k=1}^{K} \int_0^1 \left(\frac{\partial \sigma}{\partial \theta}^k \frac{d\theta^k}{dt}|l^k| + \sigma^k \frac{l^k}{|l^k|} \cdot \frac{dl^k}{dt} \right) ds. \tag{2.5}$$

We now define a unit tangent vector $b^k = l^k/|l^k|$ and a unit normal vector n^k given by $n^k = Rb^k$, where R is rotation through $\pi/2$. With v denoting the velocity of the curve Γ, using the obvious relations

$$\frac{dl^k}{dt} = \frac{d}{dt}\frac{d\xi^k}{ds} = \frac{d}{ds}\frac{d\xi^k}{dt} = \frac{dv^k}{ds} \tag{2.6}$$

and

$$\frac{d\theta^k}{dt} = \frac{1}{|l^k|}\frac{dl^k}{dt} \cdot n^k, \tag{2.7}$$

we find that (2.5) reads as

$$\frac{d}{dt}E(t) = \sum_{k=1}^{N} \int_0^1 \left(\frac{\partial \sigma}{\partial \theta}^k n^k + \sigma^k b^k \right) \cdot \frac{dv^k}{ds} ds. \tag{2.8}$$

Then (2.8) can be rewritten as

$$\frac{d}{dt}E(t) = \sum_{k=1}^{N} \int_0^1 T^k \cdot \frac{dv^k}{ds} ds, \tag{2.9}$$

where vector T^k,

$$T^k = \frac{\partial \sigma^k}{\partial \theta} n^k + \sigma^k b^k, \tag{2.10}$$

is the line tension or capillary stress vector, and dT^k/ds is the line force per unit length. Integration by parts in (2.9) leads to

$$\frac{d}{dt}E(t) = -\sum_{k=1}^{K} \int_0^1 \frac{dT^k}{ds} \cdot v^k ds + \sum_{m=1}^{M} v^m \cdot \sum_{l=1}^{3} T^{m,l}, \tag{2.11}$$

where $T^{m,l}$ is the line stress vector on grain boundary Γ^l that ends at the triple junction z^m. The grain growth process entails several possible mechanisms which, at the mesoscale, amount to consideration of exchange of matter across the curve. This results in the creation of a new area on one side. Mullins' theory is that the normal velocity v_n of a grain boundary (the rate of creation of area), is proportional to the line force (the rate at which work is done), through the factor of the mobility μ:

$$v_n = \mu \frac{dT}{ds}. \tag{2.12}$$

From the equation above we conclude that the Mullins theory can be viewed as evolution resulting in steepest descent for the energy,

$$\frac{d}{dt}E(t) = -\sum_{k=1}^{K} \int_0^1 \frac{1}{\mu^k} |v_n^k|^2 ds + \sum_{m=1}^{M} v^m \cdot \sum_{l=1}^{3} T^{m,l}. \tag{2.13}$$

The Herring condition,

$$\sum_{l=1}^{3} T^{m,l} = 0, \qquad m = 1, \ldots, M, \tag{2.14}$$

is the natural boundary condition for this problem, and it, together with (2.12), leads to the energy dissipation:

$$\frac{d}{dt}E(t) = -\sum_{k=1}^{K} \int_0^1 \frac{1}{\mu^k} |v_n^k|^2 ds \le 0. \tag{2.15}$$

From the Frenet formulas, for a grain boundary Γ we conclude that

$$\frac{dT}{ds} = \left(\frac{d^2\sigma}{d\theta^2} + \sigma \right) \kappa n \qquad \text{on } \Gamma, \tag{2.16}$$

where κ is the curvature of Γ. The evolution equations then take the form

$$v_n = \mu \left(\frac{d^2\sigma}{d\theta^2} + \sigma \right) \kappa n \qquad \text{on } \Gamma. \tag{2.17}$$

At equilibrium

$$\frac{dT^k}{ds} = 0,$$

or, equivalently,

$$\left(\frac{d^2\sigma}{d\theta^2}+\sigma\right)\kappa = 0 \qquad \text{on } \Gamma. \tag{2.18}$$

Thus we recover that in equilibrium the mean curvature, $\kappa = 0$, or the grain boundaries are straight segments.

This theory is applicable to most of the process of grain growth. However, grains do not necessarily always retain their integrity. Two types of major network changes, different from a regular grain boundary and triple junction evolution, occur occasionally during grain growth. We name such events , , , and we distinguish between two main events.

When the length of a grain boundary becomes 0, it disappears, and its two triple junctions become one quadruple-junction, a point where four grain boundaries meet. Triple junctions are typically the only stable junctions, and therefore the quadruple-junction instantly splits into two new triple junctions connected by an infinitesimally small new grain boundary. The split is energy decreasing, and the dissipative quality of the evolution is maintained.

As a small grain shrinks, it eventually becomes a point at which three, four, or five grain boundaries meet. The area of a grain with six or more grain boundaries either does not change or increases. This conclusion comes from the von Neumann–Mullins $(n-6)$ rule [31], [38]. In a system with σ and μ constant and satisfying the Herring condition, the area $A(t)$ of a grain with n sides satisfies

$$\frac{d}{dt}A(t) = \frac{\pi}{3}\sigma\mu(n-6). \tag{2.19}$$

We shall employ this as a diagnostic for numerical simulations.

The newly formed multijunction then instantly splits into triple junctions, one after the other. (In other words, quadruple junctions split into two triple junctions, the junctions with five incoming grain boundaries first split into a triple junction and a quadruple junction, and the latter immediately splits into two triple junctions.)

3. Semidiscrete formulation. Our approach for discretizing the problem is to start by defining a discrete version of the total energy, analogous to (2.4). The evolution in the continuous case was shown to be related to the gradient of the energy, and this is the path we follow for the discretized energy. In this semidiscrete formulation we assume space discretization, keeping time a continuous variable.

We begin by considering a network of grain boundaries

$$\mathbf{\Gamma} = \mathbf{\Gamma}(t) = \{\Gamma^1, \Gamma^2, \ldots, \Gamma^K\}, \quad K = K(t), \tag{3.1}$$

where each grain boundary is a piecewise linear curve represented by a set of its nodal points

$$\Gamma^k(t) = \{x_j^k(t) | j = 1, \ldots, N_k(t)\}.$$

The set of triple junctions is the set of all nodal points that belong to exactly three grain boundaries,

$$\mathbf{\Lambda} = \mathbf{\Lambda}(t)\{z^1, \ldots, z^M\}, \quad M = M(t), \tag{3.2}$$

where each $z^m(t)$ is defined as

$$z^m = x_{l_1}^{k_1}(t) = x_{l_2}^{k_2}(t) = x_{l_3}^{k_3}(t) \quad \text{for} \quad l_n \in \{1, N_{k_n}\},\ n = 1, 2, 3,$$

with $k_1, k_2, k_3 \in \{1, \dots, K(t)\}$ being distinct. In the algorithm description we also allow quadruple junctions in $\mathbf{\Lambda}$. They appear as a result of the disappearance of grains and grain boundaries, and they split into two triple junctions within the same time step. In addition, we consider a set of all grains

$$\mathbf{\Omega} = \{G^1, \dots, G^N\}, \qquad N = N(t). \tag{3.3}$$

Our approximation for the total energy uses the midpoint rule for approximating integrals. Thus, we need to evaluate different quantities at the midpoints of the segments connecting successive nodal points. We introduce notation $x^k_{j-1/2} = (x^k_j + x^k_{j-1})/2$ and $\theta^k_{j-1/2} = \theta(x^k_{j-1/2})$, the angle that the vector normal to $[x^k_{j-1}, x^k_j]$ makes with the x-axis. The unit normal and unit tangent vectors for the linear segment $[x^k_{j-1}, x^k_j]$ are given by $n^k_{j-1/2} = (\cos\theta^k_{j-1/2}, \sin\theta^k_{j-1/2})$ and $b^k_{j-1/2} = (x^k_j - x^k_{j-1})/|x^k_j - x^k_{j-1}|$, respectively. The energy density function is evaluated at midpoints and is denoted by $\sigma^{k,j-1/2}$.

The discrete energy E^h, approximating (2.4) by the midpoint rule, reads as

$$E^h(t) = \sum_{k=1}^{K} \sum_{j=1}^{N_k} \sigma^{k,j-1/2} |x^k_j(t) - x^k_{j-1}(t)|, \tag{3.4}$$

with a discretization parameter $h = \max_j |x^k_j(t) - x^k_{j-1}(t)|$.

Following the continuous derivation, we compute the rate of change of the semidiscrete energy,

$$\begin{aligned} \frac{d}{dt}E^h(t) = &\sum_k \sum_j \sigma^{k,j-1/2} \frac{d}{dt}|x^k_j(t) - x^k_{j-1}(t)| \\ &+ \sigma^{k,j-1/2}_\theta \frac{d}{dt}\theta^{j-1/2}(t)|x^k_j(t) - x^k_{j-1}(t)|. \end{aligned} \tag{3.5}$$

To simplify (3.5), we will use

$$\frac{d}{dt}|x^k_j(t) - x^k_{j-1}| = \langle \dot{x}^k_j(t) - \dot{x}^k_{j-1}, b^k_{j-1/2}(t)\rangle \tag{3.6}$$

and

$$\frac{d}{dt}\theta^k_{j-1/2}(t) \approx \frac{1}{|x^k_j(t) - x^k_{j-1}|}\langle \dot{x}^k_j - \dot{x}^k_{j-1}, n^k_{j-1/2}(t)\rangle. \tag{3.7}$$

Equality (3.7) follows from the relation

$$\delta\theta_{j-1/2} \approx \tan\delta\theta_{j-1/2} = \frac{1}{|x^k_j(t) - x^k_{j-1}|}\langle \delta x_j - \delta x_{j-1}, n_{j-1/2}\rangle, \tag{3.8}$$

which can be easily verified.

Next, for each triple junction, z_m, we define the integers $k_1(m), k_2(m), k_3(m)$, the indices of the three grain boundaries that meet at this triple junction; and $p_l(m)$, $l = 1, 2, 3$, the indices of the intervals that connect the triple junction with the nearest nodal point on the grain boundary $k_l(m)$. The expression for $p_l(m)$ is then given by

$$p_l(m) = \begin{cases} 1\frac{1}{2}, & x^{k_l(m)}_1 = z^m, \\ N_{k_l(m)} - \frac{1}{2}, & x^{k_l(m)}_{N_{k_l(m)}} = z^m, \end{cases} \qquad l = 1, 2, 3. \tag{3.9}$$

Substitution of (3.6) and (3.7) into (3.5) leads to

$$\frac{d}{dt}E^h(t) = -\sum_{k=1}^{K}\sum_{j=2}^{N_k-1}\langle \dot{x}_j^k(t), T^{k,j+1/2} - T^{k,j-1/2}\rangle + \sum_{m=1}^{M}\left\langle \dot{z}^m(t), \sum_{l=1}^{3} T^{k_l(m),p_l(m)}\right\rangle, \tag{3.10}$$

where

$$T^{k,j\pm1/2} = \frac{\partial\sigma}{\partial\theta}^{k,j\pm1/2} n^{k,j\pm1/2} + \sigma^{k,j\pm1/2} b^{k,j\pm1/2}. \tag{3.11}$$

This is in complete analogy to (2.11) of the continuous case. As a result, the motion of interior nodal points is given by

$$v_j^k(t) = \frac{2\mu_j^k}{|x_{j+1}^k - x_{j-1}^k|}(T^{k,j+1/2} - T^{k,j-1/2}), \tag{3.12}$$

where scaling by $2/|x_{j+1}^k - x_{j-1}^k|$ makes the correspondence to the continuous velocity, $v_n^k(t)$. The resulting scheme is a central difference scheme approximation to (2.17).

A discrete Herring condition at the triple junction z_m is given as

$$\sum_{l=1}^{3} T^{k_l(m),p_l(m)} = 0, \qquad m = 1,\dots,M(t). \tag{3.13}$$

It is easy to see that the semidiscrete problem has the same dissipative property as the continuous problem.

4. Numerical implementation. The data structure for execution of the simulation consists of grain boundaries, triple junctions, and grains, and is managed using standard linked lists. Each grain boundary Γ^k has pointers to the two grains sharing it as a boundary, G_0^k and G_1^k ($G(\Gamma^k) = \{G_0^k, G_1^k\}$). Each grain boundary Γ^k also has pointers to its end points, triple junctions z_0^k and z_1^k ($Z(\Gamma^k) = \{z_0^k, z_1^k\}$) . Each triple junction z_m has pointers to three grain boundaries, Γ_0^m, Γ_1^m, and Γ_2^m, that meet at that point ($\Gamma(z^m) = \{\Gamma_0^m, \Gamma_1^m, \Gamma_2^m\}$). Each grain G^n has pointers to a list of its grain boundaries, $\Gamma(G^n)$.

In this numerical implementation, grain boundaries are defined by the set of nodal points and are approximated using linear elements. Evolution is done by moving nodal points according to (3.12)–(3.13).

In our experiments, we specified a global mesh size, h, and uniformly discretized grain boundaries with local mesh size (distance between neighboring nodal points) satisfying

$$\frac{3}{4}h < |x_j^k - x_{j-1}^k| < \frac{3}{2}h. \tag{4.1}$$

The mesh size h, the typical distance defined at the beginning of the simulation, remains constant through the simulation. The number of nodal points for each grain boundary is updated, and the points are redistributed in order to keep the distance between each pair of neighboring nodal points in the range (4.1).

The simulation was done by first evolving the grain boundaries, according to Mullin's equation, and then updating the triple points according to Herring's condition, as explained in what follows.

Triple junction motion. To satisfy the Herring condition (2.14), one needs to solve a nonlinear equation for triple junctions z. In the simulation, an iterative process

$$z^{n+1} = \frac{\sum_{j=1}^{3} x_{l_j}^{k_j} / |z^n - x_{l_j}^{k_j}|}{\sum_{j=1}^{3} 1/|z^n - x_{l_j}^{k_j}|} \tag{4.2}$$

is used to find a new position of a triple junction. Points $x_{l_j}^{k_j}$ are the nearest to the triple junction inner nodal points of grain boundaries that met at z. (Under the assumption of isotropic grain boundary energy, the term that contains a grain boundary energy derivative disappears.) The process stops when

$$\left| \sum_{j=1}^{3} \frac{z - x_{l_j}^{k_j}}{|z - x_{l_j}^{k_j}|} \right| < \epsilon, \tag{4.3}$$

where ϵ is a small number (10^{-6} in our simulation.) For some triple junction configurations, however, (4.3) is impossible to satisfy (it happens when process (4.2) forces z to take the place of one of $x_{l_j}^{k_j}$). In such cases the new position of the triple junction is calculated as follows:

$$z \leftarrow z + dt \sum_{j=1}^{3} \frac{z - x_{l_j}^{k_j}}{|z^k - x_{l_j}^{k_j}|}; \tag{4.4}$$

namely, it is moved a little in the right direction.

Grain growth. As grain growth proceeds, two types of critical events occur. Short grain boundaries and small grains shrink below a certain size, triggering two processes called grain boundary flipping and grain disappearance.

Grain boundary removal. In the numerical simulation the grain boundary is assumed to be too short if its length is less than $1.5h$. This condition is related to the fact that the minimal number of nodal points for a grain boundary must be three to guarantee at least one interior (nonjunction) nodal point. When the grain boundary becomes short enough so that in order to satisfy condition (4.1) the interior nodal points must be eliminated, this grain boundary becomes subject for removal, if it is growing (e.g., it is longer than at the previous time step).

When a small grain boundary is removed, a temporary quadruple junction is created and then splits into a pair of new triple junctions (as depicted in Figure 4.1) connected by a new small (with length $h/4$) grain boundary. This new grain boundary will most likely grow during the next time step and, therefore, will not disappear in flipping events. The old grain boundaries that meet at each of two new triple junctions most likely did not have a common triple junction in the configuration prior to the grain boundary disappearance.

Grain boundary flipping (Figure 4.1) consists of the following stages. I. The short horizontal grain boundary is a subject to removal. II. After its disappearance two triple junctions are shut into a quadruple junction. III. The quadruple junction splits into two different (from the previous configuration) triple junctions connected by a short vertical grain boundary. IV. After the Herring condition is satisfied for the two new triple junctions, the new grain boundary grows.

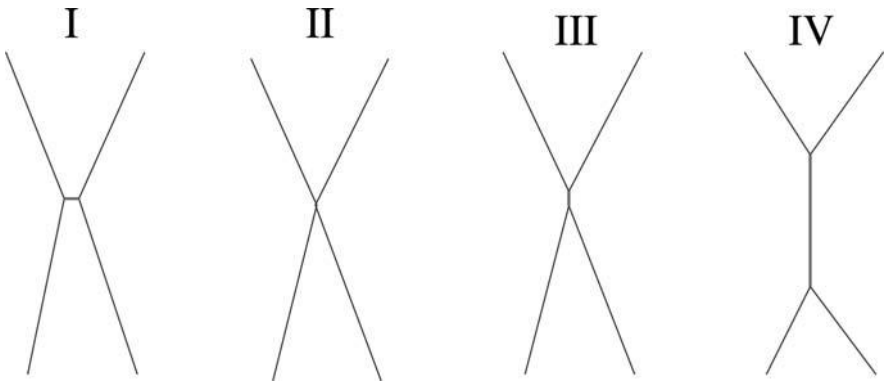


FIG. 4.1. *Critical event: a grain boundary flipping.*

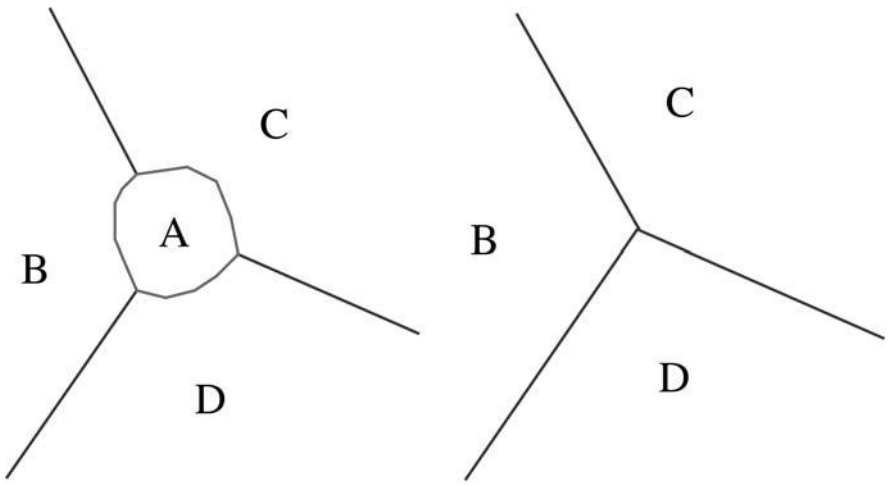


FIG. 4.2. *Critical event: a grain disappearance.*

According to the $(n-6)$ rule, a grain with less than six sides shrinks and will eventually disappear (unless the number of its boundary segments increases as a result of topological changes). Our numerical implementation removes grains when their area becomes smaller than a minimal area ($4h^2$ in our experiments). The actual process of grain disappearance involves an instantaneous shrinking of a three- and a four-sided grain with minimal area into triple or quadruple junctions, correspondingly. As the results of the numerical simulations show, a five-sided grain is more likely to first lose a grain boundary, i.e., become a four-sided grain, and only then disappear. Therefore, to avoid the technicalities that appear in evolving a junction with five incoming grain boundaries, the disappearance of such grains is performed in two time steps. At the first time step, the shortest grain boundary flips, leaving a four-sided small grain, which disappears during the next time step.

The disappearance of a grain with three grain boundaries (grain A in Figure 4.2) leads to the following local changes: the grain shrinks into one point, and each of its immediate neighbors loses one grain boundary. All grain boundaries of A and the triple junctions that were connected to these grain boundaries are erased. One new triple junction and no new grain boundaries are added to the network.

TABLE 5.1
Circular grain: mesh size dependence.

Radius	n points	h_n	δA_n	$\delta A_{2n} - \delta A_n$
1	4	1.41421	-0.0565645	-0.0046615
1	8	0.765367	-0.061226	-0.0011997
1	16	0.3901810	-0.0624257	-0.0003021
1	32	0.196034	-0.0627278	-0.0000756
1	64	0.0981353	-0.0628034	-0.0000190
1	128	0.0490825	-0.0628224	

TABLE 5.2
Circular grain: curvature dependence.

Radius r	n points	h_n	δA_n	$\delta A_{2r} - \delta A_r$
0.25	4	0.353553	-0.0282477	-0.0025274
0.50	9	0.342020	-0.0307751	-0.0004800
1.00	18	0.347296	-0.0312551	-0.0001206
2.00	36	0.348623	-0.0313757	-0.0000302
4.00	72	0.348955	-0.0314059	-0.0000075
8.00	144	0.349038	-0.0314134	

Forward Euler time discretization is employed in the algorithm, posing a strong limitation on time stepping, $\triangle t = O(h^2)$. The discrete times are denoted by $t_p,\ p = 0, 1, \ldots$, where $t_{p+1} = t_p + \triangle t,\ p = 1, \ldots,$.

5. Numerical results.

5.1. Single grain. We begin by demonstrating the accuracy of the scheme using a single grain simulation. First, we show results related to curvature and its effect on accuracy. For that purpose, we take a circular grain for which an analytic solution exists. The rate of area change must be constant for this case. In Table 5.1 we monitor the short term evolution ($t = 1$, 400 time steps) of a discrete circle of radius 1. The circle is discretized by n uniformly distributed points, $n = 4, 8, 16, 32, 64, 128$, and the distance between the neighboring points is h_n, $h_{2n}/h_n \approx 2$. The measured parameter is the area change $\delta A_n = A_n(t = 0) - A_n(t = 1)$. (The time interval is short enough to assure that the change in h during the evolution is insignificant and can be ignored.) The area change depends on the accuracy of discretization, h_n. To determine the order of accuracy, we computed successive differences $\delta A_{2n}(t = 1) - \delta A_n(t = 1)$. The results clearly show a quadratic dependence of the error on the mesh size.

In Table 5.2 we monitor a short term evolution (= 0.5, 50 unit time steps) of discrete circles with different radii r, $r = 0.25, 0.5, 1.0, 2.0, 4.0, 8.0$. The circles are discretized by n uniformly distributed points; each time n is chosen to provide the same discretization size h. The measured parameter is the area change $\delta A_r = A_r(t = 0) - A_r(t = 0.5)$. (The size of the time interval is short enough to assure that the changes in r during the evolution are insignificant and can be ignored.) The area change depends on the curvature of the circle, $\kappa = 1/r$. To determine the order of accuracy we computed successive differences, $\delta A_{2r} - \delta A_r$. The results clearly show a quadratic dependence of the error on the curvature.

In Figure 5.1 we represent the accuracy in triple junction evolution as a function of the mesh size, h. A single triple junction was considered. The incoming arcs at the initial configurations meet at angles $2\pi/3$; the grain boundaries near the triple junction were straight lines, continued later by $\mathbf{C}^2$ curves. The results shown in the figure are ratios $\|x^h(t) - x^{2h}(t)\|/\|x^{2h}(t) - x^{4h}(t)\|$, where $x(t)$ is the x coordinate of

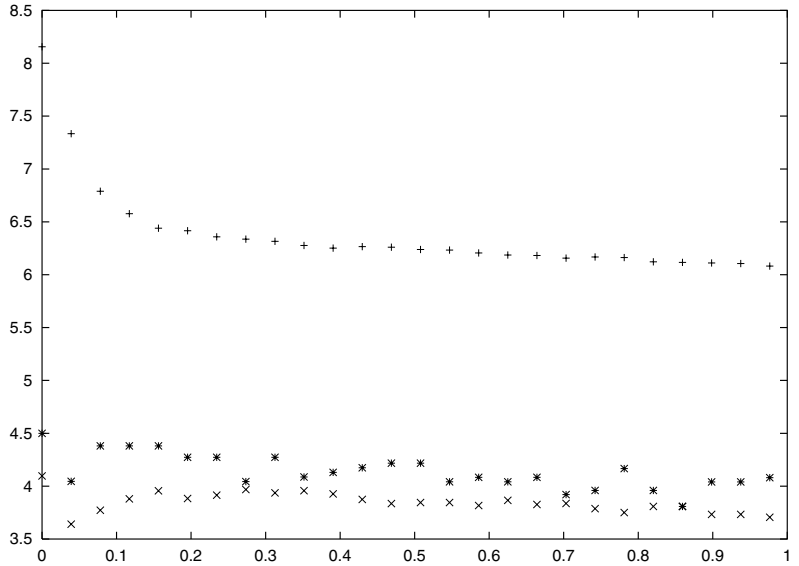


FIG. 5.1. *Triple junction: a mesh size dependence.* $\|x^h(t) - x^{2h}(t)\|/\|x^{2h}(t) - x^{4h}(t)\|$ *as a function of time. A value of* 4 *indicates a second order convergence.*

TABLE 5.3
The $(n-6)$ *rule. Verification for a single grain.*

h	$\delta A_3/(3-6)$	$\delta A_4/(4-6)$	$\delta A_7/(7-6)$
0.1	0.0213992	0.0182787	0.0163595
0.05	0.0184977	0.0148457	0.0146360
0.025	0.0130660	0.0120707	0.0120725
0.0125	0.0116220	0.0112152	0.0112345

the triple junction at time t. (If $\|x^h(t) - x^{2h}(t)\| = Ch^p$, then these ratios converge as $h \to 0$ to 2^p.) We have tracked the evolution of a triple junction for different mesh sizes $h, h = 0.1, 0.05, 0.025, 0.0125, 0.00625$. The top dotted line is for $h = 0.25$, the bottom one is for $h = 0.125$, and the middle one is for $h = 0.00625$. The results in Figure 5.1 clearly show quadratic ($p = 2$) convergence.

In Table 5.3 we verify the $n - 6$ rule. We present results for individual grains with different number of grain boundaries n, $n = 3, 4, 7$, and different mesh sizes $h, h = 0.1, 0.05, 0.025, 0.0125$. In this experiment grain boundary grid points are subject to redistribution, in order to keep the distance between neighboring grid points approximately constant. This is done in order to keep the time step as well as the number of grid points under control. The results show that as h decreases, the quantity $\delta A_n/(n-6)$ approaches a constant independent of n. The table does not show exactly $O(h^2)$ convergence. This is due to slightly different initial configurations for different values of h.

6. Simulation of a bamboo structure. In this section we discuss simulations with isotropic grain boundary energy and mobility, $\sigma = \mu = 1$. The initial configurations for these simulations were obtained using a Voronoi algorithm. Grain boundary end points at the border of the computational domain are held fixed to represent the

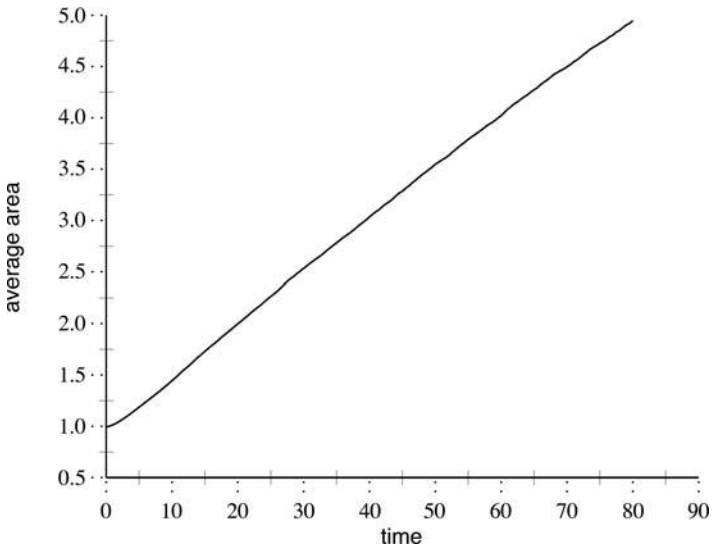


FIG. 6.1. *The average area evolution.*

presence of typical defect structures. Numerical simulations were performed to study basic properties of grain growth and stability of simple distribution functions.

As in any numerical model, to verify its reliability we would like to verify that the numerical simulations show the results predicted by, for instance, analytical considerations. Some analytical results are suggested for two-dimensional grain growth in an isotropic setting. One of them is a linearity, as a function of time, of the average area. The average area, $\langle a \rangle$, is calculated as

$$\langle a \rangle = A/N(t), \tag{6.1}$$

where A is the total area and is time independent and $N(t)$ is the total number of grains at time t. As Figure 6.1 shows, after a short relaxation time, a linear behavior prevails.

Another test to validate our simulations was the $(n-6)$ rule for a network of grains. Figure 6.2 shows the result of area change for each class of grains, e.g., grains with the same number of grain boundaries. In the figure the rates of area changes δA_n were calculated as an average area change in grains that had n grain boundaries at the beginning and the end of the statistical time interval $[t, t+0.5]$. The results are almost indistinguishable for different values of t. The $(n-6)$ rule is clearly obeyed for the network.

While the configuration does not converge to a steady state, certain distribution functions reach equilibrium, particularly the relative grain area distribution and number of edges distribution function. We define $g(n,t)$ and $f(\xi,t)$ as follows: $g(n,t)$ is the number of grains with n grain boundaries at time t, and $f(\xi,t)d\xi$ is the number of grains with relative area in $x \in [\xi, \xi+d\xi]$ at time t, and $\xi = a/\langle a \rangle$. The corresponding distribution functions are

$$G(n,t) = g(n,t)/N(t) \to G_0(n) \quad \text{as } t \text{ grows}$$

and

$$F(\xi,t) = f(\xi,t)/N(t) \to F_0(\overline{a}) \quad \text{as } t \text{ grows},$$

where $N(t)$ is the total number of grains at time t.

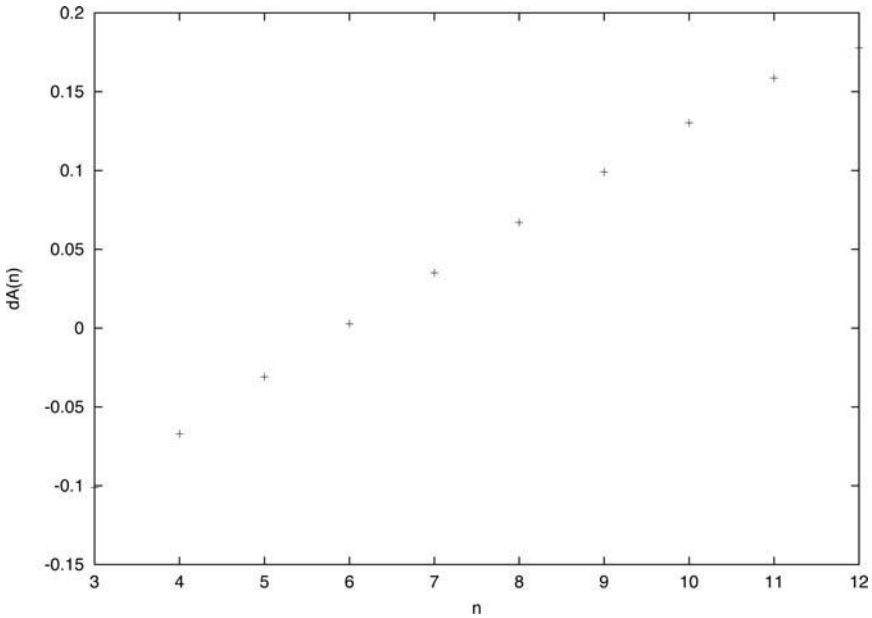


FIG. 6.2. *Verification of the* $(n-6)$ *rule for a network of grains.*

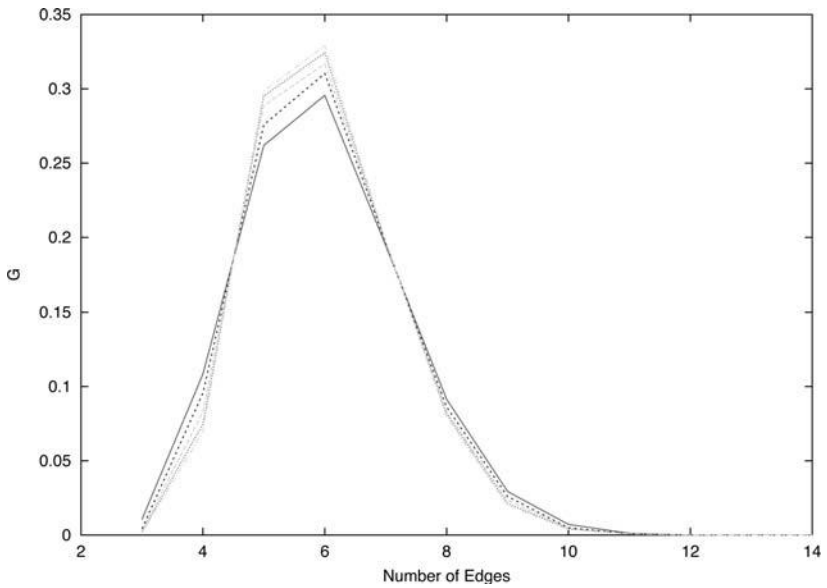


FIG. 6.3. *Evolution: the number of grain boundaries distribution function.*

Numerical studies shows that both $F(\xi, t)$ and $G(n, t)$ reach equilibrium.

In Figure 6.3, $G(n, t)$ is displayed at times $t = 0$ (initial configuration, the graph with lowest peak), $t = 2$, $t = 3$, $t = 4$, and $t = 5$. In Figure 6.4, $G(n, t)$ is displayed at times $t = 4, 6, 8, 10$. The curves are almost indistinguishable.

In Figure 6.5, $F(\xi, t)$ is displayed at times $t = 0$ (initial configuration, the graph with highest peak), $t = 1$ (graph with second highest peak), $t = 2$ (graph with next to

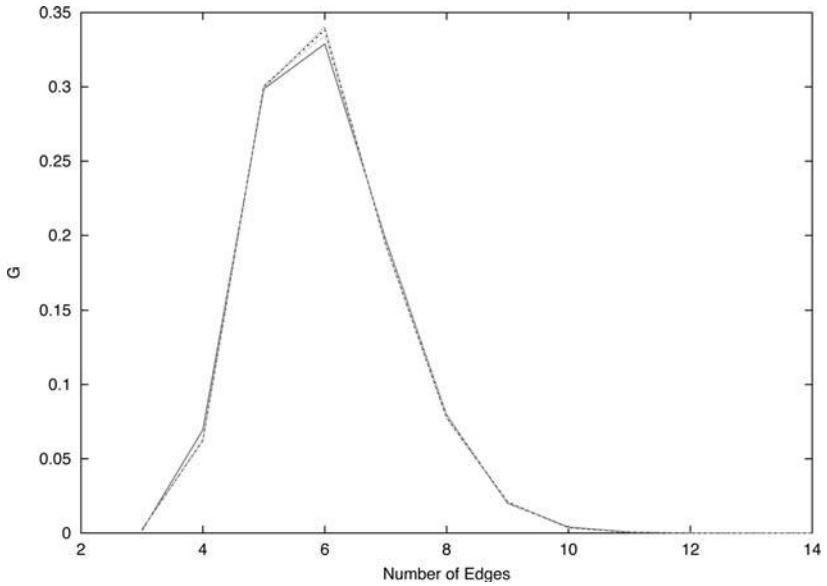


FIG. 6.4. *Steady state: the number of grain boundaries distribution function.*

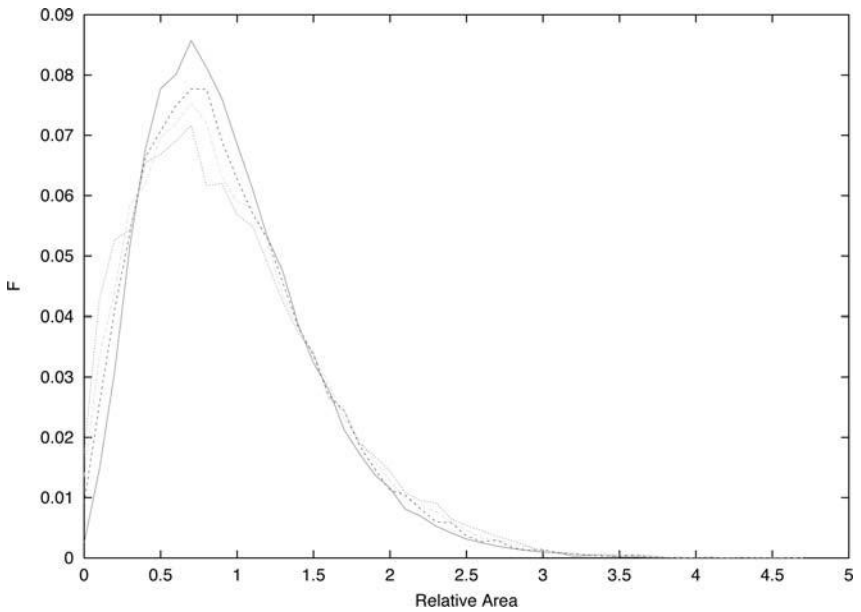


FIG. 6.5. *Evolution: the relative area distribution function.*

lowest peak), and $t = 4$ (the graph with lowest peak). In Figure 6.6 the relative area distribution function is displayed at times $t = 4, 6, 8, 10$, and the curves are almost indistinguishable.

These results demonstrate the convergence toward equilibrium of these distribution functions. This agrees with other numerical results such as MC simulations. The question of agreement with experiment is problematic [4], [5].

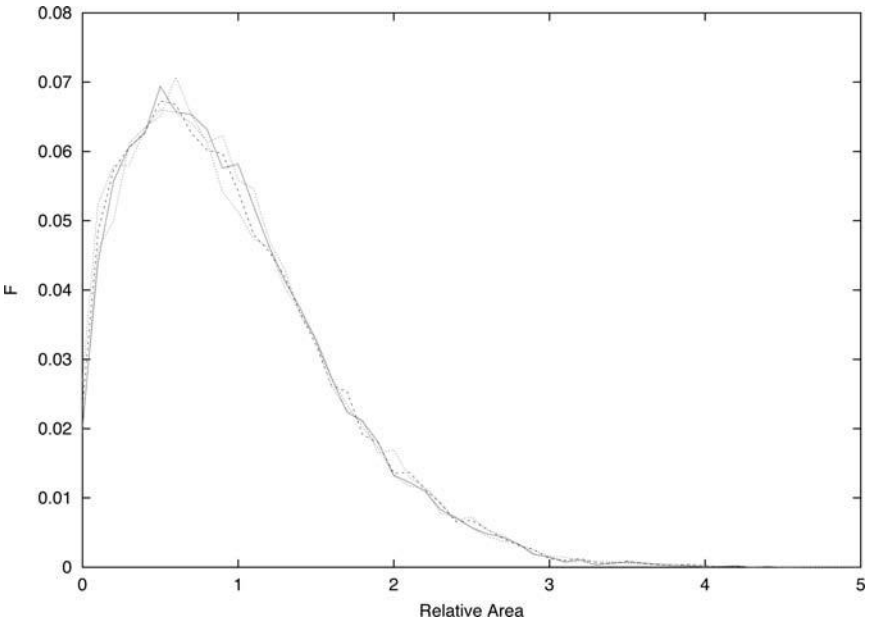


FIG. 6.6. *Steady state: the relative area distribution function.*

7. Simulation with anisotropic energies. A few preliminary results about simulations with anisotropic energies are briefly presented here to illustrate the flexibility of the method. This is part of an extensive project presently underway. For a general energy of this nature, the diagnostics presented in the last two sections no longer apply. But in this new context we have the opportunity to examine the development of texture, albeit in a reduced two-dimensional mode. For the present, let us ask for the [illegible], the distribution of grain boundary as a function of lattice misorientation. We consider first a density $\sigma(\alpha)$, which depends only on the lattice misorientation α but not on the normal angle θ. We choose cubic symmetry, which means that $\sigma(\alpha)$ is periodic of period $\pi/2$. For the convenience of the reader, we recall that in this context the Mullins equation and Herring condition take the form

$$v_n = \sigma(\alpha)\kappa \qquad \text{on } \Gamma, \tag{7.1}$$

$$\sum_{l=1}^{3} \sigma(\alpha_l) b_l = 0 \qquad \text{at a triple junction,} \tag{7.2}$$

where the triple junction involves the three curves $\{\Gamma_l\}$ with lattice misorientations $\{\alpha_l\}$. In this situation, the Herring condition reduces to Young's Law,

$$\frac{\sigma(\alpha_1)}{\sin\beta_1} = \frac{\sigma(\alpha_2)}{\sin\beta_2} = \frac{\sigma(\alpha_3)}{\sin\beta_3}, \tag{7.3}$$

where β_l is the angle subtended by the arcs opposite Γ_l. We now execute a simulation, starting from about 50,000 grains. When the relative area histogram reaches stationarity, we form the new grain boundary character distribution, the histogram of relative boundary length with respect to lattice misorientation $\alpha, -\pi/2 \le \alpha \le \pi/2$.

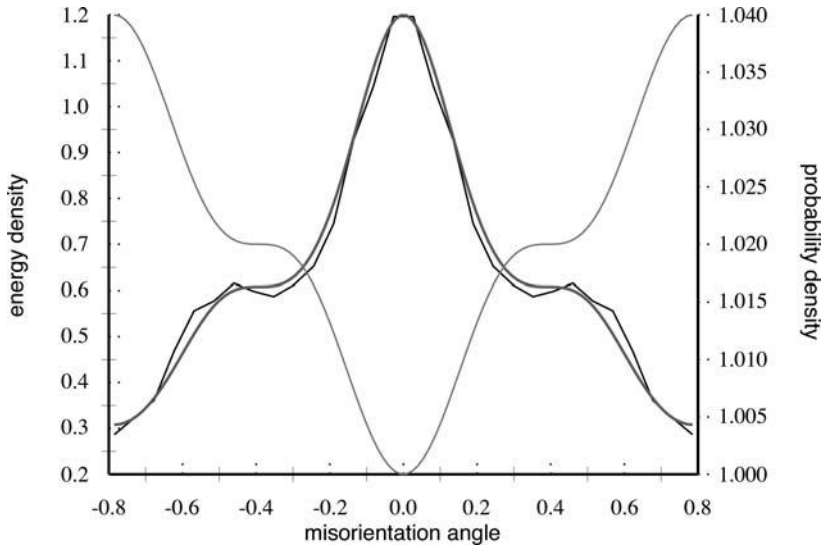


FIG. 7.1. *A simple anisotropic energy density σ,* (7.4)*, graph with minimum at zero with scale on left, and the computed grain boundary character density in graph with three maxima with Boltzmann comparison graph with a single maximum, scale on right.*

In Figure 7.1 are the resulting plots for

$$\sigma(\alpha) \;=\; 1 + \frac{\epsilon}{2}(1 - \cos^3(4\alpha)), \qquad \epsilon = 0.02. \tag{7.4}$$

This σ is very regular, with equal minima at multiples of $\pi/2$ and an inflection point between its maxima and its minima, as shown in Figure 7.1. The result has been smoothly interpolated and shows a distribution f which is compared with a Boltzmann distribution for the energy σ. This result has been found for simulations of (7.1), (7.2) for all choices of $\sigma(\alpha)$. It confirms that the thermodynamic theory of Mullins and Herring admits the statistic and thus ought to be observable as well in experiment; for a preliminary discussion, see [23]. As a check, we also executed the simulation with

$$v_n = \sigma(\alpha)\kappa \qquad \text{on } \Gamma, \tag{7.5}$$

$$\sum_{l=1}^{3} b_l = 0 \qquad \text{at a triple junction,} \tag{7.6}$$

which means that triple junctions meet at angles of $2\pi/3$. In this case the realized grain boundary character distribution is random, as it should be. It is extremely important to keep in mind that this demonstrates conclusively the role of the Herring condition in the theory.

To conclude we present the result of the simulation of a network where the energy density depends on both normal angle θ and lattice misorientation α. The simplest such energy densities consistent with the thermodynamical requirement of frame indifference may be constructed from σ of (7.4) by the expression

$$\begin{aligned} \hat{\sigma}(\theta, \alpha) &= \frac{1}{2}(\sigma(\theta - \omega) + \sigma(\theta - \omega')) \\ &= \frac{1}{2}(\sigma(\theta - \omega) + \sigma(\theta - \alpha - \omega)), \quad \alpha = \omega - \omega', \end{aligned} \tag{7.7}$$

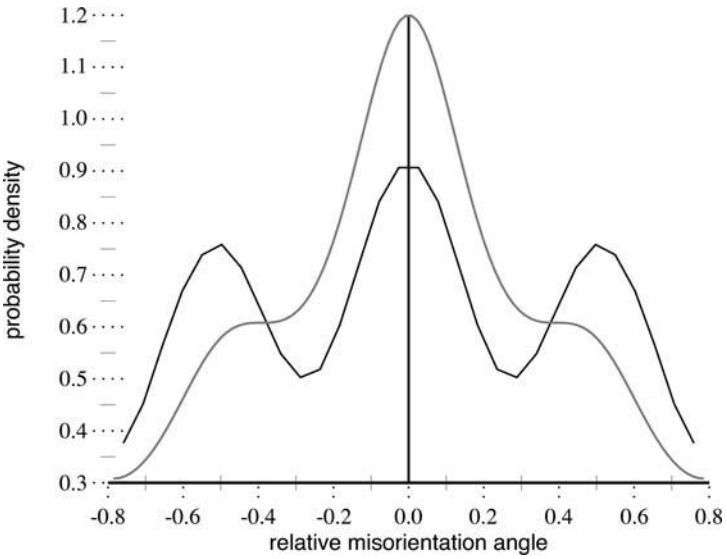


FIG. 7.2. *Histogram f of grain boundary character density (graph with three peaks) compared with Boltzmann distribution (graph with a single peak) for the energy $\hat{\sigma} = \hat{\sigma}(\theta, \alpha)$; see* (7.7).

where ω and ω' are the orientations of the grains neighboring the curve. This represents the energy of the interface as the sum of the energies of the two free interfaces which comprise it, a reasonable assumption in view of recent work by Saylor, Morawiec, and Rohrer [36]. The realized grain boundary energy distribution is shown in Figure 7.2 (three peak graph) and clearly is not a Boltzmann distribution.

8. Conclusion. In this paper we have exhibited a consistent variational approach to the mesoscale simulation of grain growth in two dimensions, and illustrated its application to a "bamboo structure," a system with constant energy and mobility, and a sample "fiber texture," where anisotropy is present. The method accurately computes curvature-driven growth with the Herring condition imposed at triple junctions. The issues we had to resolve to accomplish this include the correct discretization scheme and the treatment of critical events, and will arise in three-dimensional simulations. An important feature of the mesoscale approach is that it respects the thermodynamic formulation given by Mullins and Herring. The reduced dimensionality of its data structure permits simulation of large scale systems, indeed, systems sufficiently larg e that we may inquire about their statistics. We found that the statistics produced by our simulation are very robust. The last section, where we described sample simulations with anisotropic energy densities, illustrates the flexibility of our formulation and shows how it can be employed to discover the grain boundary character distribution, a new statistic with significant promise for predicting material properties. of We hope that the advantages will become even more pronounced for three-dimensional grain growth, which is work in progress, where the description is reduced to evolution of two-dimensional surfaces.

Future work can exploit this method to investigate interface dominated properties, for example, questions about anisotropy, texture, and abnormal grain growth, as mentioned in the Introduction. As a first step previewed in section 7, we are exploring the role of anisotropy; cf. also [9], [23].

Appendix: Pseudocode.

Initialize(N,h){
while $N_k(t) > 0.3N_k(0)$ **do**
AdvanceInteriorNodes($\{\Gamma^k\}$)
AdvanceTripleJunctions($(\{\Gamma^k\},\{z^m\})$)
ReGrid($\{\Gamma^k\}$)
EliminateSmallGrains($(\{G^n\},\{\Gamma^k\},\{z^m\})$)
EliminateSmallGrainBoundaries($(\{\Gamma^k\},\{z^m\})$)
AdvanceQuadrupleJunctions($(\{\Gamma^k\},\{z^m\})$)
end
AdvanceInteriorNodes($\{\Gamma^k\}$){
for $k = 1,\dots,K$ **do**
for $j = 2,\dots,N_k - 1$ **do**
$v^k_{j,l} = \frac{2\mu^k}{|x^k_{j+1,l} - x^k_{j-1,l}|}(T^k_{j+1/2,l} - T^k_{j-1/2,l})$
$x^k_{j,l+1} = x^k_{j,l} + \triangle t_l \; v^k_{j,l}$
end
end
}
AdvanceTripleJunctions($(\{\Gamma^k\},\{z^m\})$){
for $m = 1,\dots,M$ **do**
if $|\Gamma(z^m) = 3$ **do**
`Solve` $\psi(z^m_{l+1}) = 0$
`Set` $x^{k_i(m)}_{j',l+1} = z^m_{l+1}$ `if` $x^{k_i(m)}_{j',l}(m) = z^m_l$
end
end
}
AdvanceQuadrupleJunctions($(\{\Gamma^k\},\{z^m\})$){
for $m = 1,\dots,M$ **do**
if $|\Gamma(z^m)| = 4$ **do**
`Set` $\Gamma(z^m) = \{\Gamma^{l_1},\Gamma^{l_2},\Gamma^{l_3},\Gamma^{l_4}\}$
`such that` $G(\Gamma^{l_1})\bigcap G(\Gamma^{l_3}) = \varnothing$, $G(\Gamma^{l_2})\bigcap G(\Gamma^{l_4}) = \varnothing$
`Define grain boundary interior points closest to the junction:`
$x^{l_1} \in \Gamma^{l_1}, x^{l_2} \in \Gamma^{l_2}, x^{l_3} \in \Gamma^{l_3}, x^{l_4} \in \Gamma^{l_4}$
`Configuration 1: Set` $\Gamma_1 = \{\Gamma^{l_1},\Gamma^{l_2}\}$ $\Gamma_2 = \{\Gamma^{l_3},\Gamma^{l_4}\}$
`Define vectors` $j_1 = (\frac{x^{l_1}-z}{|x^{l_1}-z|} + \frac{x^{l_2}-z}{|x^{l_2}-z|})$, $j_2 = (\frac{x^{l_3}-z}{|x^{l_3}-z|} + \frac{x^{l_4}-z}{|x^{l_4}-z|})$
`Configuration 2: Set` $\Gamma_3 = \{\Gamma^{l_1},\Gamma^{l_4}\}, \Gamma_4 = \{\Gamma^{l_2},\Gamma^{l_3}\}$
`Define vectors` $j_3 = (\frac{x^{l_1}-z}{|x^{l_1}-z|} + \frac{x^{l_4}-z}{|x^{l_4}-z|})$, $j_4 = (\frac{x^{l_2}-z}{|x^{l_2}-z|} + \frac{x^{l_3}-z}{|x^{l_3}-z|})$
`Create 2 triple junctions:` z^{M+1}, z^{M+2}
`Create a grain boundary:` Γ^{K+1}
if $|j_1 - j_2| > |j_3 - j_4|$ **do**
$z^{M+1} = z^m + \frac{h}{8}j_1$, $\Gamma(z^{M+1}) = \{\Gamma^{K+1}\}\bigcup\Gamma_1$
$z^{M+2} = z^m + \frac{h}{8}j_2$, $\Gamma(z^{M+2}) = \{\Gamma^{K+1}\}\bigcup\Gamma_1$
end
else
$z^{M+1} = z^m + \frac{h}{8}j_3$, $\Gamma(z^{M+1}) = \{\Gamma^{K+1}\}\bigcup\Gamma_3$
$z^{M+2} = z^m + \frac{h}{8}j_4$, $\Gamma(z^{M+2}) = \{\Gamma^{K+1}\}\bigcup\Gamma_4$
end
$\Gamma^{K+1} = \{z^{M+1}, \frac{1}{2}(z^{M+1} + z^{M+2}), z^{M+2}\}$

```
      Update Λ = Λ ⋃ {z^{M+1}, z^{M+2}} ∖ {z^m}
      Update Γ = Γ ⋃ {Γ^{K+1}}
      end
  end
}
ReGrid({Γ^k}){
  for k = 1, ..., N
      L_k = Σ_{j=2}^{N_k} |x_j^k − x_{j−1}^k|
      if L_k/N_k > (3/2)h  do
          w_j ← x_j^k, j = 1, ..., N_k,
          N_k ← (N_k + 1)/2
          x_j^k ← w_{2j−1}, j = 1, ..., N_k,
      end
      else if L_k/N_k < (3/4)h  do
          w_{2j−1} ← x_{j,l+1}^k, j = 1, ..., N_k,
          w_{2j} = (w_{2j−1} + w_{2j+1})/2,  j = 1, N_k − 1
          N_k ← (2N_k − 1)
          x_j^k ← w_j, j = 1, ..., N_k,
      end
      Γ^k = {x_j^k; j = 1, ..., N_k}
}
Initialize(N,h){
      {z^m} ← RandomPoints(N)
      ConstructVoronoi({z^m} )
      GenerateGrid()
}
EliminateSmallGrains(({G^l}, {Γ^k})) {
  for n = 1, N
      if |G_n| < ε do
          Let Γ_n = Γ(G_n), Z_n = Z(G_n), Γ̄_n = {Γ ∈ Γ : z_i(Γ) ∈ Z_n, Γ ∉ Γ_n}
          if  |Γ_n| = 3 Introduce a triple junction: z^{M+1} = (1/3) Σ_{z^m ∈ Z_n} z^m
          if  |Γ_n| = 4 Introduce a quadruple junction: z^{M+1} = (1/4) Σ_{z^m ∈ Z_n} z^m
          for Γ ∈ Γ̄_n do
              for i = 1, 2
              if z_i(Γ) ∈ Z_n do z_i(Γ) = z^{M+1}
              end
          end
      end
      Update Γ = Γ ∖ Γ_n, Λ = Λ ⋃ {z^{M+1}} ∖ Z_n, Ω = Ω ∖ G_n
  end
}
CutShortGrainBoundary({Γ^k}) {
  for k = 1, ..., K
      if |Γ^k| < ε_Γ do
          Let Z(Γ^k) = {z^{m_1}, z^{m_2}}; Γ(z^{m_1}) = {Γ^k, Γ^{l_1}, Γ^{l_2}}; Γ(z^{m_2}) = {Γ^k, Γ^{l_3}, Γ^{l_4}}
          Create a quadruple junction z^{M+1} = (1/2)(z^1 + z^2)
          for i = 0, 1
              if z_i(Γ^{l_1}) = z^{m_1} then z_i(Γ^{l_1}) = z^{M+1}
              if z_i(Γ^{l_2}) = z^{m_1} then z_i(Γ^{l_2}) = z^{M+1}
```

```
                if z_i(Γ^{l_3}) = z^{m_2} then z_i(Γ^{l_3}) = z^{M+1}
                if z_i(Γ^{l_4}) = z^{m_2} then z_i(Γ^{l_4}) = z^{M+1}
            end
        Update Γ = Γ \ Γ^k, Λ = Λ ⋃ {z^{M+1}} \ {z^{m_1}, z^{m_2}}
        end
    end
}
```

Acknowledgements. We wish to acknowledge the advice and support of K. Barmak, A. D. Rollett, G. Rohrer, C. Liu, and Peng Yu. This work was begun while I. Livshits was an MRSEC postdoctoral fellow at Carnegie Mellon University.

REFERENCES

[1] B. L. Adams, S. Ta'asan, D. Kinderlehrer, I. Livshits, D. Mason, C-T. Wu, W. W. Mullins, G. S. Rohrer, A. D. Rollet, D. Saylor, *Extracting grain boundary and surface energy from measurement of triple junction geometry*, Interface Sci., 7 (1999), pp. 321–338.

[2] M. P. Anderson, D. J. Srolovitz, G. S. Grest, and P. S. Sahni, *Computer simulation of grain growth* I. *Kinetics*, Acta Metallurgica, 32 (1985), pp. 783–791.

[3] M. P. Anderson, G. S. Grest, and D. J. Srolovitz, *Computer simulation of grain growth in three dimensions*, Phil. Mag. B, 59 (1989), pp. 293–329.

[4] K. Barmak, W. Archibald, J, Kim, C.-S. Kim, A. D. Rollett, G. S. Rohrer, S. Ta'asan, S., and D. Kinderlehrer, *Grain boundary energy and grain growth in highly-textured Al films and foils: Experiment and simulation*, Mater. Sci. Forum Ser. 495–497 (2005), pp. 1255–1260.

[5] K. Barmak, D. Kinderlehrer, I. Livshits, and S. Ta'asan, *Remarks on a multiscale approach to grain growth in polycrystals*, Progr. Nonlinear Differential Equations Appl., 68 (2006), pp. 1–11.

[6] J. Bragard, A. Karma, H. Y. H. Lee, and M. Plapp, *Linking phase-field and atomistic simulations to model dendritic solidification in highly undercooled melts*, Interface Sci., 10 (2002), pp. 121–136.

[7] L. Q. Chen, *A novel computer simulation technique for modeling grain growth*, Scr. Metal. Mater., 32 (1995), pp. 115–120.

[8] F. Cleri, S. R. Phillpot, and D. Wolf, *Atomistic simulations of intergranular fracture in symmetric-tilt grain boundaries*, Interface Sci., 7 (1999), pp. 45–55.

[9] M. C. Demirel, A. Kuprat, D. C. George, G. Straub, and A. D Rollett, *Linking experimental characterization and computational modeling of grain growth in Al-foil*, Interface Sci., 10 (2002), pp. 137–142.

[10] H. Flyvbjerg, *Model for coarsening froths and foams*, Phys. Rev. E, 47 (1993), pp. 4037–4054.

[11] V. E. Fradkov, M. E. Glicksman, M. Palmer, and K. Rajan, *Topological events in two-dimensional grain growth*, Acta Metall. Mater., 42 (1994), pp. 2719–2727.

[12] V. E. Fradkov and D. Udler, *Two dimensional normal grain growth: Topological aspects*, Adv. Phys., 43 (1994), pp. 739–789.

[13] V. E. Fradkov, *Main regularities of 2-D normal growth,* Mater. Sci. Forum, 94–96 (1992), pp. 269–274.

[14] H. J. Frost, C. V. Thompson, and D. T Walton, *Simulation of thin film grain structures:* I. *Grain growth stagnation*, Acta Metall. Mater., 38 (1990), pp. 1455–1462.

[15] H. J. Frost, C. V. Thompson, C. L. Howe, and J. Whang, *A two dimensional computer simulation of capillarity-driven grain growth: Preliminary results*, Scripta Met., 22 (1988), pp. 62–70.

[16] A. J. Haslam, S. R. Phillpot, D. Wolf, D. Moldovan, and H. Gleiter, *Mechanisms of grain growth in nanocrystalline fcc metals by molecular dynamic simulation*, Materials Science and Engineering A (Switzerland), 318 (2001), pp. 293–312.

[17] C. Herring, *Surface Tension as a Motivation for Sintering*, in The Physics of Powder Metallurgy, W. E. Kingston, ed., MacGraw–Hill, New York, 1951, pp. 143–179.

[18] C. Herring, *The use of classical macroscopic concepts in surface energy problems*, in Structure and Properties of Solid Surfaces, R. Gomer and C. Smith, eds., University of Chicago Press, Chicago, 1952, pp. 5–81.

[19] P. Keblinski, D. Wolf, and S. R. Phillpot, *Molecular-dynamics simulations of grain boundary diffusion creep*, Interface Sci., 6 (1998), pp. 205–212.

[20] D. Kinderlehrer, J. Lee, I. Livshits, A. D. Rollett, and S. Ta'asan, *Mesoscale simulation of grain growth, Recrystallization and grain growth*, Mater. Sci. Forum, 467–470 (2004), pp. 1057–1062

[21] D. Kinderlehrer, J. Lee, I. Livshits, and S. Ta'asan, *Mesoscale simulation of grain growth*, in Continuum Scale Simulation of Engineering Materials, D. Raabe, L.-Q. Chen, F. Barlat, and F. Roters, eds., Wiley-VCH Verlag, Weinheim, Germany, 2004, Chapt. 16, pp. 361–372.

[22] D. Kinderlehrer and C. Liu, *Evolution of grain boundaries*, Math. Models Methods Appl. Math., 11 (2001), pp. 713–729.

[23] D. Kinderlehrer, I. Livshits, G. S. Rohrer, S. Ta'asan, and P. Yu, *Mesoscale evolution of the grain boundary character distribution, Recrystallization and grain growth*, Mater. Sci. Forum, 467–470 (2004), pp. 1063–1068.

[24] D. Kinderlehrer, F. Manolache, I. Livshits, A. D. Rollett, and S. Ta'asan, *An approach to the mesoscale simulation of grain growth*, Mat. Res. Soc. Proc. 652, M. Aindow, M. D. Asta, M. V. Glazow, D. L. Medlin, A. D. Rollett, and M. Zaiser, eds., 2001, Y1.5.1–Y1.5.11.

[25] R. Kobayashi, J. A. Warren, and W. C. Carter, *Vector-valued phase field model for crystallization and grain boundary formation*, Phys. D, 119 (1998), pp. 415–423.

[26] C. E. Krill and L. Q. Chen, *Computer simulation of 3-D grain growth using a phase-field model*, Acta Mater., 50 (1998), pp. 3057–3073.

[27] A. Kuprat, *Modeling microstructure evolution using gradient-weighted moving finite elements*, SIAM J. Sci. Comput, 22 (2000), pp. 535–560.

[28] K. Mehnert and P. Klimanek, *Monte Carlo simulation of grain growth in textured metals*, Scripta Mater., 30 (1996), pp. 699–704.

[29] M. I. Mendelev and D. J. Srolovitz, *Co-segregation effects on boundary migration*, Interface Sci., 10 (2002), pp. 191–199.

[30] M. Morhac and E. Morhacova, *Monte Carlo simulation algorithms of grain growth in polycrystalline materials*, Cryst. Res. Technol., 35 (2000), pp. 117–128.

[31] W. W. Mullins, *Two-dimensional motion of idealized grain boundaries*, J. Appl. Phys., 27 (1956), pp. 900–904.

[32] W. W. Mullins, *Solid surface morphologies governed by capillarity*, in Metal Surfaces: Structure, Energetics, and Kinetics, American Society of Metals, Metals Park, Ohio, 1963, pp. 17–66.

[33] S. R. Phillpot, P. Keblinski, D. Wolf, and F. Cleri, *Synthesis and characterization of a polycrystalline ionic thin film by large-scale molecular-dynamics simulation*, Interface Sci., 7 (1999), pp. 15–31.

[34] D. Raabe, *Cellular, lattice gas, and Boltzmann automata, in Continuum Scale Simulation of Engineering Materials*, (Raabe, D. et al., eds), Wiley-VCH Verlag, Weinheim, Chapt. 3, 2004, pp. 57–76.

[35] A. D. Rollett and P. Manohar, *The Monte-Carlo method*, in Continuum Scale Simulation of Engineering Materials, D. Raabe, L.-Q. Chen, F. Barlat, and F. Roters, eds., Wiley-VCH Verlag, Weinheim, Germany, 2004, Chapt. 4, pp. 77–114.

[36] D. M. Saylor, A. Morawiec, and G. S. Rohrer, *The relative free energies of grain boundaries in magnesia as a function of five macroscopic parameters*, Acta Mat., 51 (2003), pp. 3675–3686.

[37] B. Schoenfelder, D. Wolf, and S. R. Phillpot, *Molecular-dynamics method for simulation of grain boundary migration*, Interface Sci., 5 (1997), pp. 245–262.

[38] J. Von Neumann, *Metal Interfaces*, American Society for Metals, Cleveland, OH, 1952, p. 108–110.

[39] M. Upmanyu, R. W. Smith, and D. J. Srolovitz, *Atomistic simulation of curvature driven grain boundary migration*, Interface Sci., 6 (1998), pp. 41–58.

[40] M. Upmanyu, D. J. Srolovitz, L. S. Shvindlerman, and G. Gottstein, *Triple junction mobility: A molecular dynamics study*, Interface Sci., 7 (1999), pp. 1405–142.

[41] M. Upmanyu, D. J. Srolovitz, L. S. Shvindlerman, and G. Gottstein, *Misorientation dependence of intrinistic grain boundary mobility: Simulation and experiment*, Acta Mater., 47 (1999), pp. 3901–3914.

[42] M. Upmanyu, G. N. Hassold, A. Kazaryan, E. A. Holm, Y. Wang, B. Patton, and D. J. Srolovitz, *Boundary mobility and energy anisotropy effects on microstructural evolution during grain growth*, Interface Sci., 10 (2002), pp. 201–216.

[43] D. Weaire and J. P. Kermode, *Computer simulation of a two-dimensional soap froth* I. *Method and motivation*, Phil. Mag. B, 48 (1983), pp. 245–259.

[44] D. Weaire and J. P. Kermode, *Computer simulation of a two-dimensional soap froth* II. *Analysis of results*, Phil. Mag. B, 50 (1984), pp. 379–395.

[45] D. Weygand, Y. Brechet, and J. Lepinoux, *Mechanisms and kinetics of recrystallisation: A two dimensional vertex dynamics simulation*, Interface Sci., 9 (2001), pp. 311–317.

SIAM J. SCI. COMPUT.
Vol. 28, No. 5, pp. 1716–1729
© 2006 Society for Industrial and Applied Mathematics

A PSEUDOSPECTRAL FICTITIOUS POINT METHOD FOR HIGH ORDER INITIAL-BOUNDARY VALUE PROBLEMS*

BENGT FORNBERG†

Abstract. When pseudospectral approximations are used for space derivatives, one often encounters spurious eigenvalues. These can lead to severe time stepping difficulties for PDEs. This is especially the case for equations with high order derivatives in space, requiring multiple conditions at one or both boundaries. We note here that a very simple-to-implement fictitious point approach circumvents most of these difficulties. The new approach is tested on the Kuramoto–Sivashinsky equation and on a dispersive linear PDE featuring a time-space corner singularity.

Key words. spurious eigenvalues, PDE, pseudospectral method, Kuramoto–Sivashinsky equation, time-space corner singularities

AMS subject classification. 65M70

DOI. 10.1137/040611252

1. Introduction.

PDEs with high derivatives in space typically require multiple conditions at one or more of its boundaries, as noted in Table 1.1. Implementation of one Dirichlet condition at a boundary is straightforward when using a Chebyshev-type spectral method. This paper discusses how to best implement additional conditions.

The third order linear initial-boundary value (IBV) problem

$$\begin{cases} \text{PDE} & u_t = u_{xxx}, \\ \text{IC} & u(x,0) = f(x), \\ \text{BC} & u(\pm L, t) = 0,\ u_x(-L,t) = 0 \end{cases} \tag{1.1}$$

can be used to illustrate this topic. A numerical methods-of-lines (MOL) time stepping method for (1.1) will be stable if and only if the spatial differentiation operator is such that, when instead applied to the eigenvalue problem

$$\begin{cases} \text{ODE} & u_{xxx} = \lambda u, \\ \text{BC} & u(\pm L) = 0,\ u_x(-L) = 0, \end{cases} \tag{1.2}$$

the quantity $\xi = \lambda \Delta t$ for all the eigenvalues λ fall within the time stepping method's stability domain.

For the problem (1.2), the true eigenvalues are all real and negative. They are given by the roots to

$$e^{3L\lambda^{1/3}} - 2\sin\left(\sqrt{3}L\lambda^{1/3} + \frac{\pi}{6}\right) = 0 \tag{1.3}$$

(omitting $\lambda = 0$), and can be approximated very closely by

$$\lambda_k \approx -\left[\frac{\left(k+\frac{1}{6}\right)\pi}{\sqrt{3}L}\right]^3, \quad k = 1, 2, \ldots. \tag{1.4}$$

*Received by the editors July 7, 2004; accepted for publication (in revised form) April 27, 2006; published electronically October 16, 2006. This work was supported by NSF grants DMS-0309803 and DMS-0611681.

http://www.siam.org/journals/sisc/28-5/61125.html

†Department of Applied Mathematics, University of Colorado, 526 UCB, Boulder, CO 80309 (fornberg@colorado.edu).

TABLE 1.1
Number of conditions needed at each boundary for some time dependent PDEs in order to form well-posed IBV problems.

PDE		Number of conditions needed	
		Left side	Right side
$u_t = u_x$		0	1
$u_t = u_{xx}$		1	1
$u_t = u_{xxx}$		2	1
$u_t = u_{xxxx}$		2	2
$u_t = u_{xxxxx}$		2	3
$u_t + uu_x + u_{xxx} = 0$	KdV	1	2
$u_t + uu_x + u_{xx} + u_{xxxx} = 0$	KS	2	2

The challenge thus lies in approximating (1.2) so that (i) as many of the leading eigenvalues (and eigenvectors) as possible become accurately approximated, and (ii) the remaining (spurious) numerical eigenvalues are all located far out in the left half-plane, so as not to damage either the accuracy or the stability when using the MOL approach with a stiff ODE solver.

When using Galerkin (or tau) spectral methods—working with expansion coefficients rather than with node values as in the present pseudospectral (PS) approach—remedies have been discussed numerous times in the literature, e.g., [15], [23], and the references therein. However, especially in contexts involving variable coefficients and nonlinearities, the PS approach tends to be a lot more convenient.

In the next two sections of this paper, we summarize some approaches for resolving the problem with spurious eigenvalues when using the PS approach. We will note that an approach based on fictitious points (previously introduced for spectral methods in [13], in a slightly different context) is particularly effective, flexible, and simple to implement. In section 4, we discuss the same issues for some more general PDEs, in particular for the Kuramoto–Sivashinsky (KS) equation, and for some linearized versions of it. We note in section 5 that the issue of time-space corner singularities still needs more investigation, especially in the presence of nonlinearities. The example in section 6 illustrates this singularity issue, and also shows that the application of inhomogeneous boundary conditions (BCs) causes no further difficulties. Section 7 contains some concluding remarks.

2. Differentiation matrix (DM) approaches. With the node points denoted by $x_k,\ k = 0, 1, \ldots, n$, and the corresponding function values by u_k, the first derivative values u'_k can be approximated by means of a matrix×vector product

$$\begin{bmatrix} & & \\ & D^1 & \\ & & \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_n \end{bmatrix} = \begin{bmatrix} u'_0 \\ u'_1 \\ \vdots \\ u'_n \end{bmatrix}.$$

Requiring that this formula becomes exact for all polynomials of degree n determines the differentiation matrix D^1 uniquely. This matrix can be computed at a cost of only four operations per entry by means of the explicit formula [27]

$$D^1_{j,k} = \begin{cases} \dfrac{a_j}{a_k(x_j - x_k)} & \text{if } j \neq k, \\ \displaystyle\sum_{\substack{i=0 \\ i\neq k}}^{n} \dfrac{1}{x_k - x_i} & \text{if } j = k, \end{cases} \qquad \text{where } a_k = \prod_{\substack{i=0 \\ i\neq k}}^{n} (x_k - x_i).$$

For higher derivatives, the corresponding DMs can be obtained by immediate matrix multiplication, i.e., $D^p = (D^1)^p$. However, several direct recursions have been found which give the entries of D^p both faster and in a more numerically stable manner. For different methods and/or variations, see, for example, [12], [14], [19], [35], [36], [37], [38]. In all the following discussions, we have used Chebyshev distributed nodes, i.e., $x_k = -L\ \cos\frac{\pi k}{n},\ k = 0, 1, 2, \ldots, n,$ in case of the interval $[-L, L]$. However, the results and conclusions become the same for similar choices, such as Legendre, Gegenbauer, or Jacobi distributions.

The immediate numerical discretization of (1.2) can be written as

$$\text{ODE:} \qquad \begin{bmatrix} & D^3 & \end{bmatrix}_{n+1,n+1} \begin{bmatrix} u \end{bmatrix}_{n+1} = \lambda \begin{bmatrix} u \end{bmatrix}_{n+1}, \tag{2.1}$$

with

$$\text{BC:} \qquad \begin{bmatrix} 1 & 0 & \ldots & 0 & 0 \\ 0 & 0 & \ldots & 0 & 1 \\ d_0 & d_1 & \ldots & d_{n-1} & d_n \end{bmatrix}_{3,n+1} \begin{bmatrix} u \end{bmatrix}_{n+1} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}_3. \tag{2.2}$$

We have here also marked the sizes of the matrices and vectors. The most natural way to incorporate the first two BCs would be to simply remove the first and last rows and columns from D^3 and the first and last elements of u, leading to the eigenvalue problem $\widehat{D^3}\hat{u} = \lambda\hat{u}$ of size $(n-1) \times (n-1)$. Had there been no more BCs (as in the case for second derivatives using one condition at each end), this would have been fully satisfactory. However, we have here also a third BC. Its implementation is much less apparent, and forms the main topic of this present study.

Trefethen notes in [32, page 135] "There are two basic approaches to boundary conditions for spectral collocation methods: (I) [illegible]; or (II) [illegible]." We describe this second approach first, in subsection 2.1. Most recent work has concentrated on the first approach, discussed in subsection 2.2. We introduce then, in section 3, the fictitious point (FP) approach. This is very simple to implement and, as will be shown, lends itself especially well to the time stepping of PDEs with either homogeneous or inhomogeneous BCs.

2.1. Approach II: Elimination of an additional row and column. Merryfield and Shizgal [24] consider the possibility of adding suitable multiples of the last BC in (2.2) to the different rows in the ODE, so that also the second column of D^3 gets eliminated. The second unknown u_1 can, from this BC, be expressed in terms of the remaining unknowns:

$$u_1 = -\frac{1}{d_1} \sum_{i \neq 1} d_i u_i. \tag{2.3}$$

In order not to impose more relations than there are unknowns, we would then omit imposing the ODE at this second grid point location. The eigenvalue problem has then been reduced to size $(n-2) \times (n-2)$. We write this as

$$\widetilde{D_1^3} = \lambda\check{u}, \tag{2.4}$$

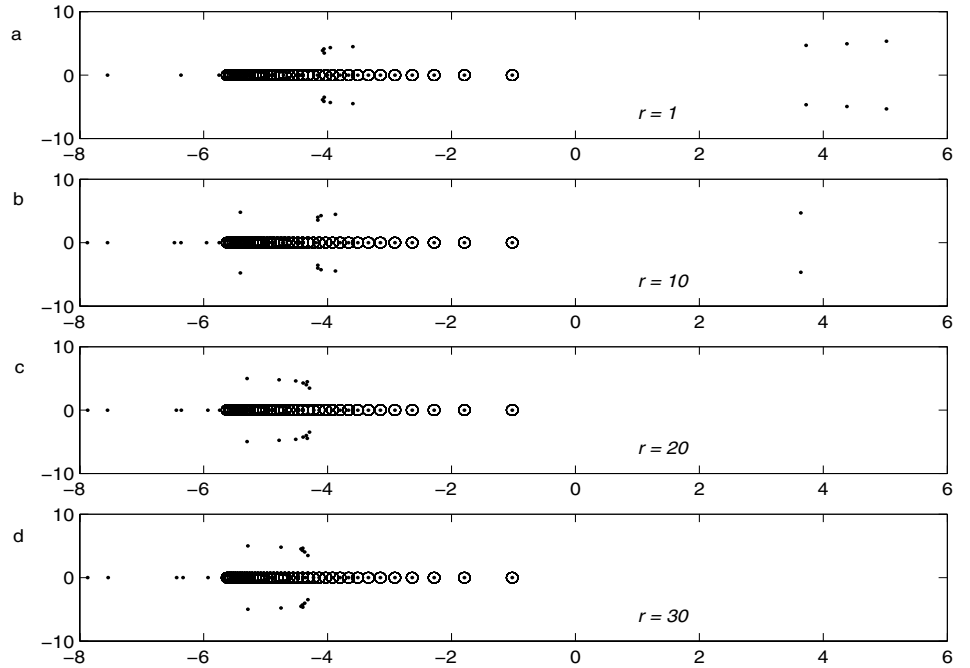


FIG. 2.1. *Complex spectra for the reduced matrix* $\check{D}_1^3$ *in the Mulholland approach, when the first and last row and column and also row and column* r, $r = 1, 10, 20, 30$, *have been eliminated. The axes have been scaled by* $x \to sign(x)\ \log_{10}(1 + |x|)$ *and similarly for* y (*in order to keep signs but compress large arguments*).

where the subscript "1" denotes that the second row and column are also eliminated (recalling that we count rows and columns starting with zero). Figure 2.1(a) (labeled $r = 1$) compares, in the case of $n = 40$ and $L = 1$, the eigenvalue distribution for (2.4) (marked by dots) against the first $n + 1$ analytic eigenvalues (marked by circles). As Merryfield and Shizgal first noted, the presence of several spurious eigenvalues far out in the right half-plane rules out this approach in the context of MOL time stepping.

Mulholland [26] generalized the approach of eliminating row 1 and column 1 to instead eliminating row r and column r, where $1 \le r \le n - 1$. Since the extra boundary condition appears at the left boundary, Merryfield and Shizgal's choice $r = 1$ is what first comes to mind. However, it transpires that the stability situation improves very much when r is increased. Figures 2.1(b), (c), (d) show the results with $r = 10, 20, 30$, respectively. Although the situation with r approaching n might appear satisfactory in the present case, there is something strange about eliminating a node in the Chebyshev grid far away from the boundary at which the extra condition had to be imposed. In fact, the generalization

$$u_r = -\frac{1}{d_r} \sum_{i \ne r} d_i u_i \tag{2.5}$$

of (2.3) becomes problematic because the weights d_i are oscillatory and decay rapidly in magnitude with increasing r. Figure 2.2 shows these in the case of $L = 1$ and $n = 20$. Some coefficients d_i/d_r in (2.5) become much larger than one, and small oscillations in u_i, $i < r$, get strongly amplified in the computed value of u_r. The

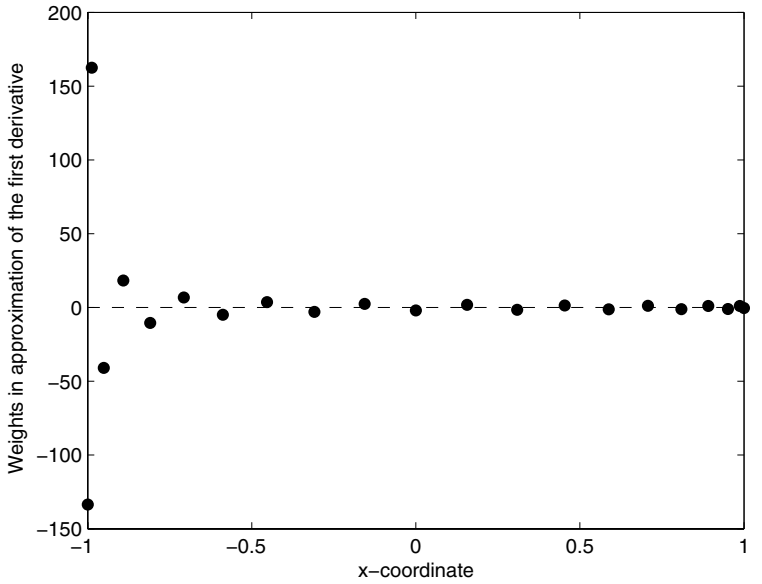


FIG. 2.2. *Weights d_i when the first derivative is approximated at the left boundary of a* 21*-node Chebyshev-spaced grid.*

reason the methodology still works, even as r is increasing, is that the resulting errors in u_r occur close to the right boundary, and the general character of solutions to (1.1) is that all modes are transported towards this right boundary and become completely absorbed there—the faster the higher the mode is. It thus would appear that the Mulholland approach will work only for very special PDEs. In fact, it fails entirely for the KS equation, which we will consider in section 4.

2.2. Approach I: Modify the DM to incorporate the boundary information. Determining the DM for collocation at $x_0,\ x_1, \ldots, x_n$ with a polynomial $p_n(x)$ over $[-1, 1]$ and then enforcing $u(-1) = u(1) = 0$ leads to the same DM as one would get if one considered only polynomials of the form $(x + 1)(x - 1)p_{n-2}(x)$ and collocated only over the internal nodes $x_1, \ldots, x_{n-1}$. In order to also enforce $u_x(-1) = 0$, we can collocate over the interior nodes with a polynomial of the form $(x+1)^2(x-1)p_{n-2}(x)$. This type of approach was taken by Funaro and Heinrichs [16], later independently by Huang and Sloan [18], and again by Heinrichs [17]. In these cases, closed form expressions for the resulting DMs were obtained. Trefethen [32, page 146] describes another approach which, for arbitrary node distributions, gives the desired DM from standard ones in just a few matrix operations. Figure 2.3(a), showing its eigenvalues, should be compared to Figure 2.1. The spurious eigenvalues are located as advantageously as in any of the previous cases from Mulholland's method, but without the need to discard the governing equation at any one of the interior grid points. One of the referees of the present paper proposed the following procedure: with $\underline{u} = [u(-1), u_1, u_2, \ldots, u_{n-1}, u(1)]^T$, form $D^1\underline{u}$, set its first component to $u'(-1)$, and then apply D^2 to get an approximation for u_{xxx} that incorporates the given values of $u(-1)$, $u(1)$, and $u'(-1)$. This procedure, which is particularly simple to implement in case of nonhomogeneous BC, gives results that are only very slightly

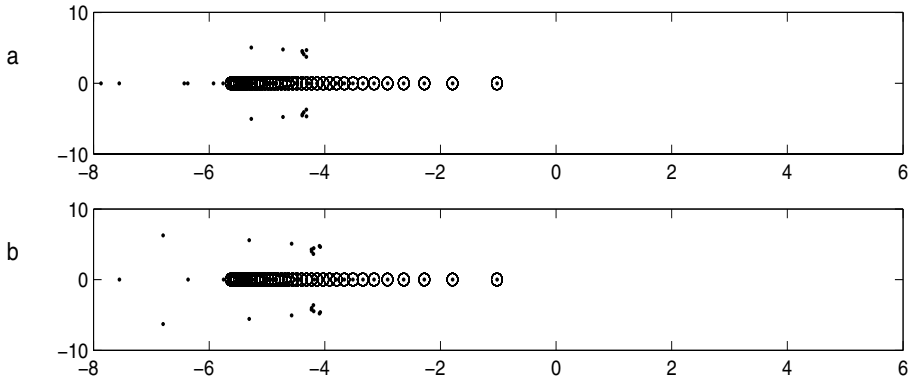


FIG. 2.3. *Complex spectrum for the same test case as in Figure* 2.1. (a) *Collocation with a polynomial of the form* $(x+1)^2(x-1)p_{n-2}(x)$. *This will agree with the FP method described in section* 3. (b) *Last method described in section* 2.2.

different from the method just mentioned (as seen in Figure 2.3(b)).

None of the methods/implementations that have just been mentioned combine all of the following desirable features:

(i) easy implementation;

(ii) easy generalization to nonhomogeneous cases;

(iii) ability to handle Robin-type BCs (involving linear combinations of values and derivatives, in place of values and/or derivatives);

(iv) ability to impose BCs on staggered-type grids (where the actual boundary location does not coincide with a grid point).

3. Fictitious point approach. Although for simplicity, our description will not be the most general, the FP approach will be seen to meet all the four criteria that were just noted.

Since we in our model problem (1.2) have one more BC than can be imposed by just fixing the values for $u(x_0)$ and $u(x_n)$, we introduce an additional grid point at some arbitrary location x_{FP}. When evaluating the matrix D^3, for example, by the algorithm in [37], this point is included just like $x_0, \ldots, x_n$. The extra BC can be expressed as a linear combination of node values:

$$\begin{bmatrix} d_{FP} & d_0 & d_1 & \ldots & d_{n-1} & d_n \end{bmatrix}_{n+2} \begin{bmatrix} u \end{bmatrix}_{n+2} = 0.$$

Since finding these weights requires the computation of only one global FD stencil (rather than a full DM), the algorithm in [12] is particularly suitable for this task. Multiples of $[d_{FP}\ d_0\ d_1\ \ldots\ d_{n-1}\ d_n]$ are next subtracted from D^3 in order to eliminate all weights at x_{FP}. The two BCs for $u(x_0)$ and $u(x_n)$ are implemented as before, i.e., we omit the first and last row and column of D^3. The key to this approach is that, apart from possible rounding errors, the resulting $\widetilde{D^3}$ matrix becomes completely independent of our choice for x_{FP}. The reason for this is that introducing this point and then eliminating all weights at it can be seen as a particularly convenient way to find the same DM as before, i.e., corresponding to collocating with polynomials of

the form

$$P(x) = (x+1)^2(x-1)p_{n-2}(x). \tag{3.1}$$

The equivalence just mentioned can be explained in more detail as follows: Data values $u_0 = 0,\ \{u_i, i = 1, 2, \ldots, n-1\},\ u_n = 0$ at node locations $x_0 = -1,\ \{x_i, i = 1, 2, \ldots, n-1\},\ x_n = 1$ determine uniquely the polynomial $P(x)$ in (3.1). We next choose an additional point x_{FP} and evaluate $u_{FP} = P(x_{FP})$. If we extend the data set above with the value u_{FP} at location x_{FP}, it will then consist of $n+2$ points, and it therefore determines a unique interpolating polynomial of degree $n+1$. This polynomial must be identical to $P(x)$. The FP approach provides the exact value for any derivative of this polynomial, at any location, i.e., its results must agree exactly with those obtained by immediate use of (3.1). The value we happened to have chosen for x_{FP} did not affect $P(x)$ (which was determined before x_{FP} was introduced), and it could therefore not in any way have influenced the outcome of the FP method.

The general idea of introducing one (or more) temporary grid points outside an interval is not new. It has long been used to implement Neumann conditions in connection with low order FD methods (e.g., [30, page 32]); other contexts include domain decomposition, block-PS methods [7], [8], and the "enslaving" idea for FD schemes [21]. In all these cases, the final computational result will depend on where the extra point(s) are located. The expression "fictitious point" was first used in [13], in a situation where the point truly vanished in the sense that not even its temporary location left any remaining trace behind. We use here the term FP in that sense. These points can just as well be placed inside a computational domain as outside it.

4. The Kuramoto–Sivashinsky equation. The KS equation can be written

$$u_t = -u\, u_x - u_{xx} - u_{xxxx}. \tag{4.1}$$

It features some similarities with Burgers' equation

$$u_t = -u\, u_x + u_{xx} \tag{4.2}$$

but is more interesting in several respects. In (4.2), the dissipative term u_{xx} counteracts the effects of steepening caused by the nonlinear term uu_x, and solutions typically evolve into pulses which travel while decaying. In (4.1), the u_{xx} term (in conjunction with the time derivative) amounts to the backwards heat equation, which is unstable for all modes. Inclusion of the u_{xxxx} term causes high modes to become dampened while low modes remain unstable. The uu_x term causes again a steepening of waves, with the effect that unstable modes of sufficient height get brought over to the stable higher-frequency regime. The KS equation models numerous phenomena in physics [34] and is a frequently used model equation for chaotic dynamics [20], [29].

A very brief Fourier-PS MATLAB code, slightly modified from [33], has been used to produce Figure 4.1. This shows how the small initial disturbance

$$u(x,0) = \begin{cases} 0.1\, \sin^2(x/2), & -12\pi < x < -10\pi, \\ 0 & \text{otherwise} \end{cases} \tag{4.3}$$

over the periodic domain $[-16\pi, 16\pi]$ causes chaotic structures to emerge.

4.1. A linear model equation. In preparation for introducing a nonperiodic PS code for the KS equation, we next consider the linear problem

$$u_t = -u_{xx} - u_{xxxx} \tag{4.4}$$

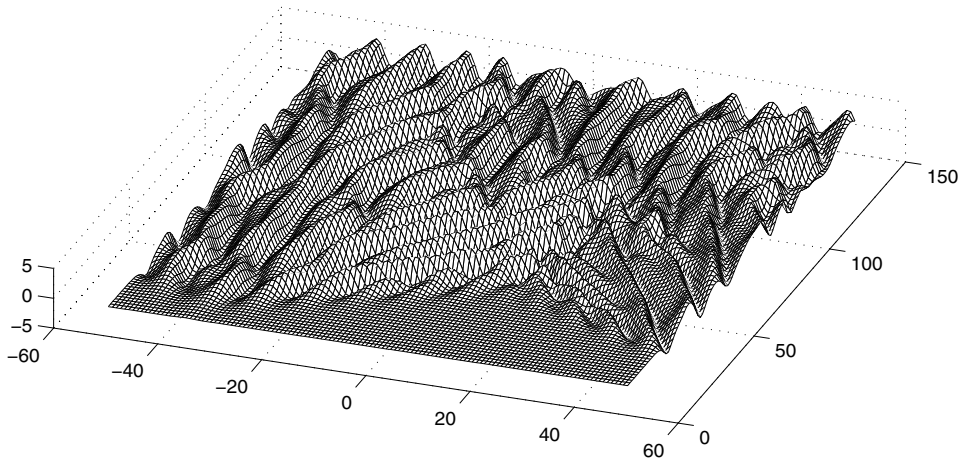


FIG. 4.1. *Periodic solution to the KS equation* (4.1) *with initial condition* (*IC*) (4.3) *computed with a periodic PS method using* 128 *points in space.*

with BCs

$$u(\pm L, t) = 0,\ u_x(\pm L, t) = 0. \tag{4.5}$$

If we at first limit ourselves still further to

$$u_t = -u_{xx} \tag{4.6}$$

with BC $u(\pm L) = 0$, the corresponding eigenvalues become

$$\lambda_k = \left[\frac{k\pi}{2L}\right]^2, \quad k = 1, 2, 3, \ldots,$$

and for

$$u_t = -u_{xxxx} \tag{4.7}$$

with BC (4.5), they become $\lambda_k = -\left(\frac{\rho_k}{2L}\right)^4$, where ρ_k is the kth positive root to $\cos(\rho) \cdot \cosh(\rho) = 1$, i.e.,

$$\lambda_k \approx -\left[\frac{(k + \frac{1}{2})\pi}{2L}\right]^4, \quad k = 1, 2, 3, \ldots .$$

The PS approach to (4.6) features all real and positive eigenvalues, as seen in Figure 4.2(a) for the case of $L = 20$ and $n = 10, 20, 30, \ldots, 80$. The lowest eigenvalues are very accurate, whereas the higher (spurious) ones become very large. Figure 4.2(b) shows similarly the FP PS method's eigenvalues for (4.7). These are all negative—again with spurious ones far away. One might have guessed that the FP-computed eigenvalues to the combined equation (4.4) would feature spurious eigenvalues both to the right and to the left (and, if so, causing maybe insurmountable time stepping difficulties). However, as Figure 4.2(c) shows, they are all far out in the left half-plane. Although the equation that gives the exact eigenvalues to (4.4) is too complicated to be practical to work with, they can nevertheless be computed accurately. The lower eigenvalues appear in close pairs, as can be seen in the tabulation along the top edge of Figure 4.3. The figure also shows graphically the errors in FP-type PS approximations based on increasingly fine Chebyshev distributions.

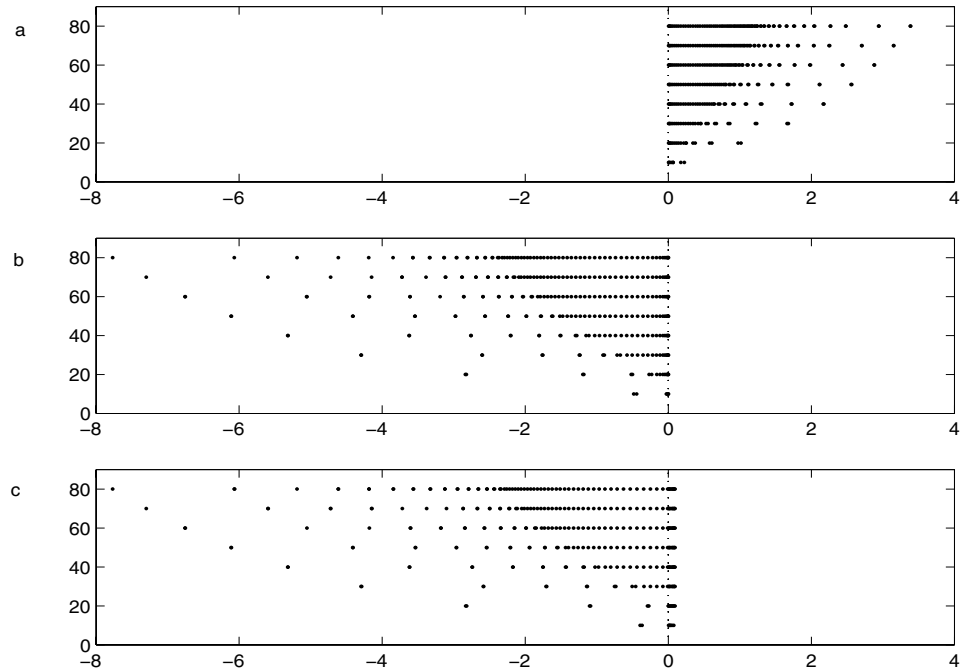


FIG. 4.2. *Eigenvalues* (*all real*) *in cases of* $N = 10, 20, 30, \ldots, 80$ *for* (a) *equation* (4.6), (b) *equation* (4.7), *and* (c) *equation* (4.4). *The scaling in the horizontal direction is the same as described in the caption for Figure* 2.1.

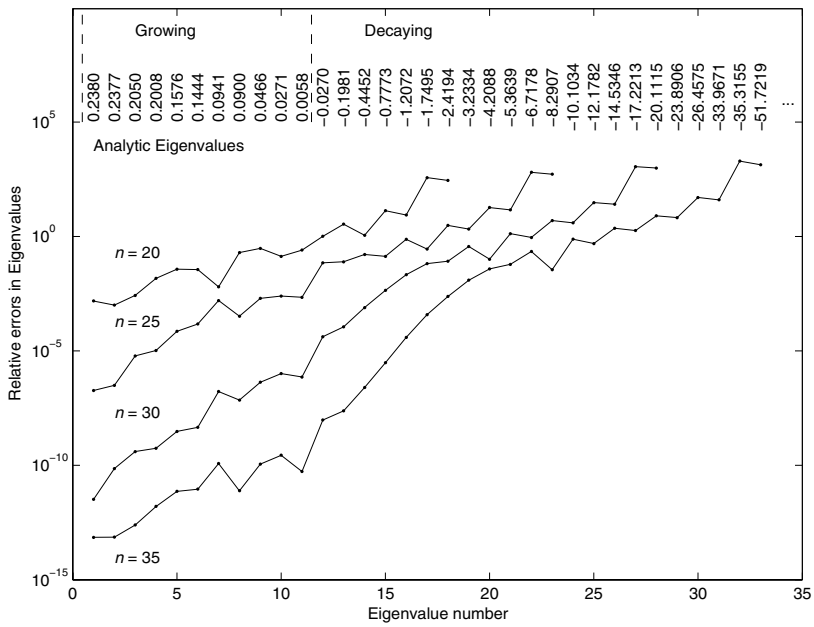


FIG. 4.3. *Relative errors in different eigenvalues for the linearized KS equation* (4.4) (*in the case of* $L = 20$).

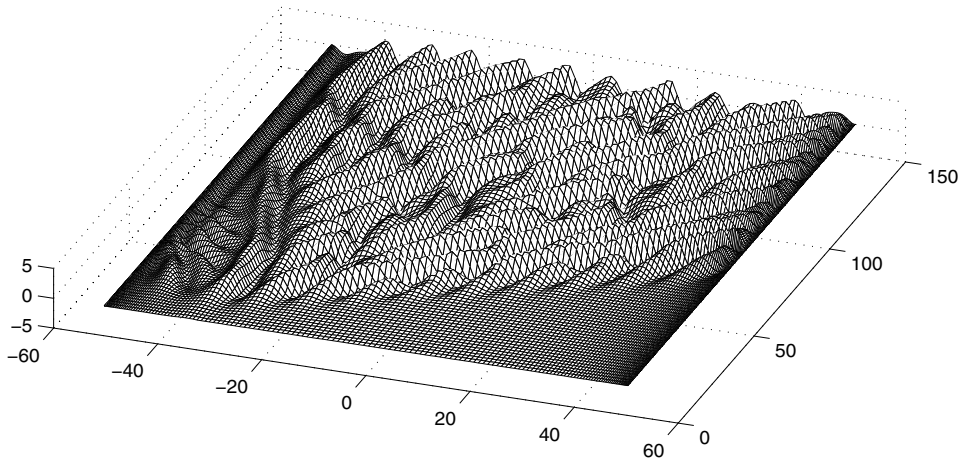


FIG. 4.4. *Nonperiodic solution to the KS equation* (4.1)*, computed by the FP PS method.*

4.2. Numerical solution of the nonperiodic KS equation. The FP PS method, as just described, has been implemented for solving the KS equation (4.1) with the nonperiodic BC $u(\pm L, t) = 0,\ u_x(\pm L, t) = 0$. With again $L = 16\pi$ and the IC (4.3), Figure 4.4 shows the resulting time evolution.

In this calculation, the time stepping (again over $0 \le t \le 150$) was performed with the 2-stage fourth order accurate Hammer–Hollingsworth implicit Runge–Kutta scheme. However, any standard stiff ODE solver should also have worked perfectly well. According to comparisons in [22], an LI (linearly implicit) scheme or an ETD (exponential time differencing) scheme would likely have been a more effective choice. The former have a long history, and were recently surveyed, for example, in [1] and [5], while Runge–Kutta-based LI schemes are considered in [28]. It should, however, be noted that Boyd [3, pages 267–269] warns against their use for certain nonlinear wave equations. ETD schemes were introduced independently several times [31], [2], [25], and [6], with a stability problem resolved in [22]. In Figure 4.4, we see again a chaotic solution that reaches approximately the same amplitude as in the periodic case. However, both quantitative and qualitative differences can be spotted compared to the periodic case that was previously displayed in Figure 4.1.

5. Time-space corner singularities. In applications leading to IBV problems, ICs and BCs typically come from different physical considerations. In order for the solutions to the PDE not to feature singularities in the time-space corners where ICs and BCs meet, an infinite number of compatibility conditions need to hold, as explored by Flyer and collaborators [4], [9], [10], [11]. For (1.1), these were analyzed in detail in [10]. Figure 5.1 shows for $k = 1$ and $k = 2$ the structure of the solutions

(5.1)
$$u_k(x,t) = t^k \left\{ {}_1F_2\left(-k, \left\{\frac{1}{3}, \frac{2}{3}\right\}, -\frac{x^3}{27t}\right) - \frac{x^2}{2t^{2/3}} \frac{\Gamma(k+1)}{\Gamma(k+\frac{1}{3})}\ {}_1F_2\left(\frac{2}{3} - k, \left\{\frac{4}{3}, \frac{5}{3}\right\}, -\frac{x^3}{27t}\right)\right\}$$

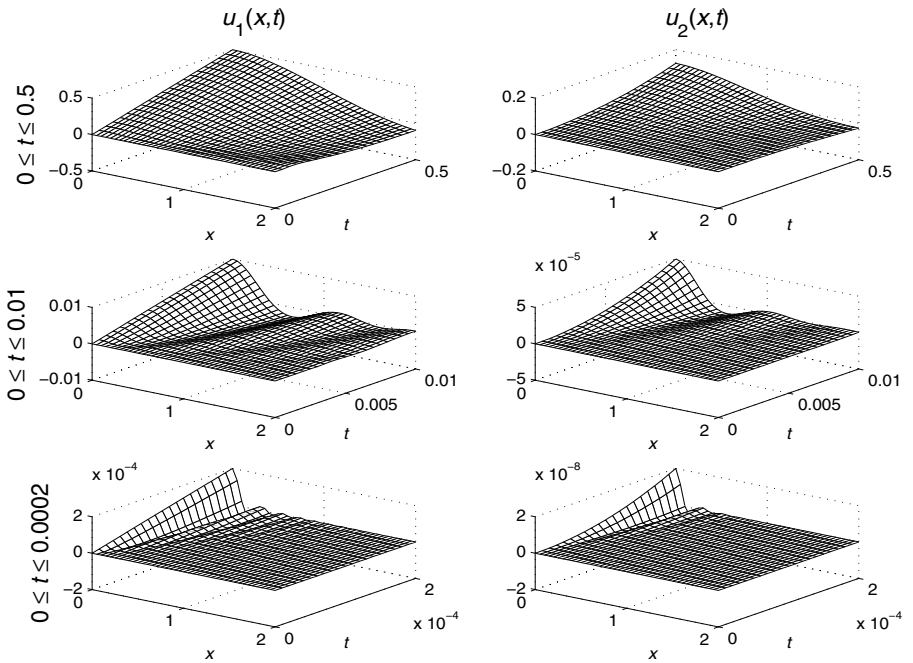


FIG. 5.1. *The corner functions $u_1(x,t)$ and $u_2(x,t)$ for* (5.2)*, displayed over progressively shorter initial time intervals.*

near the origin for the quarter-plane problems

$$
\text{(5.2)} \qquad \begin{cases} \text{PDE} & u_t = u_{xxx}, \\ \text{IC} & u(x,0) = 0, \\ \text{BC} & \begin{cases} u(t,0) &= t^k, \\ u_x(t,0) &= 0, \end{cases} \quad k = 0,1,2,\ldots; \;\; u(t,\infty) = 0, \end{cases}
$$

$x \geq 0$, $t \geq 0$. For the BCs $u(t,0) = 0, \;\; u_x(t,0) = t^k$, we denote the corresponding quarter-plane solutions by $v_k(x,t)$:

$$
v_k(x,t) = t^{k+\frac{1}{3}} \left\{ {}_1F_2\left(-k, \left\{\frac{2}{3}, \frac{4}{3}\right\}, -\frac{x^3}{27t}\right) - \frac{x^2}{2t^{2/3}} \frac{\Gamma(k+1)}{\Gamma(k+\frac{2}{3})} \, {}_1F_2\left(\frac{1}{3} - k, \left\{\frac{4}{3}, \frac{5}{3}\right\}, -\frac{x^3}{27t}\right) \right\}.
$$

The top row of subplots in Figure 5.1 shows seemingly very smooth analytic solutions, for which one would not expect any numerical difficulties. No amount of "zooming in" (uniformly in x and t) near the origin in the (x,t)-plane will visually reveal any irregularities. However, if we increase the graphical resolution in time only, the lower subplots reveal the presence of transient fine structures which accurate numerics would also need to resolve. A typical numerical approach, as explained in some detail for convective-diffusive equations in [9], would be to subtract an appropriate combination of different [illegible] before an FP PS numerical method is applied to the remaining problem.

The issue of corner singularities in the time-space domain was little noted before the advent of spectral methods, since the truncation errors of finite difference and finite element methods tended to dominate over them. However, when using PS methods,

corner-induced errors can easily dominate over other errors with many orders of magnitude. In the present test case for the KS equation (4.1), all space derivatives of the IC (4.3) and time derivatives of the BC (4.5) are zero in the corners, and this issue did not arise. However, it needs to be noted that it is one that has to be considered in most spectral IBV calculations. The precise form of the corner singularity functions for most higher order PDEs (such as (4.4) and (4.1)) remains to be investigated. For cases when they cannot be obtained in convenient closed form, it was noted in [9] that a simple change of variables makes their numerical computation straightforward.

6. Inhomogeneous boundary conditions. The description of the FP approach in section 3 applies entirely unchanged also to the case of inhomogeneous BCs. As an example, we consider the case illustrated in the center left subplot of Figure 5.1, i.e., we use the FP method to numerically solve

$$\text{(6.1)} \qquad \begin{cases} \text{PDE} & u_t = u_{xxx}, \\ \text{IC} & u(x,0) = 0, \\ \text{BC} & u(0,t) = t,\ u_x(0,t) = 0,\ u(2,t) = \left\{ \begin{array}{c} \text{values from the ana-} \\ \text{lytic solution (5.1)} \end{array} \right\} \end{cases}$$

for $0 \le t \le 0.01$. The resulting solution should agree exactly with the $k = 1$ case of (5.1). Figure 6.1(a) shows the result on a 20-node Chebyshev grid with the time stepping, this time carried out with one of MATLAB's standard stiff ODE solvers (ode15s). Part (b) of the figure shows the error compared to the analytic solution. At the very first moments, extremely high frequency transients emerge from the origin (where the ICs and BCs meet) and then race to the right across the spatial domain. After they have done so, and have "died down" at the right boundary, the numerical solution features a very high accuracy. As we noted above, these initial transients at first contain frequencies that are too high for any computational grid, and the error they cause

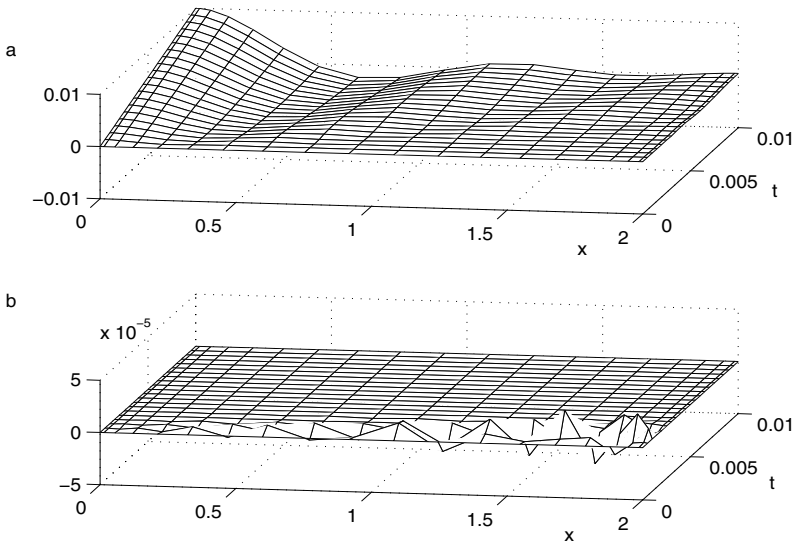


FIG. 6.1. *The $u_1(x,t)$ corner function for $u_t = u_{xxx}$.* (a) *Numerical FP solution on a* 20*-node Chebyshev grid.* (b) *The error compared to the analytic solution.*

can easily dominate all other errors by several orders of magnitude. It depends on the application whether or not accurate resolution of initial transients is important.

7. Conclusions. High order IBV problems can require several conditions at each boundary. When using Chebyshev-type PS approximations in space, it has in the past not been clear how best to implement these conditions in order to obtain good accuracy, stability, flexibility, and simplicity. The present FP approach has been shown to meet such requirements. Demonstration cases have included the KS equation as well as the dispersive wave equation $u_t = u_{xxx}$ with inhomogeneous boundary conditions. As with other Chebyshev-type PS methods, the DMs in the FP approach are not particularly close to being normal matrices. However, they are close enough that eigenvalue analysis accurately describes actual stability under time stepping. With the very high accuracy that is now easily reachable also for high order IBV problems, the singularities that such equations generally have at time-space corners (where ICs and BCs meet) will need to be taken into account, especially in computations over short times.

REFERENCES

[1] U. M. Ascher, S. J. Ruuth, and B. T. R. Wetton, *Implicit-explicit methods for time-dependent partial differential equations*, SIAM J. Numer. Anal., 32 (1995), pp. 797–823.

[2] G. Beylkin, J. M. Keiser, and L. Vozovoi, *A new class of time discretization schemes for the solution of nonlinear PDEs*, J. Comput. Phys., 147 (1998), pp. 362–387.

[3] J. P. Boyd, *Chebyshev and Fourier Spectral Methods*, 2nd ed., Dover, New York, 2000.

[4] J. P. Boyd and N. Flyer, *Compatibility conditions for time-dependent partial differential equations and the rate of convergence of Chebyshev and Fourier spectral methods*, Comput. Methods Appl. Mech. Engrg., 175 (1999), pp. 281–309.

[5] C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang, *Spectral Methods in Fluid Dynamics*, Springer-Verlag, Berlin, 1988.

[6] S. M. Cox and P. C. Matthews, *Exponential time differencing for stiff systems*, J. Comput. Phys., 176 (2002), pp. 430–455.

[7] T. A. Driscoll and B. Fornberg, *A block pseudospectral method for Maxwell's equations:* I. *One-dimensional case*, J. Comput. Phys., 140 (1998), pp. 47–65.

[8] T. A. Driscoll and B. Fornberg, *Block pseudospectral methods for Maxwell's equations* II: *Two-dimensional, discontinuous-coefficient case*, SIAM J. Sci. Comput., 21 (1999), pp. 1146–1167.

[9] N. Flyer and B. Fornberg, *Accurate numerical resolution of transients in initial-boundary value problems for the heat equation*, J. Comput. Phys., 184 (2003), pp. 526–539.

[10] N. Flyer and B. Fornberg, *On the nature of initial-boundary value solutions for dispersive equations*, SIAM J. Appl. Math., 64 (2003), pp. 546–564.

[11] N. Flyer and P. N. Swarztrauber, *The convergence of spectral and finite difference methods for initial-boundary value problems*, SIAM J. Sci. Comput., 23 (2002), pp. 1731–1751.

[12] B. Fornberg, *Generation of finite difference formulas on arbitrarily spaced grids*, Math. Comp., 51 (1988), pp. 699–706.

[13] B. Fornberg, *A Practical Guide to Pseudospectral Methods*, Cambridge University Press, Cambridge, UK, 1996.

[14] B. Fornberg, *Calculation of weights in finite difference formulas*, SIAM Rev., 40 (1998), pp. 685–691.

[15] D. R. Gardner, S. A. Trogdon, and R. W. Douglas, *A modified tau spectral method that eliminates spurious eigenvalues*, J. Comput. Phys., 80 (1989), pp. 137–167.

[16] D. Funaro and W. Heinrichs, *Some results about the pseudospectral approximation of one-dimensional fourth-order problems*, Numer. Math., 58 (1990), pp. 399–418.

[17] W. Heinrichs, *Spectral approximation of third-order problems*, J. Sci. Comput., 14 (1999), pp. 275–289.

[18] W. Huang and D. M. Sloan, *The pseudospectral method for third-order differential equations*, SIAM J. Numer. Anal., 29 (1992), pp. 1626–1647.

[19] W. Huang and D. M. Sloan, *The pseudospectral method for solving differential eigenvalue problems*, J. Comput. Phys., 111 (1994), pp. 399–409.

[20] J. M. Hyman and B. Nicolaenko, *The Kuramoto–Sivashinsky equation: A bridge between partial differential equations and dynamical systems*, Phys. D, 18 (1986), pp. 113–126.
[21] D. A. Jones, L. G. Margolin, and A. C. Poje, *Enslaved finite difference schemes for nonlinear dissipative PDEs*, Numer. Methods Partial Differential Equations, 12 (1996), pp. 13–40.
[22] A.-K. Kassam and L. N. Trefethen, *Fourth-order time-stepping for stiff PDEs*, SIAM J. Sci. Comput., 26 (2005), pp. 1214–1233.
[23] G. B. McFadden, B. T. Murray, and R. F. Boisvert, *Elimination of spurious eigenvalues in the Chebyshev tau spectral method*, J. Comput. Phys., 91 (1990), pp. 228–239.
[24] W. J. Merryfield and B. Shizgal, *Note: Properties of collocation third-derivative operators*, J. Comput. Phys., 105 (1993), pp. 182–185.
[25] D. R. Mott, E. S. Oran, and B. van Leer, *A quasi-steady state solver for the stiff ordinary differential equations of reaction kinetics*, J. Comput. Phys., 164 (2000), pp. 407–428.
[26] L. S. Mulholland, *The Eigenvalues of Third-Order Chebyshev Pseudospectral Differentiation Matrices*, Mathematics Research Report, 6, University of Strathclyde, Glasgow, UK, 1995.
[27] K. L. Nielsen, *Methods in Numerical Analysis*, MacMillan, New York, 1956.
[28] S. J. Ruuth, *Implicit-explicit Runge–Kutta methods for time dependent partial differential equations*, Appl. Numer. Math., 25 (1997), pp. 151–167.
[29] B. I. Shraiman, *Order, disorder, and phase turbulence*, Phys. Rev. Lett., 57 (1986), pp. 325–328.
[30] G. D. Smith, *Numerical Solution of Partial Differential Equations*, Oxford University Press, New York, 1965.
[31] A. Taflove, *Computational Electrodynamics: The Finite-Difference Time-Domain Method*, Artech House, Boston, 1995.
[32] L. N. Trefethen, *Spectral Methods in MATLAB*, SIAM, Philadelphia, 2000.
[33] L. N. Trefethen, 1-*page,* 1-*second Matlab ETDRK*4 *code for Kuramoto–Sivashinsky equation*, available online from http://web.comlab.ox.ac.uk/oucl/work/nick.trefethen/.
[34] L. N. Trefethen, *The PDE Coffee Table Book*, in preparation.
[35] J. R. Quan and C. T. Chang, *New insights in solving distributed system equations by the quadrature methods*—I., Comput. Chem. Engrg., 13 (1989), pp. 779–788.
[36] J. R. Quan and C. T. Chang, *New insights in solving distributed system equations by the quadrature methods*—II., Comput. Chem. Engrg., 13 (1989), pp. 1017–1024.
[37] J. A. C. Weideman and S. C. Reddy, *A MATLAB differentiation matrix suite*, ACM Trans. Math. Software (TOMS), 26 (2000), pp. 465–519. (See also http:/dip.sun.ac.za/˜weideman/research/differ.html.)
[38] B. D. Welfert, *Generation of pseudospectral differentiation matrices* I, SIAM J. Numer. Anal., 34 (1997), pp. 1640–1657.

SIAM J. SCI. COMPUT.
Vol. 28, No. 5, pp. 1730–1765
© 2006 Society for Industrial and Applied Mathematics

A HIGH-ORDER ACCURATE PARALLEL SOLVER FOR MAXWELL'S EQUATIONS ON OVERLAPPING GRIDS*

WILLIAM D. HENSHAW†

Abstract. A scheme for the solution of the time-dependent Maxwell's equations on composite overlapping grids is described. The method uses high-order accurate approximations in space and time for Maxwell's equations written as a second-order vector wave equation. High-order accurate symmetric difference approximations to the generalized Laplace operator are constructed for curvilinear component grids. The modified equation approach is used to develop high-order accurate approximations that use only three time levels and have the same time-stepping restriction as the second-order scheme. Discrete boundary conditions for perfect electrical conductors and for material interfaces are developed and analyzed. The implementation is optimized for component grids that are Cartesian, resulting in a fast and efficient method. The solver runs on parallel machines with each component grid distributed across one or more processors. Numerical results in two and three dimensions are presented for the fourth-order accurate version of the method. These results demonstrate the accuracy and efficiency of the approach.

Key words. Maxwell's equations, overlapping grids, high-order accurate, symmetric finite difference

AMS subject classifications. 65M06, 65M12, 78M20

DOI. 10.1137/050644379

1. Introduction.

A numerical scheme is described for the fast and accurate solution of the time-dependent Maxwell's equations in complex geometry. Maxwell's equations are solved as a second-order vector wave equation rather than the more common approach of treating the equations as a first-order system. The governing equations are discretized to high-order accuracy in space and time on domains that are represented with composite overlapping grids. High-order accurate symmetric finite difference discretizations of the generalized Laplace operator are devised for use on curvilinear grids. High-order accurate centered approximations to boundary conditions are developed and analyzed. The three-level time-stepping scheme is based on the modified equation approach and achieves high-order accuracy through a Taylor series in time that uses the differential equations to replace time derivatives by space derivatives. The time-step restriction, as dictated by stability, is the same as that used with the second-order version of the method. The interface between materials with different electric and magnetic properties is treated using component grids that align with the interface. High-order accurate centered approximations to the interface jump conditions are developed that maintain the full accuracy of the scheme at the interface. The numerical approach presented here is particularly fast and efficient on Cartesian component grids. A typical overlapping grid for a region Ω will consist of body fitted curvilinear grids near the boundaries together with one or more background Cartesian grids covering most of the domain. For sufficiently fine grids, the majority of the grid

*Received by the editors November, 5, 2005; accepted for publication (in revised form) April 26, 2006; published electronically October 20, 2006. This research was supported under the auspices of the U.S. Department of Energy by the University of California, Lawrence Livermore National Laboratory, under contract W-7405-Eng-48.

http://www.siam.org/journals/sisc/28-5/64437.html

†Center for Applied Scientific Computing, Lawrence Livermore National Laboratory, Livermore, CA 94551 (henshaw1@llnl.gov).

points will reside on the Cartesian grids. In this case, the efficiency of the scheme approaches that of a single Cartesian grid.

The time-dependent Maxwell's equations for linear, isotropic, and nondispersive materials are

$$\partial_t \mathbf{E} = \frac{1}{\epsilon}\nabla \times \mathbf{H} - \frac{1}{\epsilon}\mathbf{J}, \tag{1.1}$$

$$\partial_t \mathbf{H} = -\frac{1}{\mu}\nabla \times \mathbf{E}, \tag{1.2}$$

$$\nabla \cdot (\epsilon \mathbf{E}) = \rho, \quad \nabla \cdot (\mu \mathbf{H}) = 0. \tag{1.3}$$

Here $\mathbf{E} = \mathbf{E}(\mathbf{x}, t)$ is the electric field, $\mathbf{H} = \mathbf{H}(\mathbf{x}, t)$ is the magnetic field, $\rho = \rho(\mathbf{x}, t)$ is the electric charge density, $\mathbf{J} = \mathbf{J}(\mathbf{x}, t)$ is the electric current density, $\epsilon = \epsilon(\mathbf{x})$ is the electric permittivity, and $\mu = \mu(\mathbf{x})$ is the magnetic permeability. This first-order system for Maxwell's equations can also be written in a second-order form. By taking the time derivatives of (1.2) and (1.1) and using (1.3) it follows that

$$\epsilon\mu\ \partial_t^2 \mathbf{E} = \Delta \mathbf{E} + \nabla\Big(\nabla \ln \epsilon\ \cdot \mathbf{E}\Big) + \nabla \ln \mu \times \Big(\nabla \times \mathbf{E}\Big) - \nabla\left(\frac{1}{\epsilon}\rho\right) - \mu \partial_t \mathbf{J}, \tag{1.4}$$

$$\epsilon\mu\ \partial_t^2 \mathbf{H} = \Delta \mathbf{H} + \nabla\Big(\nabla \ln \mu\ \cdot \mathbf{H}\Big) + \nabla \ln \epsilon \times \Big(\nabla \times \mathbf{H}\Big) + \epsilon \nabla \times \left(\frac{1}{\epsilon}\mathbf{J}\right). \tag{1.5}$$

It is evident that the equations for the electric and magnetic field are decoupled with each satisfying a vector wave equation with lower-order terms. In the case of constant μ and ϵ and no charges, $\rho = \mathbf{J} = 0$, the equations simplify to the classical second-order wave equations,

$$\partial_t^2 \mathbf{E} = c^2\ \Delta \mathbf{E}, \qquad \partial_t^2 \mathbf{H} = c^2\ \Delta \mathbf{H}, \tag{1.6}$$

where $c^2 = 1/(\epsilon\mu)$. There are some advantages to solving the second-order form of the equations rather than the first-order system. One advantage is that in some cases it is necessary only to solve for one of the variables, say $\mathbf{E}$. If the other variable, $\mathbf{H}$, is required, it can be determined by integrating (1.2) as an ordinary differential equation with known $\mathbf{E}$. Alternatively, as a postprocessing step $\mathbf{H}$ can be computed from an elliptic boundary value problem formed by taking the curl of (1.1). Another advantage of the second-order form, which simplifies the implementation on an overlapping grid, is that there is no need to use a staggered grid formulation. Many schemes approximating the first-order system (1.2)–(1.3) rely on a staggered arrangement of the components of $\mathbf{E}$ and $\mathbf{H}$ such as the popular Yee scheme [44] for Cartesian grids. From a numerical point of view, there is an advantage in that the principal part of the spatial operator of the second-order system (1.4) is the Laplace operator Δ, which is a strongly elliptic operator and thus amenable to the discretization through compact difference approximations. The corresponding operator $-\nabla \times \nabla$ for the first-order system (1.1)–(1.2), on the other hand, has a large null space consisting of all functions that are a gradient of a scalar field, and requires more care in discretizing. The development of appropriate boundary conditions is sometimes cited as a disadvantage of solving the second-order form of the equations. The second-order continuous formulation requires additional boundary conditions as discussed in section 2, as well as additional numerical boundary conditions for higher-order accurate discretizations. A primary purpose of this work is to show how to define accurate centered approximations to the boundary conditions that are based upon compatibility conditions consistent with the governing equations.

There have been a variety of numerical approaches developed for solving the Maxwell's equations in the time domain, and there is extensive literature on the subject including the books by Cohen [4] and Taflove and Hagness [41]. Some references to the different approaches will now be given. One of the most popular approaches is the staggered grid scheme of Yee [44], commonly known as the finite-difference time-domain (FDTD) method. For complex geometry, a [illegible] approximation to the boundary is often used with the scheme. Improvements to the accuracy of the basic stair-step technique have been considered by, for example, Dey and Mittra [7], Ditkowski, Dridi, and Hesthaven [9], and Kreiss and Petersson [27]. The Yee scheme has been extended to curvilinear grids (see, for example, Lee, Palendech, and Mittra [30]) and to unstructured grids such as in the discrete surface-integral (DSI) scheme originally developed by Madsen [32] and extended by Gedney, Lansing, and Rascoe [13]. Finite volume methods have been applied to the solution of Maxwell's equations as discussed, for example, in Holland [24]. Jurgens and Zingg [26] used high-order finite-difference approximations on block-structured grids for the equations written as a first-order system. Schemes based on the finite element method (FEM) have been developed to solve Maxwell's equations; see, for example, Nédélec [33], Jin [25], and Rodrigue and White [37]. The spectral element method has been used by a variety of authors including Yang, Gottlieb, and Hesthaven [43]. High-order spectral penalty methods and discontinuous Galerkin-type approaches have also been successful; see, for example, Hesthaven and Warburton [23]. Use of hybrid grids, part structured and part unstructured, has become increasingly popular. Rylander and Bondeson [38], for example, presented a hybrid FEM-FDTD method that retains the symmetry of the discrete operators, even at the interface between the FEM and FDTD methods. Yee, Chen, and Chang [45] considered the solution of Maxwell equations on overlapping grids using a second-order accurate method. Driscoll and Fornberg [10, 11] developed a block pseudospectral method for Maxwell's equations on two-dimensional overlapping grids. They solved the equations as a first-order system using a high-order difference method on the Cartesian background grid and a pseudospectral method on the boundary fitted curvilinear grids.

2. Governing equations and boundary conditions. The numerical scheme is based upon the solution of the second-order form of Maxwell's equations. To begin with, the material properties $\epsilon(\mathbf{x})$ and $\mu(\mathbf{x})$ are assumed to be constant. The case of multiple materials will be treated in section 7. In three space dimensions, the initial boundary value problem for Maxwell's equations on a domain Ω is given by

$$\partial_t^2 \mathbf{E} = c^2 \, \Delta \mathbf{E}, \qquad \mathbf{x} \in \Omega, \tag{2.1}$$

$$\mathbf{n} \times \mathbf{E} = 0, \;\; \nabla \cdot \mathbf{E} = 0, \qquad \mathbf{x} \in \partial\Omega_E \quad \text{(PEC BCs)}, \tag{2.2}$$

$$\mathcal{B}_F(\mathbf{E}) = 0, \;\; \nabla \cdot \mathbf{E} = 0, \qquad \mathbf{x} \in \partial\Omega_F \quad \text{(far field BCs)}, \tag{2.3}$$

$$\mathbf{E}(\mathbf{x},0) = \mathbf{E}^{(0)}(\mathbf{x}), \quad \mathbf{E}_t(\mathbf{x},0) = \mathbf{E}_t^{(0)}(\mathbf{x}) \qquad \text{(initial conditions)}, \tag{2.4}$$

where $\mathbf{E} = (E_x, E_y, E_z)$ and $c = 1/\sqrt{\epsilon\mu}$. The outward normal to the boundary $\partial\Omega$ is $\mathbf{n}$. At a boundary, $\partial\Omega_E$, next to a perfect electrical conductor (PEC), the boundary condition (BC) is that the tangential components of the field are zero; i.e., $\mathbf{n} \times \mathbf{E} = 0$. For boundaries where the computational domain is truncated, i.e., $\partial\Omega_F$, appropriate far-field boundary conditions are applied. These are denoted by $\mathcal{B}_F(\mathbf{E}) = 0$. The initial conditions should satisfy the constraints $\nabla \cdot \mathbf{E}^{(0)} = 0$ and $\nabla \cdot \mathbf{E}_t^{(0)} = 0$. Note that the second-order formulation requires the additional boundary condition $\nabla \cdot \mathbf{E} = 0$. The fact that the Gauss law, $\nabla \cdot \mathbf{E} = 0$, holds everywhere can be seen by noting that

the divergence of the electric field, $\delta = \nabla \cdot \mathbf{E}$, satisfies a second-order wave equation, $\delta_{tt} = c^2\Delta\delta$, with zero initial conditions and zero boundary conditions, and thus δ will remain identically zero.

In two space dimensions the so-called $\mathrm{TE_Z}$ mode (transverse electric mode with respect to z) is considered. Let $\mathbf{w} = (u, v, w) = (E_x, E_y, H_z)$ denote the vector holding the x- and y-components of the electric field and the z-component of the magnetic field. In addition, let $\mathbf{u} = (u, v) = (E_x, E_y)$. The initial boundary value problem for Maxwell's equations in this case is

$$\partial_t^2 \mathbf{w} = c^2 \Delta \mathbf{w}, \qquad \mathbf{x} \in \Omega, \tag{2.5}$$

$$\mathbf{u}(\mathbf{x},0) = \mathbf{u}^{(0)}(\mathbf{x}), \quad \mathbf{u}_t(\mathbf{x},0) = \mathbf{u}_t^{(0)}(\mathbf{x}), \quad t = 0 \qquad \text{(initial conditions)}, \tag{2.6}$$

$$w(\mathbf{x},0) = w^{(0)}(\mathbf{x}), \quad w_t(\mathbf{x},0) = w_t^{(0)}(\mathbf{x}), \quad t = 0 \qquad \text{(initial conditions)}, \tag{2.7}$$

$$\mathbf{n} \times \mathbf{u} = 0, \quad \nabla \cdot \mathbf{u} = 0, \quad \partial_n w = 0, \qquad \mathbf{x} \in \partial\Omega_{\mathrm{E}} \quad \text{(PEC BCs)}, \tag{2.8}$$

$$\mathcal{B}_F(\mathbf{w}) = 0, \quad \nabla \cdot \mathbf{u} = 0, \qquad \mathbf{x} \in \partial\Omega_F \quad \text{(far field BCs)}. \tag{2.9}$$

As in three dimensions, the initial conditions should satisfy the Gauss law, $\nabla\cdot\mathbf{u}^{(0)} = 0$, $\nabla \cdot \mathbf{u}_t^{(0)} = 0$. In two dimensions it may be possible to solve for just one component, $w = H_z$, or the two components, $\mathbf{u} = (E_x, E_y)$. However, all three components will be considered here in order to develop the numerical scheme for either case.

3. Summary of the numerical scheme. Consider the second-order wave equation with a forcing function, $u_{tt} = c^2\Delta u + f$. An attractive time-stepping method to solve this second-order wave equation is the [illegible] approach based on the Taylor series

$$u(\mathbf{x}, t+\Delta t) - 2u + u(\mathbf{x}, t-\Delta t) = 2\tfrac{\Delta t^2}{2!}\partial_t^2 u + 2\tfrac{\Delta t^4}{4!}\partial_t^4 u + 2\tfrac{\Delta t^6}{6!}\partial_t^6 u + \cdots .$$

Using the wave equation to eliminate even numbers of time derivatives in terms of space derivatives gives

$$\begin{aligned} u(\mathbf{x}, t+\Delta t) - 2u + u(\mathbf{x}, t-\Delta t) = {} & 2\tfrac{\Delta t^2}{2!}\big(c^2\Delta u + f\big) \\ & + 2\tfrac{\Delta t^4}{4!}\Big\{(c^2\Delta)^2 u + c^2\Delta f + f_{tt}\Big\} + \cdots . \end{aligned} \tag{3.1}$$

This last expansion can be used to derive an approximation of any order that uses only three time levels. Letting $U_{\mathbf{i}}^n \approx u(\mathbf{x}_{\mathbf{i}}, t^n)$ denote a discrete approximation to u at the grid point $\mathbf{x}_{\mathbf{i}}$ and time $t^n = n\Delta t$, a fourth-order accurate approximation is

$$U_{\mathbf{i}}^{n+1} - 2U_{\mathbf{i}}^n + U_{\mathbf{i}}^{n-1} = \Delta t^2\big(c^2\Delta_{4h}U_{\mathbf{i}}^n + f\big) + \tfrac{\Delta t^4}{12}\big(c^4(\Delta^2)_{2h}U_{\mathbf{i}}^n + c^2\Delta_{2h}f + f_{tt}\big), \tag{3.2}$$

where Δ_{mh} denotes some mth-order approximation to the Laplace operator Δ and $(\Delta^2)_{mh}$ denotes an mth-order approximation to the square of the Laplace operator. The precise form of these approximations will be given in section 5. Using a second-order approximation for Δ^2 prevents the discrete stencil from becoming wider than that of the fourth-order approximation for Δ, while retaining the order of accuracy. Higher-order approximations follow easily. The modified equation time-stepping approach is related to the Numerov and Stoermer methods for ODEs [29], and for the second-order wave equation it has been considered by numerous authors, including Dablain [6] and Shubin and Bell [39]. Anné, Joly, and Tran [1] showed that in one

space dimension the time-step restriction required for stability is independent of the order of accuracy. Gustafsson and Mossberg [15] used a similar approach for treating the equations written as a first-order system.

The time-stepping scheme is not restricted to the modified equation approach. The spatial discretization and boundary conditions can be employed in a method-of-lines approach and used with any number of time-stepping schemes for the second-order ODE $u_{tt} = F(u,t)$. High-order Stoermer approximations [29] have been implemented and used with success. Although these schemes require a smaller time step as the order of accuracy increases, the approach may be attractive to simplify the implementation details for the case of when $\epsilon(\mathbf{x})$ and $\mu(\mathbf{x})$ vary in space.

High-order accurate finite difference approximations that use wide stencils require special treatment at boundaries. One common approach for treating approximations near boundaries is to use one-sided difference approximations. Another approach, the one primarily taken here, is to use high-order accurate centered approximations to boundary conditions that use compatibility conditions derived from the continuous formulation [20, 14]. In particular, appropriate compatibility conditions for a PEC boundary are now given. The essential boundary conditions for a PEC are given by (2.2). By taking $2m$-time derivatives of these boundary conditions and using (2.1) it follows that

$$\mathbf{n} \times (\Delta^m \mathbf{E})(\mathbf{x},t) = 0, \qquad \nabla \cdot (\Delta^m \mathbf{E})(\mathbf{x},t) = 0, \qquad \mathbf{x} \in \partial\Omega_E, \tag{3.3}$$

for $m = 0, 1, 2, \ldots$. In the two-dimensional case, taking $2m$-time derivatives of the Neumann boundary condition for H_z leads to the compatibility conditions

$$\partial_n \Delta^m H_z(\mathbf{x},t) = 0, \qquad \mathbf{x} \in \partial\Omega_E, \tag{3.4}$$

for $m = 0, 1, 2, \ldots$. Discrete approximations to the compatibility conditions (3.3)–(3.4) can be used as numerical boundary conditions. In general, the use of centered boundary conditions such as those based on these compatibility conditions will be more stable and more accurate than boundary conditions based on one-sided differences.

In summary, the proposed scheme for Maxwell's equations uses the modified equation time-stepping approximation based on (3.1), such as (3.2) for a fourth-order scheme, together with discrete approximations to the boundary conditions (2.2) and compatibility conditions (3.3) or (3.4). These equations are discretized on an overlapping grid. Symmetric difference approximations to the equations are developed in section 5. The solution values at the overlapping grid interpolation points are obtained using high-order accurate interpolation as described in section 4. On a Cartesian grid the basic scheme is neutrally stable and nondissipative. From an analysis of similar problems [35], it can be expected that such a scheme may develop some weak instabilities near boundaries of curvilinear grids and near interpolation points of the overlapping grid. These instabilities are easily dealt with, however, by the addition of a high-order spatial dissipation. This dissipation does not change the accuracy of the scheme and can be added locally in regions near boundaries and interpolation points. Computational results show that this dissipation generally has a negligible effect on the overall accuracy of the solution, even for long times.

4. Overlapping grids. The governing equations for Maxwell's equations are discretized on an overlapping grid $\mathcal{G}$ (see Figures 7 and 10). The overlapping grid consists of a set of component grids, $\mathcal{G} = \{G_g\}$. The component grids overlap and

cover the domain Ω. Typically, body fitted curvilinear grids are used near the boundaries, while one or more background Cartesian grids are used in the remainder of the domain. Each component grid G_g is a logically rectangular, curvilinear grid in d space dimensions defined by a smooth mapping $\mathbf{x} = \mathbf{G}_g(\mathbf{r})$, from parameter space $\mathbf{r} \in [0,1]^d$ (the unit-square or unit-cube) to physical space $\mathbf{x} \in \mathbb{R}^d$. This mapping is used to define the location of grid points at any desired resolution. Derivatives of the mapping, $\partial\mathbf{x}/\partial\mathbf{r}$, will appear as coefficients in the discrete approximations, as shown in section 5. Thus, for high-order approximations it is important that the mapping $\mathbf{G}_g(\mathbf{r})$ be sufficiently smooth to avoid numerical artifacts such as spurious reflections. Points on a grid are classified as discretization points (where the PDE or boundary conditions are discretized), unused points, and interpolation points (where the solution is interpolated from a different grid). The values of the solution at interpolation points are determined by a tensor-product Lagrange interpolant, using w points in each direction. For a pth-order approximation to the second-order wave equation, $p = 2, 4, \ldots$, the width of the interpolation stencil is taken as $\mathrm{w} = p + 1$, following the analysis in [3]. Let $N_m^{(g)}$ grid denote the number of grid cells on grid G_g in each coordinate direction, $m = 1, 2, \ldots, d$. Let $\mathbf{i} = (i_1, i_2, i_3)$ denote a multi-index, and let $\mathbf{x}_\mathbf{i} = \mathbf{G}(\mathbf{r}_\mathbf{i})$ denote the grid point corresponding to the unit-square coordinate $\mathbf{r}_\mathbf{i} = (i_1\Delta r_1, i_2\Delta r_2, i_3\Delta r_3)$, where $i_m = 0, 1, \ldots, N_m$ and $\Delta r_m = 1/N_m$. Ghost points will correspond to $i_m = -1, -2, \ldots$ and $i_m = N_m + 1, N_m + 2, \ldots$. For more details on the discretization of problems on overlapping grids, see [3, 21, 22].

Introduce a time step Δt, and let $U_\mathbf{i}^n$ denote a grid function at time $t^n = n\Delta t$, representing an approximation to $E_x(\mathbf{x}_\mathbf{i}, t^n)$, for example. Introduce the standard forward, backward, and centered divided difference operators in the r_1 direction by

$$D_{+r_1}U_\mathbf{i} = (U_{i_1+1,i_2,i_3} - U_\mathbf{i})/(\Delta r_1), \quad D_{-r_1}U_\mathbf{i} = (U_\mathbf{i} - U_{i_1-1,i_2,i_3})/(\Delta r_1),$$
$$D_{0r_1}U_\mathbf{i} = (U_{i_1+1,i_2,i_3} - U_{i_1-1,i_2,i_3})/(2\Delta r_1).$$

Similar definitions hold for difference operators in the other coordinate directions, D_{+r_m}, D_{-r_m}, and D_{0r_m} for $m = 2, 3$. On a Cartesian grid, the grid spacings will be denoted by Δx, Δy, and Δz with corresponding difference operators D_{+x}, D_{+y}, D_{+z}, etc. The difference operators in time are D_{+t}, D_{-t}, and D_{0t}, where, for example, $D_{+t}U_\mathbf{i}^n = (U_\mathbf{i}^{n+1} - U_\mathbf{i}^n)/\Delta t$.

5. High-order accurate symmetric difference approximations on curvilinear grids. In some applications, such as long time integrations, it can be advantageous to use methods that are both high-order accurate and have low dissipation. To achieve these goals it will be helpful to use symmetric difference approximations to the discrete operators that are used in the solution of Maxwell's equations. Standard finite-difference approximations to the Laplace operator on curvilinear grids will not in general be symmetric. This section develops high-order accurate symmetric difference approximations for general logically rectangular curvilinear grids.

To understand the relationship between symmetric approximations and low dissipation schemes, consider the solution of the second-order wave equation,

$$w_{tt} = \nabla \cdot (c^2 \nabla w), \tag{5.1}$$

for the function $w = w(\mathbf{x}, t)$ on a periodic domain $\Omega = [0, 2\pi]^d$. Let $(u, v) = \int_\Omega u^* v \, d\mathbf{x}$ and $\|u\| = (u, u)^{1/2}$ denote the usual L_2 inner product and norm on this domain. The wave equation (5.1) has no dissipation, and there is an "energy" that remains

constant over time. This energy can be determined by multiplying the equation by w_t, integrating over the domain, and using integration by parts, giving

$$\tfrac{1}{2}\partial_t \left\{ \|w_t\|^2 + \|c\nabla w\|^2 \right\} = 0.$$

The "energy" $\mathcal{E} = \|w_t\|^2 + \|c\nabla w\|^2$ is thus constant on a periodic domain. Discrete approximations to the wave equation (5.1) will not in general have a discrete energy that remains constant. For stability reasons many approximations will have some dissipation built into the spatial approximation or into the time-stepping method. There is, however, a class of stable methods with no dissipation. Consider, for example, the second-order accurate centered approximation

$$D_{+t}D_{-t}W^n_{\mathbf{i}} = c^2(D_{+,x}D_{-,x} + D_{+,y}D_{-,y})W^n_{\mathbf{i}} \tag{5.2}$$

for use on a Cartesian grid, where $W^n_{\mathbf{i}} \approx w(\mathbf{x}_i, t^n)$. Let $(U,V)_h = \sum_{\mathbf{i}} U^*_{\mathbf{i}} V_{\mathbf{i}}\, \Delta x_1 \Delta x_2$ and $\|U\|_h = (U,U)_h^{1/2}$ denote the discrete L_2 inner product and norm on the Cartesian grid where the sum is over the grid points $i_m = 0, 1, \ldots, N_m - 1$. Equation (5.2) has a discrete "energy"

$$\mathcal{E}_h = \|D_{-t}W^n\|_h^2 + c^2(D_{-x}W^n, D_{-x}W^{n-1})_h + c^2(D_{-y}W^n, D_{-y}W^{n-1})_h$$

that is conserved, as will be shown below. Equation (5.2) is a special case of the more general discrete approximation

$$D_{+t}D_{-t}W^n = AW^n, \qquad n = 2, 3, 4, \ldots, \tag{5.3}$$

$$W^0 = W^{(0)}, \quad W^1 = W^{(1)} \quad \text{(initial conditions)}, \tag{5.4}$$

where $W^n \in \mathbb{R}^N$ is the vector of all N grid point values $W^n_{\mathbf{i}}$, and $A \in \mathbb{R}^{N\times N}$ is the matrix representing the spatial discretization (including eliminated boundary conditions). The three-level time-stepping schemes used in this paper are of this form (although for solutions on general overlapping grids a high-order dissipation is also added; see section 9.2). The following lemma indicates the conditions on A so that all solutions to (5.3) remain uniformly bounded in time.

LEMMA 5.1. [illegible] (5.3)–(5.4) [illegible] n [illegible] A [illegible] λ [illegible] A [illegible] Δt [illegible] $-4 < \lambda\Delta t^2 < 0$ [illegible] λ [illegible]. Suppose that $\widehat{V}$ is an eigenvector of A corresponding to the eigenvalue λ. Looking for a solution of the form $V^n = \kappa^n \widehat{V}$ implies that κ satisfies the characteristic equation $\kappa - 2 + \kappa^{-1} = \lambda\Delta t^2$ with

$$\kappa = 1 + \tfrac{1}{2}\lambda\Delta t^2 \pm \sqrt{(1 + \tfrac{1}{2}\lambda\Delta t^2)^2 - 1}. \tag{5.5}$$

When $\lambda = 0$ ($\kappa = 1$) or $\lambda\Delta t^2 = -4$ ($\kappa = -1$) there is a double root and the solutions can grow linearly in time, with $V^n = (c_0 + c_1(n\Delta t))\widehat{V}$ for $\lambda = 0$, and $V^n = (-1)^n\ (c_0 + c_1(n\Delta t))\widehat{V}$ for $\lambda\Delta t^2 = -4$. Therefore in these two cases the solutions will not be uniformly bounded for general initial conditions. A root, κ, of the characteristic equation will be a simple root and satisfy $|\kappa| = 1$ if and only if λ is real and $-4 < \lambda\Delta t^2 < 0$. In this case the solution will be of the form $V^n = (c_0\kappa^n + c_1\kappa^{-n})\widehat{V}$ and will remain uniformly bounded. If λ is not real or $\lambda > 0$ or $\lambda\Delta t^2 < -4$, there will be one root with $|\kappa| > 1$ and the approximation (5.3)

is not uniformly bounded. If all λ satisfy $-4 < \lambda\Delta t^2 < 0$ and there is a complete set of eigenvectors, then the general solution can be represented as a linear combination of the eigenvectors and this solution will remain uniformly bounded. If there is not a complete set of eigenvectors, then the Jordan normal form of A has a Jordan block of size larger than 1. In this case there are solutions of the form $V^n = (n\Delta t)\kappa^n$, and hence the solutions do not remain bounded in this case. □

The conditions of Lemma 5.1 will be satisfied when A is symmetric and the eigenvalues of A are less than zero. In this case a discrete energy estimate can be obtained as follows: Taking the inner product of $W^{n+1} - W^{n-1}$ with (5.3) and rearranging terms gives

$$\tfrac{1}{\Delta t^2}|W^{n+1} - W^n|^2 - \langle W^{n+1}, AW^n\rangle = \tfrac{1}{\Delta t^2}|W^n - W^{n-1}|^2 - \langle W^n, AW^{n-1}\rangle.$$

Here $\langle W, V\rangle = \sum_{i=1}^N W_i^* V_i$ and $|W| = \langle W, W\rangle^{\frac{1}{2}}$. It follows that the quantity

$$\mathcal{E}_h^n = \tfrac{1}{\Delta t^2}|W^n - W^{n-1}|^2 - \langle W^n, AW^{n-1}\rangle \tag{5.6}$$

$$= \tfrac{1}{\Delta t^2}\left\langle D_{+t}W^{n-1}, \left(I + \tfrac{\Delta t^2}{4}A\right)D_{+t}W^{n-1}\right\rangle - \langle W^{n-\frac{1}{2}}, AW^{n-\frac{1}{2}}\rangle \tag{5.7}$$

remains constant for all time, where $W^{n-\frac{1}{2}} := (W^n + W^{n-1})/2$. Note that two different time levels of the solution appear in the last term of (5.6). However, it can be seen from the alternative form (5.7) that $\mathcal{E}_h^n$ is positive provided Δt satisfies $-4 < \lambda\Delta t^2 < 0$ for all eigenvalues of A (see Kreiss, Petersson, and Ysström [28] or Cohen [4] for further details).

The above results provide a motivation for developing symmetric difference approximations for discretizations of Maxwell's equations. To this end, consider the generalized Laplace operator, L, defined by

$$Lw = \nabla \cdot (k\nabla w). \tag{5.8}$$

A straightforward approach to discretizing L on a curvilinear grid is to use the mapping method, as follows. Using the chain rule, the operator L can be written in general curvilinear coordinates (with summation convention) as

$$Lw = k(\mathbf{r})\frac{\partial r_n}{\partial x_i}\frac{\partial r_m}{\partial x_i}\frac{\partial^2 w}{\partial r_m \partial r_n} + \frac{\partial r_n}{\partial x_i}\left\{k(\mathbf{r})\frac{\partial}{\partial r_n}\left(\frac{\partial r_m}{\partial x_i}\right) + \frac{\partial k}{\partial r_n}\frac{\partial r_m}{\partial x_i}\right\}\frac{\partial w}{\partial r_m}. \tag{5.9}$$

The metric terms $\partial r_m/\partial x_n$ are computed from the mapping that defines the grid and are thus assumed to be known. The derivatives with respect to the parameter space coordinates r_m can be approximated with central difference approximations. Standard finite difference approximations, FDm, of order m for $m = 2, 4, 6, \ldots$, can be defined in this way. The resulting approximations will not, however, be symmetric on general curvilinear grids. High-order accurate symmetric approximations on rectangular grids and unstructured grids have been developed for this type of operator by various authors; see, for example, [4, 16]. The remainder of this section will show how to construct high-order accurate symmetric approximations to L for arbitrary curvilinear grids.

The operator L can be written in [illegible] form, or [illegible] form, in general curvilinear coordinates as

$$Lw = \tfrac{1}{J}\sum_{m=1}^{d}\sum_{n=1}^{d}\tfrac{\partial}{\partial r_m}\left(A^{mn}\tfrac{\partial w}{\partial r_n}\right), \quad A^{mn} = kJ\sum_{\mu=1}^{d}\tfrac{\partial r_m}{\partial x_\mu}\tfrac{\partial r_n}{\partial x_\mu}, \tag{5.10}$$

where J denotes the determinant of the Jacobian matrix $[\partial x_i/\partial r_j]$. To improve readability in the subsequent discussion, let $(r,s)=(r_1,r_2)$ and $(h_r,h_s)=(\Delta r_1,\Delta r_2)$. The discrete inner product on a curvilinear grid is defined as

$$(U_{\mathbf{i}},V_{\mathbf{i}})_h=\sum_{\mathbf{i}} U^*_{\mathbf{i}}V_{\mathbf{i}}\ J_{\mathbf{i}}\ h_rh_s. \tag{5.11}$$

Consider approximations to the two operators, $\mathcal{L}_{rr}=J^{-1}\partial_r(a\partial_r)$ and $\mathcal{L}_{rs}=J^{-1}\partial_r(b\partial_s)$, that are representative of the terms appearing in the expression (5.10) for L. The operator $\mathcal{L}_{rr}$ will be approximated in the discrete form $\mathcal{L}_{rr}w=J^{-1}\partial_r(a\partial_r w)\approx J^{-1}\mathcal{D}_+(a_{i-\frac12}\mathcal{D}_-)w(\mathbf{r_i})$, where $\mathcal{D}_+\approx\partial_r w(r+h_r/2)$ and $\mathcal{D}_-\approx\partial_r w(r-h_r/2)$ are some appropriately defined high-order accurate approximations to the derivatives at locations midway between grid points and $a_{i-\frac12}\approx a(r-h_r/2)$. If the adjoint of $J^{-1}\mathcal{D}_+$ with respect to (5.11) is $-J^{-1}\mathcal{D}_-$, then a symmetric approximation will result since

$$(v,J^{-1}\mathcal{D}_+(a_{i-\frac12}\mathcal{D}_-)w)_h=-(J^{-1}\mathcal{D}_-,a_{i-\frac12}\mathcal{D}_-w)_h=(J^{-1}\mathcal{D}_+(a_{i-\frac12}\mathcal{D}_-)v,w)_h.$$

To simplify notation let $D_+=D_{+,r}$ and $D_-=D_{-,r}$. An order $2m+2$ approximation for $\mathcal{D}_+$ is obtained from the expansion

$$\tfrac{\partial w}{\partial r}(r+h_r/2)=D_+\left[1+\sum_{n=1}^{m}\alpha_n h_r^{2n}(D_+D_-)^n\right]w(r+h_r/2)+\mathcal{O}(h_r^{2m+2}),$$

where the coefficients α_i can be determined from equating coefficients in the expression $y=\sin(y)\left[1+\sum_{n=1}^{\infty}\alpha_n(-4\sin^2(y))^n\right]$ (see, for example, [12] or [14]). The first four values of α_i are given by $\alpha_1=-\frac16$, $\alpha_2=\frac{3}{640}$, $\alpha_3=-\frac{5}{7168}$, and $\alpha_4=\frac{35}{294912}$. Assuming that the coefficient $a(r)$ is given at the grid points, a_i, approximations to $a_{i-1/2}$ will be needed. These approximations can be obtained by interpolation in a straightforward fashion and, to orders $m=2$ and $m=4$, are given by

$$a^{(2)}_{i-1/2}=\tfrac12(a_i+a_{i-1}),\quad a^{(4)}_{i-1/2}=\tfrac{9}{16}(a_i+a_{i-1})-\tfrac{1}{16}(a_{i+1}+a_{i-2}),$$

where $a^{(m)}_{i-1/2}=a_{i-1/2}+O(h_r^m)$. Note that other forms of interpolation may be appropriate when the coefficient k is discontinuous. Define $\mathcal{D}_+$ and $\mathcal{D}_-$ as

$$\mathcal{D}_\pm=D_\pm\left[1+\sum_{n=1}^{\infty}\alpha_n h_r^{2n}(D_+D_-)^n\right].$$

Expanding the expression $\mathcal{D}_+(a\mathcal{D}_-)$ in powers of h_r^2 leads to

$$\begin{aligned}\tfrac{\partial}{\partial r}\left(a\tfrac{\partial}{\partial r}\right)&=D_+(a^{(m)}_{i-1/2}D_-)-\tfrac{h_r^2}{24}\left[D_+(a^{(m-2)}_{i-1/2}D_+D_-^2)+D_+^2D_-(a^{(m-2)}_{i-1/2}D_-)\right]\\&+\tfrac{h_r^4}{24^2}\left[D_+^2D_-(a^{(m-4)}_{i-1/2}D_+D_-^2)\right]+\tfrac{3h_r^4}{640}\left[D_+(a^{(m-4)}_{i-1/2}D_+^2D_-^3)+D_+^3D_-^2(a^{(m-4)}_{i-1/2}D_-)\right]\\&-\tfrac{5h_r^6}{7168}\left[D_+^4D_-^3(a^{(m-6)}_{i-1/2}D_-)+D_+(a^{(m-6)}_{i-1/2}D_+^3D_-^4)\right]\\&-\tfrac{3h_r^6}{640\cdot24}\left[D_+^2D_-(a^{(m-6)}_{i-1/2}D_+^2D_-^3)+D_+^3D_-^2(a^{(m-6)}_{i-1/2}D_+D_-^2)\right]+O(h_r^8).\end{aligned} \tag{5.12}$$

Note that different orders of approximation for $a_{i-1/2}$ are used, so that the difference stencil for an mth-order formulation is no larger than $m+1$ points wide in each direction. Equation (5.12) can be used to generate symmetric approximations of

order m by involving terms up to and including those of order h_r^m. For example, a fourth-order symmetric approximation to $J^{-1}\partial_r(a\partial_r)$ is

$$\mathcal{L}_{rr}^{(4)}[a]w := J_{\mathbf{i}}^{-1}\Big\{D_+(a_{i-1/2}^{(4)}D_-) - \tfrac{h_r^2}{24}\big[D_+(a_{i-1/2}^{(2)}D_+D_-^2) + D_+^2D_-(a_{i-1/2}^{(2)}D_-)\big]\Big\}w.$$

It is easy to show that the approximation $\mathcal{L}_{rr}^{(4)}[a]$ is symmetric since the adjoint of $J^{-1}D_+$ is $-J^{-1}D_-$ with respect to the discrete inner product (5.11).

The operator $\mathcal{L}_{rs} = J^{-1}\partial_r(b\partial_s)$ for the mixed derivative terms is approximated using

(5.13)
$$\mathcal{L}_{rs} \approx J^{-1}D_{0r}\left[1 + \sum_{n=1}^{\infty}\beta_n h_r^{2n}(D_{+r}D_{-r})^n\right]\left\{bD_{0s}\left[1 + \sum_{n=1}^{\infty}\beta_n h_s^{2n}(D_{+s}D_{-s})^n\right]\right\},$$

where $D_{+r} = D_{+,r_1}$, $D_{+s} = D_{+,r_2}$, etc. and where $b = b_{\mathbf{i}}$ denotes the value of b at a grid point. In this case the values of b that appear are located at the grid points and no interpolation is needed. From the expansion of (5.13), a fourth-order approximation to $J^{-1}\partial_r(b\partial_s)$ is

$$\mathcal{L}_{rs}^{(4)}[b]w := J_{\mathbf{i}}^{-1}\Big\{D_{0r}(bD_{0s}) - \tfrac{1}{6}\big[h_s^2D_{0r}(bD_{+s}D_{-s}D_{0s}) + h_r^2D_{0r}D_{+r}D_{-r}(bD_{0s})\big]\Big\}w.$$

This operator is not symmetric, but the combination $\mathcal{L}_{rs}^{(4)}[b]+\mathcal{L}_{sr}^{(4)}[b]$ will be symmetric. A fourth-order symmetric approximation to $L = \nabla\cdot(k\nabla)$ denoted by $\mathcal{L}^{(4)}$ is thus

$$\mathcal{L}^{(4)} = \sum_{m=1}^{d}\sum_{n=1}^{d}\mathcal{L}_{r_mr_n}^{(4)}[A^{mn}],$$

where the coefficients A^{mn} are defined by (5.10). The symmetric approximation can be easily generalized to the case where $k(\mathbf{x})$ is replaced by a symmetric tensor; this only changes the definition of A^{mn}.

The symmetric approximations have been implemented in two and three space dimensions for orders $m = 2$, 4, 6, and 8. Figure 1 compares the accuracy of the symmetric difference operators of order m, denoted by SDm, to those of the standard finite-difference operators of order m, denoted by FDm. The maximum-norm errors in computing the Laplacian of a trigonometric function on a smooth nonorthogonal grid are presented in the figure for a sequence of grids of increasing resolution. The estimated convergence rate for each approximation is also indicated. These estimated rates are computed from a least squares fit to the log of the error versus the log of the grid spacing. The results show that the symmetric schemes, SDm, have similar errors to the finite-difference schemes, FDm. The convergence rates are reasonably close to the expected values. Note that the numerical values of the computed rates are sensitive to small changes in the data.

To illustrate that the discrete operators SDm are indeed symmetric on arbitrary logically rectangular grids, a computation was performed on the grid shown in Figure 2. The grid for the unit square, with periodic boundary conditions, is obtained by randomly perturbing the grid points of a Cartesian grid. The grid points are defined as $\mathbf{x}_{\mathbf{i}} = ((i_1+\mathcal{R}^{(x)}/4)\Delta x, (i_2+\mathcal{R}^{(y)}/4)\Delta y)$, where $\mathcal{R}^{(x)}$ and $\mathcal{R}^{(y)}$ are pseudorandom numbers uniformly distributed in $[-1, 1]$. Figure 2 shows the results from computations solving Maxwell's equations with $c = 1$ on a two-dimensional perturbed

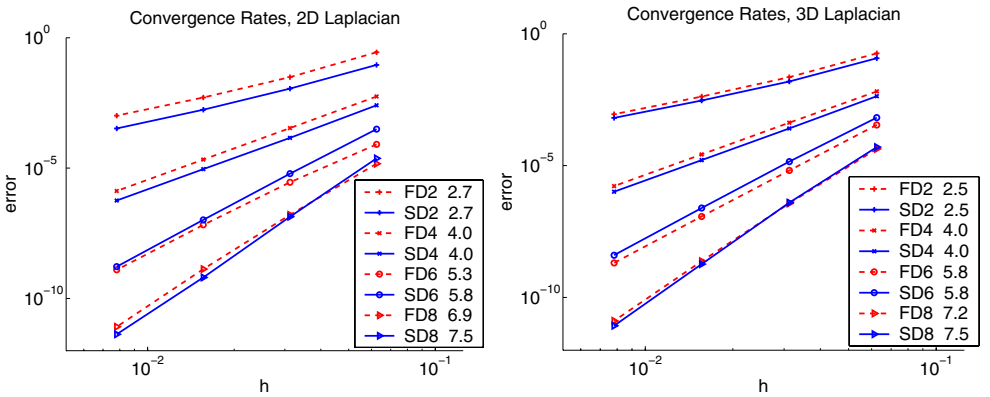


FIG. 1. *Convergence rates for the maximum-norm errors in computing the Laplacian on a smooth nonorthogonal grid. FDm denotes the standard finite-difference approximations of order m, while SDm denotes the symmetric difference approximations of order m. The estimated convergence rate for each approximation is indicated in the legend.*

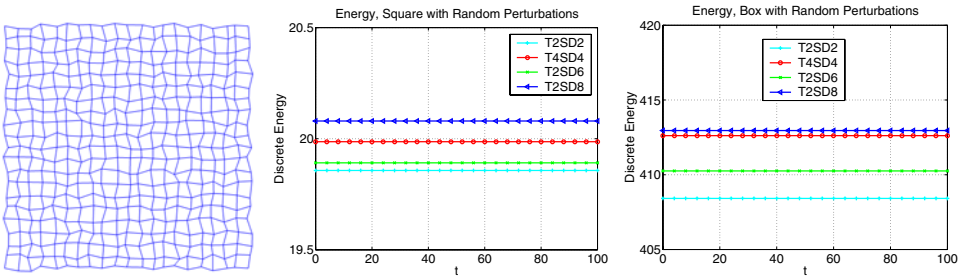


FIG. 2. *The computation of Maxwell's equations on a randomly perturbed Cartesian grid using the symmetric approximations. The discrete energy remains almost constant (to a relative error of about 10^{-10} in double precision) using the symmetric difference approximations, even on a non-smooth, nonorthogonal grid. Results for a perturbed grid for a square and a perturbed grid for a box are shown for the various schemes TmSDn of order of accuracy n in time and m in space.*

square. The discrete energy (5.6) for the component E_x (other components are similar) is plotted for different schemes for $t \in [0, 100]$. Let the symmetric time-stepping schemes be denoted by TnSDm, where n is the order of accuracy in time and m is the order of accuracy in space. Figure 2 presents results for the methods T2SD2, T4SD4, T2SD6, and T2SD8. The scheme T4SD4 is given by (3.2). The schemes T2SDm use second-order accurate time stepping with an mth order accurate approximation to the Laplacian, $U_{\mathbf{i}}^{n+1} - 2U_{\mathbf{i}}^{n} + U_{\mathbf{i}}^{n-1} = \Delta t^2 c^2 \Delta_{mh} U_{\mathbf{i}}^{n}$. The discrete energy is essentially constant over the computation (to within a relative error of about 10^{-10} in 64 bit double precision) for all schemes that use the symmetric operators. By comparison, schemes using the standard finite-difference approximations, denoted by TmFDn, are not energy preserving. These schemes are not stable for this grid without the addition of some dissipation. Figure 2 also shows the corresponding results for a perturbed three-dimensional box.

6. Analysis of the scheme. In this section the stability of the fourth-order accurate time-stepping scheme and boundary conditions is analyzed for a model problem. The analysis considers the solution to Maxwell's equations on the unit

square, $\Omega = [0,1]^2$, with PEC boundary conditions at $x = 0$ and $x = 1$ and periodic boundary conditions in y. A Cartesian grid is introduced with mesh points $\mathbf{x_i} = (x_i, y_j) = (i\Delta x, j\Delta y)$, where $\mathbf{i} = (i,j)$ and $i = -2, 1, 0, 1, 2, \ldots, N_1, N_1+1, N_1+2$ with $\Delta x = 1/N_1$, and $j = 0, 1, 2, \ldots, N_2 - 1$ with $\Delta y = 1/N_2$. Two lines of ghost points have been introduced outside the boundaries at $x = 0$ and $x = 1$. Discrete approximations to $E_x(\mathbf{x_i}, t^n)$ and $E_y(\mathbf{x_i}, t^n)$ are given by U_{ij}^n and V_{ij}^n, respectively, with $\mathbf{U_i^n} = (U_{\mathbf{i}}^n, V_{\mathbf{i}}^n)$. For this two-dimensional example the discretization for the magnetic field will not be presented since it is the same as the discretization for E_x. The interior equations (2.5) are discretized in a fourth-order manner using the modified equation approximation (3.2)

$$\mathbf{U_i^{n+1}} - 2\mathbf{U_i^n} + \mathbf{U_i^{n-1}} = (c\Delta t)^2 \Delta_{4h} \mathbf{U_i^n} + \tfrac{(c\Delta t)^4}{12} \Delta_{2h}^2 \mathbf{U_i^n} \quad \text{for} \quad i = 0, 1, \ldots, N_1, \quad j = 0, 1, \ldots, N_2 - 1. \tag{6.1}$$

Here Δ_{4h} is a fourth-order approximation to Δ, and Δ_{2h}^2 is a second-order approximation to Δ^2 given by

$$\Delta_{4h} \equiv D_{+x}D_{-x}\big(1 - \tfrac{\Delta x^2}{12} D_{+x}D_{-x}\big) + D_{+y}D_{-y}\big(1 - \tfrac{\Delta y^2}{12} D_{+y}D_{-y}\big),$$
$$\Delta_{2h}^2 \equiv \big(D_{+x}D_{-x} + D_{+y}D_{-y}\big)^2.$$

The divided difference operators D_{+x}, D_{-x}, D_{0x}, etc. were defined in section 4. The boundary conditions at $x = 0$ ($i = 0$) and $x = 1$ ($i = N_1$) are discretized as

$$V_{\mathbf{i}}^n = 0 \qquad \text{for } i = 0, N_1, \tag{6.2}$$

$$D_{0x}\big(1 - \tfrac{\Delta x^2}{6} D_{+x}D_{-x}\big) U_{\mathbf{i}}^n = 0 \qquad \text{for } i = 0, N_1, \tag{6.3}$$

$$D_{+x}D_{-x}\big(1 - \tfrac{\Delta x^2}{12} D_{+x}D_{-x}\big) V_{\mathbf{i}}^n = 0 \qquad \text{for } i = 0, N_1, \tag{6.4}$$

$$D_{0x}D_{+x}D_{-x} U_{\mathbf{i}}^n = 0 \qquad \text{for } i = 0, N_1, \tag{6.5}$$

$$\big(D_{+x}D_{-x}\big)^2 V_{\mathbf{i}}^n = 0 \qquad \text{for } i = 0, N_1, \tag{6.6}$$

where three additional [illegible] have been introduced. The boundary conditions (6.2)–(6.6) are, respectively, approximations to $E_y = 0$, fourth-order approximations to $\nabla \cdot \mathbf{E} = 0$, $\Delta E_y = 0$, and second-order approximations to $\nabla \cdot \Delta \mathbf{E} = 0$ and $\Delta^2 E_y = 0$. The latter three equations arise from the compatibility conditions (3.3). The scheme is completed by specifying initial values for two time levels, for example,

$$\mathbf{U_{ij}^0} = \mathbf{E}^{(0)}(\mathbf{x}_{ij}), \tag{6.7}$$

$$\mathbf{U_{ij}^1} = \mathbf{E}^{(0)}(\mathbf{x}_{ij}) + \Delta t \mathbf{E}_t^{(0)}(\mathbf{x}_{ij}) + \tfrac{\Delta t^2}{2} \Delta_{2h} \mathbf{E}^{(0)}(\mathbf{x}_{ij}) + \tfrac{\Delta t^3}{3!} \Delta_{2h} \mathbf{E}_t^{(0)}(\mathbf{x}_{ij}). \tag{6.8}$$

LEMMA 6.1. [illegible] (6.1) [illegible] (6.2)–(6.6) [illegible]

$$\Delta t < c^{-1} \left(\Delta x^{-2} + \Delta y^{-2}\right)^{-1/2}. \tag{6.9}$$

[illegible]. Making the ansatz $\widetilde{U}_{\mathbf{i}}^n = \kappa^n \cos(\alpha\pi x_i) e^{i2\pi\beta y_j}$ and $\widetilde{V}_{\mathbf{i}}^n = \kappa^n \sin(\alpha\pi x_i) e^{i2\pi\beta y_j}$, it follows that $\widetilde{U}_{\mathbf{i}}^n$ and $\widetilde{V}_{\mathbf{i}}^n$ satisfy (6.1) and boundary conditions (6.2)–(6.6) provided that κ satisfies the characteristic equation

$$\kappa - 2 + \kappa^{-1} = -4G, \tag{6.10}$$

where $\xi_x = \pi\alpha\Delta x$, $\xi_y = 2\pi\beta\Delta y$, $\sigma_x = \sin^2(\xi_x/2)$, $\sigma_y = \sin^2(\xi_y/2)$, $\lambda_x = c\frac{\Delta t}{\Delta x}$, $\lambda_y = c\frac{\Delta t}{\Delta y}$, and

$$G = \lambda_x^2\sigma_x[1+\tfrac{1}{3}\sigma_x] + \lambda_y^2\sigma_y[1+\tfrac{1}{3}\sigma_y] - \tfrac{1}{3}[\lambda_x^2\sigma_x + \lambda_y^2\sigma_y]^2,$$
$$\alpha = 0,1,2,\ldots,N_1, \quad \beta = 0,1,2,\ldots,N_2-1.$$

The two roots of the characteristic equation (6.10) are $\kappa_\pm = (1-2G)\pm\sqrt{(1-2G)^2-1}$, where $\kappa_+ = \kappa_-^{-1}$. The roots will satisfy $|\kappa_\pm| = 1$, and the solutions will remain uniformly bounded provided that $0 < G < 1$ (following the argument in the proof of Lemma 5.1). Letting $\gamma = \lambda_x^2\sigma_x + \lambda_y^2\sigma_y$, it follows that G can be written in the form

$$\begin{aligned} G &= \gamma - \tfrac{1}{3}\gamma^2 + \tfrac{1}{3}(\lambda_x^2\sigma_x^2 + \lambda_y^2\sigma_y^2) \\ &\le \gamma - \tfrac{1}{3}\gamma^2 + \tfrac{1}{3}(\lambda_x^2\sigma_x + \lambda_y^2\sigma_y) = \gamma + \tfrac{1}{3}\gamma(1-\gamma). \end{aligned}$$

The function G is always positive for $0 < \gamma < 1$. The function $\widetilde{G}(\gamma) = \gamma + \frac{1}{3}\gamma(1-\gamma)$ is less than 1 for $0 < \gamma \le 1$, and thus $0 < G < 1$ provided that $\gamma < 1$. Therefore the condition that $\widetilde{U}_{\mathbf{i}}^n$ and $\widetilde{V}_{\mathbf{i}}^n$ remain uniformly bounded is

$$\lambda_x^2\sigma_x + \lambda_y^2\sigma_y \le \lambda_x^2 + \lambda_y^2 < 1. \tag{6.11}$$

Let $\kappa_{\alpha,\beta}$ and $G_{\alpha,\beta}$ denote the values of κ_+ and G as functions of α and β. The general solution can be written as

$$U_{\mathbf{i}}^n = \sum_\alpha\sum_\beta (A_{\alpha,\beta}\kappa_{\alpha,\beta}^n + B_{\alpha,\beta}\kappa_{\alpha,\beta}^{-n})\cos(\alpha\pi x_{\mathbf{i}})e^{i2\beta\pi y_{\mathbf{i}}},$$
$$V_{\mathbf{i}}^n = \sum_\alpha\sum_\beta (C_{\alpha,\beta}\kappa_{\alpha,\beta}^n + D_{\alpha,\beta}\kappa_{\alpha,\beta}^{-n})\sin(\alpha\pi x_{\mathbf{i}})e^{i2\beta\pi y_{\mathbf{i}}},$$

where the values of $A_{\alpha,\beta}$, $B_{\alpha,\beta}$, $C_{\alpha,\beta}$, and $D_{\alpha,\beta}$ are determined by the initial conditions. The solution will be stable provided that $0 < G_{\alpha,\beta} < 1$ for all permissible α,β. It follows from the stability condition (6.11) and the definitions of λ_x and λ_y that the solution to the model problem is stable provided that the time step satisfies (6.9). □

Note that

$$\kappa_{\alpha,\beta} = 1 + ik\Delta t + \tfrac{1}{2}(ik\Delta t)^2 + \tfrac{1}{3!}(ik\Delta t)^3 + \tfrac{1}{4!}(ik\Delta t)^4 + O(\Delta t h^4, \Delta t^3 h^2, \Delta t^5),$$

where $h = \sqrt{\Delta x^2 + \Delta y^2}$, $k = \sqrt{k_x^2 + k_y^2}$, $k_x = \pi\alpha$, and $k_y = 2\pi\beta$. This shows that κ_{mn} is a fourth-order accurate approximation to $e^{ik\Delta t}$. The discrete eigenfunctions $\widetilde{U}_{\mathbf{i}}^n$ and $\widetilde{V}_{\mathbf{i}}^n$ are thus fourth-order accurate approximations to the eigenfunctions of the continuous problem, and it is apparent that the discrete approximation to the model problem is fourth-order accurate.

In three dimensions an argument similar to that used for the two-dimensional case shows that the stability condition is $\Delta t < c^{-1}\left(\Delta x^{-2} + \Delta y^{-2} + \Delta z^{-2}\right)^{-1/2}$.

6.1. Relationship to the Yee scheme. The current approach can be compared to the classic Yee scheme [44]. For the two-dimensional TE_Z system

$$u_t = w_y, \quad v_t = -w_x, \quad w_t = u_y - v_x,$$

the Yee scheme can be written as

$$D_{+t}U^n_{i+\frac{1}{2},j} = D_{+y}W^{n+\frac{1}{2}}_{i+\frac{1}{2},j-\frac{1}{2}}, \tag{6.12}$$

$$D_{+t}V^n_{i,j+\frac{1}{2}} = -D_{+x}W^{n+\frac{1}{2}}_{i-\frac{1}{2},j+\frac{1}{2}}, \tag{6.13}$$

$$D_{+t}W^{n-\frac{1}{2}}_{i+\frac{1}{2},j+\frac{1}{2}} = D_{-y}U^n_{i+\frac{1}{2},j} - D_{-x}V^n_{i,j+\frac{1}{2}}, \tag{6.14}$$

where the subscripts denote the staggering of the variables with $W^{n+\frac{1}{2}}_{i+\frac{1}{2},j+\frac{1}{2}}$, for example, being cell-centered. It is also true that $D_{+x}U^n_{i+\frac{1}{2},j} + D_{+y}V^n_{i,j+\frac{1}{2}} = 0$ (assuming the initial conditions satisfy this). A discrete second-order form for these equations can be found by first applying the operator D_{-t} to (6.12). Combining this with the other equations gives

$$\begin{aligned} D_{+t}D_{-t}U^n_{i+\frac{1}{2},j} &= D_{+y}D_{-y}U^n_{i+\frac{1}{2},j} - D_{+y}D_{+x}V^n_{i-\frac{1}{2},j+\frac{1}{2}} \\ &= (D_{+x}D_{+x} + D_{+y}D_{-y})U^n_{i+\frac{1}{2},j}, \end{aligned}$$

and similarly for the other variables. Thus each component satisfies the second-order wave equation discretized with the standard central difference operators. Neglecting boundaries, the Yee scheme therefore effectively uses the same discretization as the second-order accurate version of the current scheme.

7. Boundary conditions at material interfaces. In this section the propagation of electromagnetic waves through different dielectric (insulating) materials will be considered. The domain Ω is partitioned into a set of subdomains, $\Omega = \bigcup_k \Omega_k$, with each Ω_k representing a region with different material properties,

$$\epsilon(\mathbf{x}) = \epsilon_k, \quad \mu(\mathbf{x}) = \mu_k \quad \text{for } \mathbf{x} \in \Omega_k.$$

The assumption is made that ϵ and μ are constant within each subdomain (although this restriction is not essential). The boundaries of each subdomain, $\partial\Omega_k$, are assumed to be sufficiently smooth so that normal and tangent vectors to the interface can be defined. Overlapping grids will be used to discretize each subdomain separately. The subdomain boundaries will be represented with boundary fitted grids (see, for example, Figure 7). Maxwell's equations in second-order form are discretized in each subdomain using the high-order accurate difference approximations discussed in previous sections. Ghost points are used on the grids on both sides of the interface as shown in the one-dimensional overlapping grid of Figure 3. Boundary conditions at the interface are required for the numerical approximation. These boundary conditions are used to determine the solution values at the ghost points. As in the case of the PEC boundary, centered numerical boundary conditions are developed from compatibility conditions consistent with the governing equations. The compatibility conditions are based on interface jump conditions derived from the first-order form of Maxwell's equations. For $\rho = 0$, with spatially variable ϵ and μ, these are

$$\epsilon(\mathbf{x})\mathbf{E}_t = \nabla \times \mathbf{H}, \qquad \mu(\mathbf{x})\mathbf{H}_t = -\nabla \times \mathbf{E}, \tag{7.1}$$

$$\nabla \cdot (\epsilon(\mathbf{x})\mathbf{E}) = 0, \qquad \nabla \cdot (\mu(\mathbf{x})\mathbf{H}) = 0. \tag{7.2}$$

Under the assumption that $\mathbf{E}$ and $\mathbf{H}$ remain bounded, the basic jump conditions at a material interface $\mathcal{I}$ are derived by integrating each of the equations (7.1)–(7.2) over

an appropriate control volume that spans the interface (see, for example, [2]), and the result is

$$[\epsilon \mathbf{n}\cdot\mathbf{E}]_{\mathcal{I}}=0, \qquad [\mu \mathbf{n}\cdot\mathbf{H}]_{\mathcal{I}}=0, \tag{7.3}$$

$$[\boldsymbol{\tau}_m\cdot\mathbf{E}]_{\mathcal{I}}=0, \qquad [\boldsymbol{\tau}_m\cdot\mathbf{H}]_{\mathcal{I}}=0. \tag{7.4}$$

Here $[\mathbf{f}]_{\mathcal{I}}$ denotes the jump in $\mathbf{f}$ across the interface, $\mathbf{n}=\mathbf{n}(\mathbf{x})$ is the unit normal vector to the interface, and $\boldsymbol{\tau}_m=\boldsymbol{\tau}_m(\mathbf{x})$ is a unit tangent vector to the material interface. Since there are two linearly independent tangents in three dimensions, there will be two linearly independent conditions $[\boldsymbol{\tau}_m\cdot\mathbf{E}]=0$, $m=1,2$ (or just one condition in two dimensions). Jump conditions on the first spatial derivatives of the solution follow directly from equations (7.2), and also by taking one time derivative of (7.4) and combining with equations (7.1):

$$[\nabla\cdot\mathbf{E}]_{\mathcal{I}}=0, \qquad [\nabla\cdot\mathbf{H}]_{\mathcal{I}}=0, \tag{7.5}$$

$$[\mu^{-1}\boldsymbol{\tau}_m\cdot\nabla\times\mathbf{E}]_{\mathcal{I}}=0, \qquad [\epsilon^{-1}\boldsymbol{\tau}_m\cdot\nabla\times\mathbf{H}]_{\mathcal{I}}=0. \tag{7.6}$$

Note that $[\nabla\cdot(\epsilon\mathbf{E})]_{\mathcal{I}}=0$ can be replaced by $[\nabla\cdot\mathbf{E}]_{\mathcal{I}}=0$ since $\nabla\cdot\mathbf{E}$ is identically zero on either side of the interface. A similar remark applies to the condition $[\nabla\cdot\mathbf{H}]_{\mathcal{I}}=0$. Additional compatibility conditions can be derived by taking an even number of time derivatives of the four conditions (7.3)–(7.6) and using the vector wave equation (1.6) to transform the time derivatives into space derivatives. This gives the complete set of interface conditions

$$[\epsilon \mathbf{n}\cdot\Delta^n\mathbf{E}/(\epsilon\mu)^n]_{\mathcal{I}}=0, \qquad [\mu \mathbf{n}\cdot\Delta^n\mathbf{H}/(\epsilon\mu)^n]_{\mathcal{I}}=0, \tag{7.7}$$

$$[\boldsymbol{\tau}\cdot\Delta^n\mathbf{E}/(\mu\epsilon)^n]_{\mathcal{I}}=0, \qquad [\boldsymbol{\tau}\cdot\Delta^n\mathbf{H}/(\mu\epsilon)^n]_{\mathcal{I}}=0, \tag{7.8}$$

$$[\nabla\cdot(\Delta^n\mathbf{E})]_{\mathcal{I}}=0, \qquad [\nabla\cdot(\Delta^n\mathbf{H})]_{\mathcal{I}}=0, \tag{7.9}$$

$$[\mu^{-1}\boldsymbol{\tau}\cdot\nabla\times\Delta^n\mathbf{E}/(\mu\epsilon)^n]_{\mathcal{I}}=0, \qquad [\epsilon^{-1}\boldsymbol{\tau}\cdot\nabla\times\Delta^n\mathbf{H}/(\mu\epsilon)^n]_{\mathcal{I}}=0 \tag{7.10}$$

for $n=0,1,2,3,\ldots$. These interface jump conditions impose constraints on each spatial derivative of the solution.

Discrete versions of the jump conditions are used as numerical boundary conditions in the following manner. First consider the case of a second-order accurate approximation that uses one ghost line on each side of the interface. Let $\mathbf{U}^{(k)}_{\mathbf{i}}$ denote the discrete approximation to the electric field where $k=1,2$ denotes the solution on each side of the material interface. The dependence of the solution on time is suppressed for this discussion. To be specific, let the interface correspond to the value of $i_1=0$ and the ghost line values correspond to $i_1=-1$. The values $\mathbf{U}^{(1)}_{-1,i_2,i_3}$ and $\mathbf{U}^{(2)}_{-1,i_2,i_3}$ of the solution on the ghost lines (one ghost line value on each side of the interface) are determined by imposing the following centered interface conditions at $i_1=0$:

$$[\epsilon \mathbf{n}\cdot\Delta_{2h}\mathbf{U}^{(k)}_{\mathbf{i}}/(\epsilon\mu)]_{\mathcal{I}}=0, \qquad [\boldsymbol{\tau}_m\cdot\Delta_{2h}\mathbf{U}^{(k)}_{\mathbf{i}}/(\mu\epsilon)]_{\mathcal{I}}=0, \tag{7.11}$$

$$[\nabla_{2h}\cdot(\mathbf{U}^{(k)}_{\mathbf{i}})]_{\mathcal{I}}=0, \quad [\mu^{-1}\boldsymbol{\tau}_m\cdot\nabla_{2h}\times\mathbf{U}^{(k)}_{\mathbf{i}}]_{\mathcal{I}}=0. \tag{7.12}$$

Here $\nabla_{2h}\cdot$, $\nabla\times_{2h}$, and Δ_{2h} are some appropriate second-order accurate centered difference approximations. Equations (7.11)–(7.12) implicitly specify that the time derivatives of the jump conditions (7.3)–(7.4) are zero (to second-order accuracy). Therefore, by imposing the conditions (7.11)–(7.12) the basic jump conditions (7.3)–(7.4)

should also be approximately satisfied (assuming they are satisfied at the initial time). Therefore it may not be essential to explicitly impose the conditions (7.3)–(7.4). In practice, however, the solution values on the interface itself are constrained to satisfy the basic jump conditions,

$$[\epsilon \mathbf{n}\cdot \mathbf{U}_{\mathbf{i}}^{(k)}]_{\mathcal{I}}=0, \quad [\boldsymbol{\tau}_m\cdot \mathbf{U}_{\mathbf{i}}^{(k)}]_{\mathcal{I}}=0, \tag{7.13}$$

by setting the solution values on the side of the interface with the larger value of ϵ in terms of the values on the other side. For example, if $\epsilon_2 \le \epsilon_1$, then

$$\mathbf{U}^{(1)}_{0,i_2,i_3}=\frac{\epsilon_2}{\epsilon_1}\Big(\mathbf{n}\cdot \mathbf{U}^{(2)}_{0,i_2,i_3}\Big)\mathbf{n}+\sum_{m=1}^{2}\Big(\boldsymbol{\tau}_m\cdot \mathbf{U}^{(2)}_{0,i_2,i_3}\Big)\boldsymbol{\tau}_m. \tag{7.14}$$

A fourth-order accurate approximation will use two ghost lines on each side of the interface. The values at the ghost points are determined using fourth-order accurate approximations to the interface conditions for the first and second derivatives,

$$[\epsilon \mathbf{n}\cdot \Delta_{4h}\mathbf{U}_{\mathbf{i}}^{(k)}/(\epsilon\mu)]_{\mathcal{I}}=0, \qquad [\boldsymbol{\tau}_m\cdot \Delta_{4h}\mathbf{U}_{\mathbf{i}}^{(k)}/(\mu\epsilon)]_{\mathcal{I}}=0, \tag{7.15}$$

$$[\nabla_{4h}\cdot (\mathbf{U}_{\mathbf{i}}^{(k)})]_{\mathcal{I}}=0, \quad [\mu^{-1}\boldsymbol{\tau}_m\cdot \nabla_{4h}\times \mathbf{U}_{\mathbf{i}}^{(k)}]_{\mathcal{I}}=0, \tag{7.16}$$

together with second-order accurate approximations to the interface conditions for the third and fourth derivatives,

$$[(\nabla\cdot\Delta)_{2h}(\mathbf{U}_{\mathbf{i}}^{(k)})]_{\mathcal{I}}=0, \quad [\mu^{-1}\boldsymbol{\tau}_m\cdot (\nabla\times\Delta)_{2h}\mathbf{U}_{\mathbf{i}}^{(k)}/(\epsilon\mu)]_{\mathcal{I}}=0, \tag{7.17}$$

$$[\epsilon \mathbf{n}\cdot (\Delta^2)_{2h}\mathbf{U}_{\mathbf{i}}^{(k)}/(\epsilon\mu)^2]_{\mathcal{I}}=0, \qquad [\boldsymbol{\tau}_m\cdot (\Delta^2)_{2h}\mathbf{U}_{\mathbf{i}}^{(k)}/(\epsilon\mu)^2]_{\mathcal{I}}=0. \tag{7.18}$$

The fourth-order approximation will also impose equations (7.13). The accuracy and stability of these conditions for a model problem are considered in section 8.

A complicating factor in applying these interface boundary conditions is that the discrete equations couple the unknown values of $\mathbf{U}_{\mathbf{i}}^{(k)}$ at all ghost points on both sides of the interface. Therefore, the solution to these equations requires the solution of a system of equations with the number of unknowns proportional to the number of points on the interface. In practice, it has been found that these equations can be approximately solved using an iterative technique (see section 9.3 for further details).

8. Analysis of the material interface conditions. The accuracy and stability of the centered boundary conditions for material interfaces are considered in this section. A semidiscrete one-dimensional model problem is studied using the Laplace transform in time and mode analysis in space. Second-order accurate and fourth-order accurate interface conditions are analyzed.

A one-dimensional model problem for $u=u(x,t)$ and $v=v(x,t)$ on the interval $[a,b]$ with a material interface at $x=0$, $a<x<b$, is

$$\epsilon(x)u_{tt}=u_{xx}, \quad \epsilon(x)v_{tt}=v_{xx}, \tag{8.1}$$

$$[\epsilon u]_{\mathcal{I}}=0, \qquad [v]_{\mathcal{I}}=0, \tag{8.2}$$

$$[u_x]_{\mathcal{I}}=0, \qquad [v_x]_{\mathcal{I}}=0, \tag{8.3}$$

where $\epsilon(x)=\epsilon_1$ for $x<0$ and $\epsilon(x)=\epsilon_2$ for $x>0$. The variable u represents the normal component of the field (say, E_x), while the variable v represents one of the

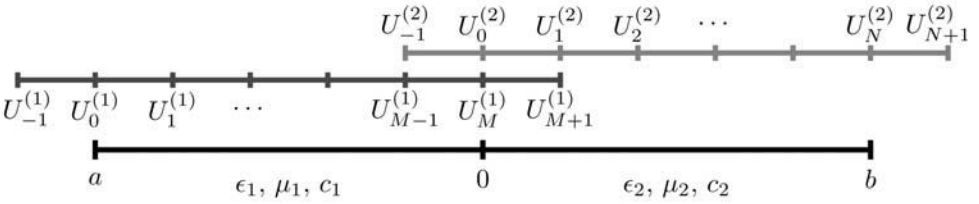


FIG. 3. *The overlapping grids for a one-dimensional material interface located at $x = 0$. Ghost points are introduced on the two grids at the interface.*

tangential components of the field (say E_y). Note that for this model problem the equations for the normal and tangential components are decoupled.

Introduce a one-dimensional overlapping grid as shown in Figure 3. A (formally) second-order accurate semidiscrete approximation is

$$\epsilon_1 U^{(1)}_{tt} = D_+D_-U^{(1)}_j, \quad \epsilon_1 V^{(1)}_{tt} = D_+D_-V^{(1)}_j \quad \text{for } j = M, M-1, M-2, \dots, \tag{8.4}$$

$$\epsilon_2 U^{(2)}_{tt} = D_+D_-U^{(2)}_j, \quad \epsilon_2 V^{(2)}_{tt} = D_+D_-V^{(2)}_j \quad \text{for } j = 0, 1, 2, \dots \tag{8.5}$$

with interface conditions

$$D_0U^{(1)}_M = D_0U^{(2)}_0, \qquad D_0V^{(1)}_M = D_0V^{(2)}_0, \tag{8.6}$$

$$D_+D_-U^{(1)}_M = D_+D_-U^{(2)}_0, \quad D_+D_-V^{(1)}_M/\epsilon_1 = D_+D_-V^{(2)}_0/\epsilon_2. \tag{8.7}$$

Here $U^{(k)}_i(t)$ and $V^{(k)}_i(t)$ denote the approximations to u and v on the grids while $D_+ = D_{+x}$, $D_- = D_{-x}$, and $D_0 = D_{0x}$. The grid points are denoted by $\mathbf{x}^{(k)}_i$. Approximations to the jump conditions $[u_{xx}]_{\mathcal{I}} = 0$ and $[v_{xx}/\epsilon]_{\mathcal{I}}$ have been imposed in place of (8.2); these follow from (8.1)–(8.2). The interface equations (8.6) and (8.7) can be solved for the values at the ghost points, $U^{(1)}_{M+1}, U^{(2)}_{-1}$, in terms of interior values:

$$U^{(1)}_{M+1} = U^{(2)}_1 + (U^{(1)}_M - U^{(2)}_0) = U^{(2)}_1 + (\epsilon_2/\epsilon_1 - 1)U^{(2)}_0, \tag{8.8}$$

$$U^{(2)}_{-1} = U^{(1)}_{M-1} - (U^{(1)}_M - U^{(2)}_0) = U^{(1)}_{M-1} - (1 - \epsilon_1/\epsilon_2)U^{(1)}_M, \tag{8.9}$$

$$V^{(1)}_{M+1} = \frac{2\epsilon_1}{\epsilon_1+\epsilon_2}\, V^{(2)}_1 + \frac{\epsilon_2-\epsilon_1}{\epsilon_1+\epsilon_2}\big(2V^{(1)}_M - V^{(1)}_{M-1}\big), \tag{8.10}$$

$$V^{(2)}_{-1} = \frac{2\epsilon_2}{\epsilon_1+\epsilon_2}\, V^{(1)}_{M-1} + \frac{\epsilon_1-\epsilon_2}{\epsilon_1+\epsilon_2}\big(2V^{(2)}_0 - V^{(2)}_1\big). \tag{8.11}$$

The interface boundary conditions have the useful property that when $\epsilon_1 = \epsilon_2$, the solution to the interface problem is the same as if there were no interface at all.

The stability of the discrete approximations to the interface problem will now be analyzed. A necessary condition for stability is the Godunov–Ryabenkii condition, which requires that there be no solutions to the following eigenvalue problem for $\mathrm{Re}(s) > 0$:

$$\epsilon_1 s^2 W^{(1)}_j = D_+D_-W^{(1)}_j \qquad \text{for } j = M, M-1, M-2, \dots, \tag{8.12}$$

$$\epsilon_2 s^2 W^{(2)}_j = D_+D_-W^{(2)}_j \qquad \text{for } j = 0, 1, 2, \dots, \tag{8.13}$$

$$D_0W^{(1)}_M = D_0W^{(2)}_0, \qquad (D_+D_-W^{(1)}_M)/\epsilon^{\alpha}_1 = (D_+D_-W^{(2)}_0)/\epsilon^{\alpha}_2, \tag{8.14}$$

$$\|W^{(1)}\|_h < \infty, \qquad \|W^{(2)}\|_h < \infty. \tag{8.15}$$

Here $\alpha = 0$ corresponds to the problem for the normal component $U_i^{(k)}$, while $\alpha = 1$ corresponds to the problem for the tangential component $V_i^{(k)}$. The discrete norms of the grid functions are defined from $\|W^{(1)}\|_h^2 = \sum_{j=-\infty}^{M} |W_j^{(1)}|^2\ h$ and $\|W^{(2)}\|_h^2 = \sum_{j=0}^{\infty} |W_j^{(2)}|^2\ h$. If this problem (8.12)–(8.15) were to have a solution, there would be a solution to (8.4)–(8.7) of the form $U_i^{(k)}(t) = e^{st} W_i^{(k)}$ that would grow arbitrarily fast in time as $h \to 0$, and the scheme would not be stable.

LEMMA 8.1. [illegible] (8.4)–(8.7) [illegible]. The solution to the eigenvalue problem (8.12)–(8.15) is of the form

$$W_j^{(1)} = A\kappa_1^{M-j}, \quad W_j^{(2)} = B\kappa_2^j,$$

where $\kappa_k = 1 + s_k/2 - \sqrt{s_k}\sqrt{1 + s_k/4}$ is a root of the characteristic equation $\kappa - 2 + \kappa^{-1} = s_k$, with $s_k = \epsilon_k s^2 h^2$, for $k = 1, 2$. The branch of the square root is taken so that $|\kappa_1| < 1$ and $|\kappa_2| < 1$ for $\mathrm{Re}(s) > 0$. Applying the interface conditions (8.8) and (8.9) implies

$$\begin{aligned} A(\kappa_1^{-1} - \kappa_1) &= B(\kappa_2 - \kappa_2^{-1}), \\ A(\kappa_1^{-1} - 2 + \kappa_1)/\epsilon_1^{\alpha} &= B(\kappa_2 - 2 + \kappa_2^{-1})/\epsilon_2^{\alpha}. \end{aligned}$$

For the case $\alpha = 0$ the determinant condition for nontrivial solutions is

$$(\kappa_1^{-1} - \kappa_1)(\kappa_2 - 2 + \kappa_2^{-1}) = (\kappa_2 - \kappa_2^{-1})(\kappa_1^{-1} - 2 + \kappa_1),$$

which simplifies to $\kappa_1\kappa_2 = 1$. This contradicts the fact that $|\kappa_1\kappa_2| < 1$ and thus there can be no nontrivial solutions. For $\alpha = 1$ it follows that $A = B$ and for a nontrivial solution, $-(\kappa_1 - \kappa_1^{-1}) = \kappa_2 - \kappa_2^{-1}$. From the definitions of κ_1 and κ_2 it follows that $\sqrt{s_1}\sqrt{1 + s_1/4} = -\sqrt{s_2}\sqrt{1 + s_2/4}$. The solution to the last equation is $s_1 = -s_2 - 4$ (the other possible solution $s_1 = s_2$ is on the wrong branch of the square root), which implies $s = \pm i/(h\sqrt{\epsilon_1 + \epsilon_2})$. This contradicts the assumption that $Re(s) > 0$, and thus there are no nontrivial solutions for $\alpha = 1$. It thus follows that the Godunov–Ryabenkii condition is satisfied. □

The accuracy of the approximation can be determined by examining the problem for the error, $E_i^{(k)} = U_i^{(k)} - u(\mathbf{x}_i^{(k)}, t)$:

$$\begin{aligned} \epsilon_1 s^2 E_j^{(1)} &= D_+ D_- E_j^{(1)} + h^2 F_1 \qquad \text{for } j = M, M-1, M-2, \ldots, \\ \epsilon_2 s^2 E_j^{(2)} &= D_+ D_- E_j^{(2)} + h^2 F_2 \qquad \text{for } j = 0, 1, 2, \ldots, \\ D_0 E_M^{(1)} - D_0 E_0^{(2)} &= h^2 G_1, \qquad (D_+ D_- E_M^{(1)})/\epsilon_1^{\alpha} - (D_+ D_- E_0^{(2)})/\epsilon_2^{\alpha} = h^2 G_2, \end{aligned}$$

where the terms h^2F_1, h^2F_2, h^2G_1, and h^2G_2 are the truncation errors. In the usual manner (for example, see [14]), by subtracting out a solution to the Cauchy problem it may be assumed that $F_1 = F_2 = 0$. The solution to the error equations is then of the form

$$E_j^{(1)} = A\kappa_1^{M-j}, \quad E_j^{(2)} = B\kappa_2^j,$$

and applying the interface conditions gives the equations for A and B,

$$\begin{bmatrix} (\kappa_1^{-1} - \kappa_1)/(2h) & -(\kappa_2 - \kappa_2^{-1})/(2h) \\ \epsilon_1^{1-\alpha} s^2 & -\epsilon_2^{1-\alpha} s^2 \end{bmatrix} \begin{bmatrix} A \\ B \end{bmatrix} = \begin{bmatrix} h^2 G_1 \\ h^2 G_2 \end{bmatrix}.$$

Using $\kappa_k - \kappa_k^{-1} = -2\sqrt{s_k}\sqrt{1+s_k/4}$, it follows that

$$A = \frac{\epsilon_1^{1-\alpha} s G_1 + \epsilon_2^{1/2} G_2}{\epsilon_1^{1-\alpha}(\epsilon_1^{1/2}+\epsilon_2^{1/2})s^2}\, h^2 + O(h^4), \quad B = \frac{\epsilon_1^{1-\alpha} s G_1 - \epsilon_1^{1/2} G_2}{\epsilon_1^{1-\alpha}(\epsilon_1^{1/2}+\epsilon_2^{1/2})s^2}\, h^2 + O(h^4).$$

Therefore the solution to the semidiscrete problem is second-order accurate.

Now consider the fourth-order accurate approximation

$$\epsilon_1 U_{tt}^{(1)} = D_+D_-(1 - \tfrac{h^2}{12}D_+D_-)U_j^{(1)} \quad \text{for } j = M, M-1, M-2, \dots, \tag{8.16}$$

$$\epsilon_2 U_{tt}^{(2)} = D_+D_-(1 - \tfrac{h^2}{12}D_+D_-)U_j^{(2)} \quad \text{for } j = 0, 1, 2, \dots, \tag{8.17}$$

$$D_0(1 - \tfrac{h^2}{6}D_+D_-)U_M^{(1)} = D_0(1 - \tfrac{h^2}{6}D_+D_-)U_0^{(2)}, \tag{8.18}$$

$$D_+D_-(1 - \tfrac{h^2}{12}D_+D_-)U_M^{(1)}/\epsilon_1^{\alpha} = D_+D_-(1 - \tfrac{h^2}{12}D_+D_-)U_0^{(2)}/\epsilon_2^{\alpha}, \tag{8.19}$$

$$D_0D_+D_-U_M^{(1)}/\epsilon_1 = D_0D_+D_-U_0^{(2)}/\epsilon_2, \tag{8.20}$$

$$(D_+D_-)^2U_M^{(1)}/\epsilon_1^{1+\alpha} = (D_+D_-)^2U_0^{(2)}/\epsilon_2^{1+\alpha}, \tag{8.21}$$

where $\alpha = 0$ corresponds to the interface conditions for the normal component and $\alpha = 1$ corresponds to the conditions for the tangential component. Two ghost points have been added to the grids on each side of the interface. Appropriate initial conditions are also required to complete the specification of the problem; their precise form is not germane to the following discussion. When $\epsilon_1 = \epsilon_2$, the solution to the interface problem reduces to the solution of the equations with no interface present since in this case $U_{M+m}^{(1)} = U_m^{(2)}$ and $U_{-m}^{(2)} = U_{M+m}^{(1)}$ for $m = 1, 2$.

The Godunov–Ryabenkii condition, necessary for the stability of solutions to these difference equations, requires that there be no solutions with $\mathrm{Re}(s) > 0$ to the eigenvalue problem

$$\epsilon_1 s^2 W_j^{(1)} = D_+D_-(1 - \tfrac{h^2}{12}D_+D_-)W_j^{(1)} \quad \text{for } j = M, M-1, M-2, \dots,$$

$$\epsilon_2 s^2 W_j^{(2)} = D_+D_-(1 - \tfrac{h^2}{12}D_+D_-)W_j^{(2)} \quad \text{for } j = 0, 1, 2, \dots,$$

$$D_0(1 - \tfrac{h^2}{6}D_+D_-)W_M^{(1)} = D_0(1 - \tfrac{h^2}{6}D_+D_-)W_0^{(2)},$$

$$D_+D_-(1 - \tfrac{h^2}{12}D_+D_-)W_M^{(1)}/\epsilon_1^{\alpha} = D_+D_-(1 - \tfrac{h^2}{12}D_+D_-)W_0^{(2)}/\epsilon_2^{\alpha},$$

$$D_0D_+D_-W_M^{(1)}/\epsilon_1 = D_0D_+D_-W_0^{(2)}/\epsilon_2,$$

$$(D_+D_-)^2W_M^{(1)}/\epsilon_1^{1+\alpha} = (D_+D_-)^2W_0^{(2)}/\epsilon_2^{1+\alpha},$$

$$\|W^{(1)}\|_h < \infty, \qquad \|W^{(2)}\|_h < \infty.$$

LEMMA 8.2. [illegible] (8.16)–(8.21) [illegible] $sh \ll 1$ [illegible]. For $\mathrm{Re}(s) > 0$ the bounded solutions may be written as

$$W_j^{(1)} = A_1\kappa_1^{M-j} + B_1\widetilde{\kappa}_1^{M-j}, \quad W_j^{(2)} = A_2\kappa_2^{j} + B_2\widetilde{\kappa}_2^{j},$$

where κ_k are the roots approximating the solution to the continuous problem and $\widetilde{\kappa}_k$ are the spurious roots.[1] These are roots of the characteristic equation

$$(\kappa - 2 + \kappa^{-1})(1 - \tfrac{1}{12}(\kappa - 2 + \kappa^{-1})) = s_k,$$

[1] This form must be modified to use $j\kappa_k^j$ instead of $\widetilde{\kappa}^j$ when there is a double root, $\kappa_k = \widetilde{\kappa}_k$.

where $s_k = \epsilon_k s^2 h^2$. In particular

$$\kappa_k = 1 + z_k/2 - \sqrt{z_k}\sqrt{1 + z_k/4}, \quad \widetilde{\kappa}_k = 1 + \widetilde{z}_k/2 - \sqrt{\widetilde{z}_k}\sqrt{1 + \widetilde{z}_k/4},$$
$$z_k = 6(1 - \sqrt{1 - s_k/3}), \qquad \widetilde{z}_k = 6(1 + \sqrt{1 - s_k/3}),$$

where the branch of the square root is chosen so that $|\kappa_k| < 1$ and $|\widetilde{\kappa}_k| < 1$ for $Re(s) > 0$. Note that for $s_k = 3$ there is a double root $\kappa_k = \widetilde{\kappa}_k$, but this occurs when sh is order 1 and thus will not be considered. For $sh \ll 1$, the roots take the form

$$\kappa_k = 1 - \sqrt{\epsilon_k} sh + \epsilon_k (sh)^2/2 - (\sqrt{\epsilon_k} sh)^3/6 + (\sqrt{\epsilon_k} sh)^4/24 + O((sh)^5),$$
$$\widetilde{\kappa}_k = \frac{1}{7 + 4\sqrt{3}} + O(sh).$$

It can be seen that κ_k is a fourth-order approximation to $e^{-\sqrt{\epsilon_k} sh}$. Substitution of $W_j^{(1)}$ and $W_j^{(1)}$ into the four boundary conditions implies

$$\begin{bmatrix} \mathcal{D}_1(\kappa_1) & \mathcal{D}_1(\widetilde{\kappa}_1) & \mathcal{D}_1(\kappa_2) & \mathcal{D}_1(\widetilde{\kappa}_2) \\ s_1/\epsilon_1^{\alpha} & s_1/\epsilon_1^{\alpha} & -s_2/\epsilon_2^{\alpha} & -s_2/\epsilon_2^{\alpha} \\ \mathcal{D}_3(\kappa_1)/\epsilon_1 & \mathcal{D}_3(\widetilde{\kappa}_1)/\epsilon_1 & \mathcal{D}_3(\kappa_2)/\epsilon_2 & \mathcal{D}_3(\widetilde{\kappa}_2)/\epsilon_2 \\ \mathcal{D}_4(\kappa_1)/\epsilon_1^{1+\alpha} & \mathcal{D}_4(\widetilde{\kappa}_1)/\epsilon_1^{1+\alpha} & -\mathcal{D}_4(\kappa_2)/\epsilon_2^{1+\alpha} & -\mathcal{D}_4(\widetilde{\kappa}_2)/\epsilon_2^{1+\alpha} \end{bmatrix} \begin{bmatrix} A_1 \\ B_1 \\ A_2 \\ B_2 \end{bmatrix} = \mathbf{0}, \tag{8.22}$$

where $\mathcal{D}_1(\kappa) = \frac{1}{2}(\kappa - \kappa^{-1})(1 - \frac{1}{6}(\kappa - 2 + \kappa^{-1}))$, $\mathcal{D}_3(\kappa) = \frac{1}{2}(\kappa - \kappa^{-1})(\kappa - 2 + \kappa^{-1})$, and $\mathcal{D}_4(\kappa) = (\kappa - 2 + \kappa^{-1})^2$. Let $\mathcal{M}$ denote the matrix appearing in equation (8.22). Note that $\kappa_k - \kappa_k^{-1} = -2\sqrt{z_k}\sqrt{1 + z_k/4}$ and $\kappa_k - 2 + \kappa_k^{-1} = z_k$, which can be used to simplify the entries in $\mathcal{M}$. For stability there should be no values of s with $Re(s) > 0$ when the determinant of $\mathcal{M}$ is zero. This condition is difficult to prove for general values of sh. For $sh \ll 1$, corresponding to the resolved modes in the discrete solution, the elements of the determinant can be expanded in powers of sh and one finds that for $sh \to 0$,

$$\det(\mathcal{M}) \sim -2^9 3^3 \sqrt{3}(\epsilon_1^{-1/2} + \epsilon_2^{-1/2})\ (sh)^3 \qquad \text{for } \alpha = 0, \tag{8.23}$$

$$\det(\mathcal{M}) \sim -2^8 3^3 \sqrt{3} \frac{1}{\epsilon_1^2 \epsilon_2^2}(\epsilon_1^{1/2} + \epsilon_2^{1/2})(\epsilon_1 + \epsilon_2)\ (sh)^3 \quad \text{for } \alpha = 1. \tag{8.24}$$

Therefore there are no unstable modes for $sh \ll 1$ with $Re(s) > 0$. This proves the lemma. □

The stability of the interface boundary conditions was investigated further by computing the reflection and transmission of an electromagnetic wave from a straight interface. The computational domain is the region $\Omega = [-1, 1] \times [0, 1]$, with the interface located at $x = 0$. A Cartesian grid, $\mathcal{G}_1$, with $N_1 \times M_1$ grid cells covers the left half of the domain, $\Omega_1 = [-1, 0] \times [0, 1]$, where the material properties are ϵ_1 and μ_1. Another Cartesian grid, $\mathcal{G}_2$, with $N_2 \times M_2$ grid cells covers the right half of the domain, $\Omega_2 = [0, 1] \times [0, 1]$, where the material properties are ϵ_2 and μ_2. The incident field was chosen to be $(E_x, E_y) = (-k_2, k_1)/|\mathbf{k}| \exp(2\pi i(\mathbf{k}\cdot\mathbf{x} - \omega t))$, where $\mathbf{k} = (k_1, k_2)$ and $\omega/|\mathbf{k}| = c_1$. The reflected and transmitted fields are easily determined [2]. The solution was evaluated using the fourth-order modified equation method (6.1) and the fourth-order accurate interface boundary conditions (7.15)–(7.18). See section 9.3 for further details of the implementation. Periodic boundary conditions were chosen at $y = 0$ and $y = 1$, and the exact solution was given as a boundary condition at $x = -1$ and $x = 1$. Figure 4 shows results for the case $\epsilon_1 = 1$, $\mu_1 = 1$ ($c_1 = 1$), $\epsilon_2 = 4$, $\mu_2 = 1$

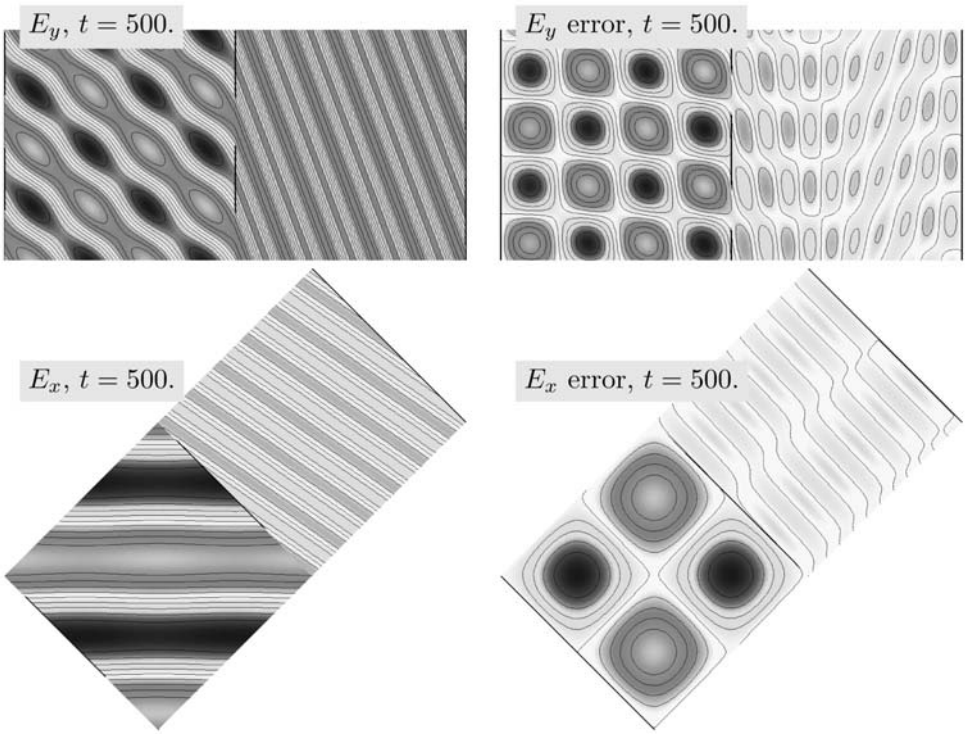


FIG. 4. *The reflection and transmission of a plane wave by a straight interface computed with the fourth-order accurate method. Top left: contours of* E_y *at* $t = 500$. *Top right: the error in* E_y *at* $t = 500$ *(the maximum error is* .01*). Bottom: the domain is rotated by* 45°. *Bottom left: contours of* E_x *at* $t = 500$. *Bottom right: the error in* E_x *at* $t = 500$ *(the maximum error is* .05*).*

$(c_2 = 1/2)$, $\mathbf{k} = (2, 2)$, and $N_1 = M_1 = N_2 = M_2 = 80$. The solution was computed to time $t = 500$ (approximately 60000 time steps). The solution remained stable as evidenced by the smooth behavior of the solution and error. Figure 4 shows similar results for a domain rotated by 45 degrees using $\mathbf{k} = (1, 1)$, $\epsilon_1 = 1$, $\mu_1 = 1$ $(c_1 = 1)$, $\epsilon_2 = 16$, $\mu_2 = 1$ $(c_2 = 1/4)$, and $N_1 = M_1 = N_2 = M_2 = 80$. The solution remained stable in this case as well.

For the long-time nondissipative calculations of Figure 4, the material interface conditions must be solved accurately to avoid a late time instability. More accuracy is required the larger the number of time steps. For accuracy reasons alone, only a few iterations are needed. For the problem of scattering by a dielectric cylinder of section 10.3, when dissipation is added to the equations, only a few iterations are needed to solve the interface conditions since the dissipation suppresses this weak instability. If the need arises, the interface equations should be amenable to solution by a Krylov-based method.

9. Implementation of the scheme. Some aspects of the implementation of the discrete approximations to the differential operators and boundary conditions will be considered in this section. The intent of these remarks is to give a basic outline of the implementation; a more detailed description is left for a future report. In typical cases the majority of grid points on an overlapping grid belong to Cartesian grids where the discrete approximations are straightforward to implement. The Cartesian

grid approximations are very fast to evaluate and use little memory. The majority of the programming effort, however, is concerned with implementing the approximations to the boundary conditions and material interface conditions on curvilinear grids.

9.1. Solving the PEC boundary conditions. The solution of the discrete approximations to the boundary conditions for a PEC are based upon the boundary conditions $\mathbf{n}\times\mathbf{E}=0$ and $\nabla\cdot\mathbf{E}=0$, together with the compatibility conditions (3.3). To simplify the implementation, some approximations are made that do not seem to affect the overall accuracy, as evidenced by the numerical results from section 10. The divergence of $\mathbf{E}$ can be written in general curvilinear coordinates as

$$\nabla\cdot\mathbf{E}=J^{-1}\sum_{m=1}^{d}\tfrac{\partial}{\partial r_m}(\mathbf{a}_m\cdot\mathbf{E}),\quad \mathbf{a}_m=J\nabla_{\mathbf{x}}r_m=J(\partial_x r_m,\partial_y r_m,\partial_z r_m).$$

For a PEC boundary, corresponding to the parameter space line $r_1=0$, the boundary condition $\nabla\cdot\mathbf{E}=0$ becomes a condition relating the r_1-derivative of $\mathbf{a}_1\cdot\mathbf{E}=-J\|\nabla_{\mathbf{x}}r_1\|\mathbf{n}\cdot\mathbf{E}$ to tangential derivatives of the components $\mathbf{a}_2\cdot\mathbf{E}_2$ and $\mathbf{a}_3\cdot\mathbf{E}$, $\partial_{r_1}(\mathbf{a}_1\cdot\mathbf{E})=-\partial_{r_2}(\mathbf{a}_2\cdot\mathbf{E}_2)-\partial_{r_3}(\mathbf{a}_3\cdot\mathbf{E})$. For an orthogonal grid, $\mathbf{a}_2\cdot\mathbf{E}(0,r_2,r_3)=0$ and $\mathbf{a}_3\cdot\mathbf{E}(0,r_2,r_3)=0$ on the boundary since $\mathbf{a}_2$ and $\mathbf{a}_3$ are in the tangent plane. In addition, $\mathbf{a}_1\cdot\mathbf{E}$ is (always) proportional to the normal component of $\mathbf{E}$, and thus the $\nabla\cdot\mathbf{E}=0$ boundary condition reduces to a normal derivative of this component:

$$J\ \nabla\cdot\mathbf{E}=\partial_{r_1}(\mathbf{a}_1\cdot\mathbf{E})\quad\text{for } r_1=0\quad\text{(orthogonal grid)}.$$

For general grids, $\mathbf{a}_2\cdot\mathbf{E}_2$ and $\mathbf{a}_3\cdot\mathbf{E}$ are not zero. A second-order accurate approximation to the boundary conditions at $i_1=0$ is

$$\boldsymbol{\tau}_m\cdot\mathbf{U}_{\mathbf{i}}=0, \tag{9.1}$$

$$D_{0,r_1}(\mathbf{a}_1\cdot\mathbf{U}_{\mathbf{i}})=-D_{0,r_2}(\mathbf{a}_2\cdot\mathbf{U}_{\mathbf{i}})-D_{0,r_3}(\mathbf{a}_3\cdot\mathbf{U}_{\mathbf{i}}), \tag{9.2}$$

$$\boldsymbol{\tau}_m\cdot D_{+r}D_{-r}\mathbf{U}_{\mathbf{i}}=0, \tag{9.3}$$

where $\boldsymbol{\tau}_m$, $m=1,2$, are tangent vectors to the boundary (there is only one in two dimensions). The equations (9.2)–(9.3) determine the solution on the ghost point, $\mathbf{U}_{-1,i_2,i_3}$. Equation (9.3) approximates the more accurate compatibility condition $\boldsymbol{\tau}_m\cdot\Delta\mathbf{E}=0$. These equations can be solved without iteration since the terms appearing on the right-hand side of (9.2) involve known values on the boundary once (9.1) is applied.

Let $D^{(4)}_{r_m}=D_{0,r_m}(1-\frac{1}{12}\Delta {r_1}^2 D_{+,r_m}D_{-,r_m})$ denote the fourth-order accurate approximation to $\partial/\partial r_m$. A fourth-order accurate approximation to the boundary conditions at $i_1=0$ is

$$\boldsymbol{\tau}_m\cdot\mathbf{U}_{\mathbf{i}}=0, \tag{9.4}$$

$$D^{(4)}_{r_1}(\mathbf{a}_1\cdot\mathbf{U})_{\mathbf{i}}=-D^{(4)}_{r_2}(\mathbf{a}_2\cdot\mathbf{U}_{\mathbf{i}})-D^{(4)}_{r_3}(\mathbf{a}_3\cdot\mathbf{U}_{\mathbf{i}}), \tag{9.5}$$

$$\boldsymbol{\tau}_m\cdot(\Delta_{4h}\mathbf{U}_{\mathbf{i}})=0, \tag{9.6}$$

$$D_{0,r_1}(\mathbf{a}_1\cdot\Delta_{2h}\mathbf{U}_{\mathbf{i}})=-D_{0,r_2}(\mathbf{a}_2\cdot\Delta_{2h}\mathbf{U}_{\mathbf{i}})-D_{0,r_3}(\mathbf{a}_3\cdot\Delta_{2h}\mathbf{U}_{\mathbf{i}}), \tag{9.7}$$

$$(D_{+,r_1}D_{-,r_1})^2(\boldsymbol{\tau}_m\cdot\mathbf{U}_{\mathbf{i}})=0. \tag{9.8}$$

Equations (9.5)–(9.8) will determine the solution at the two ghost lines, $\mathbf{U}_{-1,i_2,i_3}$ and $\mathbf{U}_{-2,i_2,i_3}$. Condition (9.8) is an approximation to the more accurate compatibility

condition $\Delta_{2h}^2(\boldsymbol{\tau}_m \cdot \mathbf{U}_\mathbf{i}) = 0$. Δ_{4h} is the fourth-order accurate FD4 approximation, and Δ_{2h} is the second-order accurate FD2 approximation to the Laplacian as defined in section 5. Equations (9.6)–(9.7) couple the unknown values at the ghost points with neighboring points. These equations can be solved by iterating on the unknown values using an appropriate initial guess; only one or two iterations are normally required.

9.2. Numerical dissipation. A high-order dissipation is added to the scheme to stabilize the approximation from weak, high-frequency (with respect to the grid) instabilities that may develop near interpolation points and boundaries. The most unstable modes tend to be the [illegible] that oscillate in sign from one grid point to the next. The appearance of an instability can be explained in the following way (see Reyna [35] for a careful analysis for the scalar wave equation on an overlapping grid). Except for simple cases of a single Cartesian grid or multiple matching Cartesian grids, the discrete operator for the method, including interpolation and boundary conditions, will not be exactly symmetric. As a result it can be expected that some of the eigenvalues of the discrete system will have a nonzero imaginary part. It follows from Lemma 5.1 that the corresponding eigenvectors will be unstable. Since the scheme is accurate, one can expect that the well-resolved modes will have accurate eigenvalues. The most unstable modes are likely associated with high-frequency modes that are not well resolved on the grid. The high-order dissipation is added to suppress these unstable modes. The dissipation is particularly effective for modes that vary rapidly in space. By default the dissipation is always added to the method.

At the continuous level, the dissipation of the second-order wave equation takes the form $u_{tt} = c^2\Delta u - \alpha\, h^{2p}(-\Delta)^p u_t$, where h is a measure of the local grid spacing. For the fourth-order accurate scheme a fourth-order dissipation with $p = 2$ is typically used,

$$U_\mathbf{i}^{n+1} - 2U_\mathbf{i}^n + U_\mathbf{i}^{n-1} = (c\Delta t)^2\Delta_{4h}U_\mathbf{i}^n + \tfrac{(c\Delta t)^4}{12}(\Delta^2)_{2h}U_\mathbf{i}^n - \alpha_d(c\Delta t)^2\sum_{m=1}^{d}(\Delta r_m)^4(D_{+r_m}D_{-r_m})^2(U_\mathbf{i}^n - U_\mathbf{i}^{n-1})/\Delta t. \tag{9.9}$$

The coefficient α_d is typically chosen to be about 1. The scheme remains fourth-order accurate provided that $\alpha_d = O(1)$. The damping acts as a low pass filter, having little effect on the waves that are well resolved on the grid but rapidly damping the modes that are not resolved on the grid. It is also possible to use a sixth-order dissipation with the fourth-order scheme, but this does not seem to provide any significant advantage.

9.3. Material interfaces. The boundary conditions for material interfaces have been implemented in two dimensions. Both second- and fourth-order accurate approximations have been developed; see sections 8 and 10.3 for some fourth-order accurate computations. The implementation is currently restricted to the case when the grids that meet at the interface have matching grid points (as shown, for example, in the grid in Figure 7). Currently only a simple iteration is used to solve the coupled interface equations. A more sophisticated solution scheme could be useful, but this option has not been investigated.

The solution of the fourth-order accurate approximations in two dimensions will be discussed here; the second-order accurate case is similar. Assume, as in section 7, that the interface corresponds to the line $i_1 = 0$, with the unknown ghost point values being $\mathbf{U}^{(k)}_{-m,i_2,i_3}$ for the ghost lines $m = 1, 2$ on the two sides of the interface, $k = 1, 2$. As a first step, the basic jump conditions (7.3)–(7.4) for the electric field are imposed.

These conditions are used to assign the values of the electric field $\mathbf{U}^{(k)}_{0,i_2,i_3}$ for the side of the interface with the larger value of ϵ. The interface compatibility conditions are given by (7.15)–(7.18). The solution to these equations will determine the values at the ghost points on both sides of the interface. Approximations for Δ_{4h}, ∇_{4h}, $(\nabla\cdot\Delta)_{2h}$, etc. are obtained with the mapping method (section 5). Three of the conditions (7.15), (7.17), (7.18) cause the equations to be coupled between adjacent points along the boundary, although on an orthogonal grid only the higher-order correction conditions (7.17), (7.18) lead to coupling. At the start of the iteration the values on the ghost points on both sides of the interface are given initial values by extrapolation. Then, at each step in the iteration, a small system of equations is solved at each point, $(0, i_2)$, along the boundary for the 8 unknown values $\mathbf{U}^{(k)}_{-m,i_2,i_3}$, $m = 1, 2$, $k = 1, 2$, using the 8 equations (7.15)–(7.18). There will be 12 unknowns and 12 equations in three dimensions. When solving these equations the values at adjacent points are kept fixed at their most recent values.

For the long-time computations presented at the end of section 8, which included no dissipation, it was found necessary to solve the interface equations very accurately (approximately 30 iterations) to avoid the development of late time instabilities. With the addition of a small fourth-order dissipation the number of iterations can be reduced to just a few. For the computations presented in section 10.3, which included a small fourth-order dissipation, it was found sufficient to use only one or two iterations to approximately solve the interface equations.

9.4. Parallel implementation. The algorithms described here have been implemented in the computer code **mx**. The **mx** solver is implemented at a high level in C++. It is based upon the Overture class library.[2] Most of the numerically intensive kernels are written in Fortran77. The overhead in using C++ is found to be negligible for any reasonable size problem (see section 10.5). The computer code **mx** runs on both serial and distributed memory parallel computers.

In a distributed parallel computing environment, the grid functions on each component grid (representing the solution variables such as E_x, E_y, E_z) are represented as multidimensional arrays. These arrays can be distributed across one or more processors. The grid functions are implemented using parallel distributed arrays from the P++ array class library [34]. Each P++ array can be independently distributed across the available processors. The distributed array consists of a set of serial arrays, one serial array for each processor. Each serial array is a multidimensional array that can be operated on using array operations. The data from the serial array can also be passed to a Fortran function, for example. When running in parallel, the serial arrays contain extra ghost lines that hold copies of the data from the serial arrays on neighboring processors. P++ is built on top of the Multiblock PARTI parallel communication library [40], which is used for ghost boundary updates and copying blocks of data between arrays with possibly different parallel distributions.

A special parallel overlapping grid interpolation routine has been developed for updating the points on grids that interpolate from other grids. Overlapping grid interpolation is based on a multidimensional tensor product Lagrange interpolant. In parallel, the Lagrange formula is evaluated on the processor that owns the data in the stencil (the donor points); the resulting sums are collected into a message and then sent to the processor that owns the interpolation points. There is at most one message that needs to be sent between any two processors. In this way, the number

[2] http://www.llnl.gov/CASC/Overture

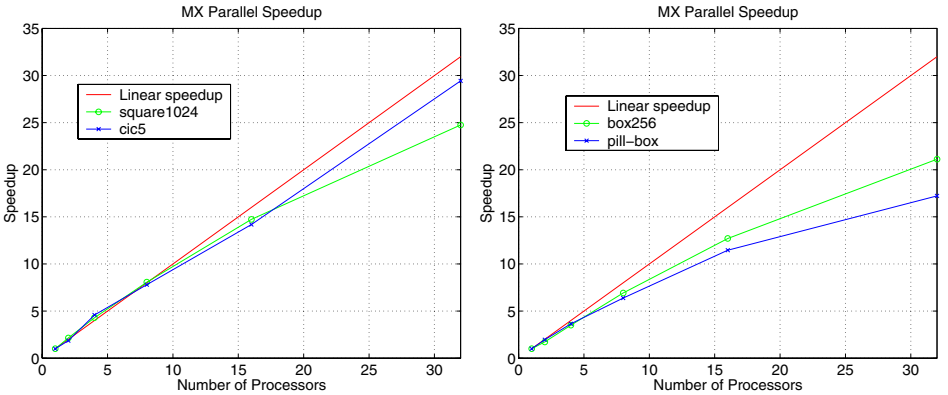


FIG. 5. *Strong scaling parallel results for the fourth-order accurate Maxwell solver. Left: speedup for a square with* 1024^2 *(*1.1*e*6*) points compared to a two-dimensional circle-in-a-channel grid with* 3.8*e*6 *grid points. Right: speedup for a box with* 256^3 *(*1.78*e*7*) grid points, compared to a pill-box grid with* 3.2*e*6 *grid points. Computations were performed on a Linux cluster.*

of messages and the size of the messages that need to be passed between processors are quite small.

Figure 5 shows some results from solving Maxwell's equations with the **mx** solver on a distributed memory Linux cluster (with 2.4 GHz Zeon processors). The equations are solved to fourth-order accuracy. In these examples the number of grid points remains fixed as the number of processors is increased. The figure compares two-dimensional results for a square with 1024^2 $(1.1e6)$ grid points to that of a circle-in-channel overlapping grid with $3.8e6$ grid points. The parallel scaling is very good in two dimensions for these problems since the number of computational points is large compared to the number of points where communication is required. Also shown are three-dimensional results for a box with 256^3 $(1.78e7)$ grid points compared to a grid for a pill-box with about $3.2e6$ grid points (the pill-box appears in Figure 10). In the three-dimensional cases, there are relatively many more points where communication is required since for N^3 grid points in three-dimensions there are $O(N^2)$ points on the interprocessor boundaries. The parallel scaling is still reasonable with the results for the overlapping grid being comparable to that for the box.

10. Numerical results. Numerical results will now be presented that demonstrate the accuracy and efficiency of the fourth-order accurate version of the scheme. The scheme has also been verified on a number of other problems including the scattering of a plane wave by a two-dimensional cylinder and the computation of eigenmodes of a three-dimensional cylindrical cavity.

10.1. The method of analytic solutions. The [illegible] is an extremely useful technique for constructing exact solutions to check the accuracy of a computer program. This method, also sometimes known as the [illegible] [36], or [illegible] [3], adds forcing functions to the governing equations and boundary conditions. These forcing functions are determined so that some given functions, $\mathbf{E}^{\text{true}}(\mathbf{x},t)$, will be the exact solution to the forced equations. With this approach, the error in the discrete solution can be easily determined. Two common choices for exact solutions are low-degree polynomials or trigonometric functions. The polynomial solutions are useful for checking that the

fourth-order scheme can exactly solve a problem on a Cartesian grid, or overlapping Cartesian grids, when the degree of the polynomials is less than or equal to four. The trigonometric functions will be used in the convergence results given in this section. The exact solutions are chosen to be divergence free. In two and three dimensions the functions are

$$(E_x^{\text{true}}, E_y^{\text{true}}, H_z^{\text{true}}) = \left(\tfrac{1}{2}c_x c_y, \tfrac{1}{2}s_x s_y, c_x s_y\right)\cos(\pi t),$$
$$(E_x^{\text{true}}, E_y^{\text{true}}, E_z^{\text{true}}) = \left(c_x c_y c_z, \tfrac{1}{2}s_x s_y c_z, \tfrac{1}{2}s_x c_y s_z\right)\cos(\pi t),$$

where $c_x = \cos(\pi x)$, $s_y = \sin(\pi x)$, etc. Define the error in the discrete approximation $U_{\mathbf{i}}^n$ to E_x at a given time t^n by

$$\mathrm{e}_{\mathbf{i}}^{E_x} = U_{\mathbf{i}}^n - E_x^{\text{true}}(\mathbf{x}_{\mathbf{i}}, t^n), \tag{10.1}$$

with similar definitions for $\mathrm{e}_{\mathbf{i}}^{E_y}$, $\mathrm{e}_{\mathbf{i}}^{E_z}$, and $\mathrm{e}_{\mathbf{i}}^{H_z}$. Define the discrete max-norm by

$$\|U_{\mathbf{i}}^n\|_\infty = \max_{\mathbf{i}} |U_{\mathbf{i}}^n|, \tag{10.2}$$

where the maximum is taken over all valid points in the overlapping grid.

Table 1 shows convergence results for a two-dimensional cylinder in a channel. The coarse grid, $\mathcal{G}_1$, for this case consists of a Cartesian grid for the square $[-2,2]^2$ with 61×61 grid points, and an annular grid centered at the origin with inner radius $\frac{1}{2}$, with outer radius $\frac{3}{4}$, and containing 81×5 grid points. The grids $\mathcal{G}_2$ and $\mathcal{G}_4$ are, respectively, two and four times finer in each direction. PEC boundary conditions are taken on all boundaries. The results in the table show that the solution is converging at a rate close to 4 in the maximum norm. The nondimensionalized divergence in $\mathbf{E}$, defined as $\delta_{\mathbf{E}} = \|\nabla \cdot \mathbf{E}\|_\infty / \|\nabla \mathbf{E}\|_\infty$, is small and also converging at a rate close to 4. The convergence rate σ was estimated by assuming an error of the form $E = Ch^\sigma$ and making a least squares fit for σ to the equation $\log(E) = \sigma \log(h) + \log(C)$.

TABLE 1

Convergence rates for a two-dimensional cylinder in a channel grid. The maximum errors in E_x, E_y, H_z and $\delta_{\mathbf{E}} = \|\nabla \cdot \mathbf{E}\|_\infty / \|\nabla \mathbf{E}\|_\infty$ are shown at $t = 1$ for a trigonometric exact solution. The estimated convergence rate, $E \propto h^\sigma$, is also indicated.

Grid	$\|\mathrm{e}^{E_x}\|_\infty$	$\|\mathrm{e}^{E_y}\|_\infty$	$\|\mathrm{e}^{H_z}\|_\infty$	$\delta_{\mathbf{E}}$
$\mathcal{G}_1$	$9.1e{-5}$	$6.4e{-5}$	$1.3e{-4}$	$1.7e{-4}$
$\mathcal{G}_2$	$4.2e{-6}$	$4.5e{-6}$	$8.1e{-6}$	$1.1e{-5}$
$\mathcal{G}_4$	$2.2e{-7}$	$2.9e{-7}$	$5.1e{-7}$	$7.8e{-7}$
Rate σ	4.40	3.93	4.02	3.94

Table 2 shows convergence results for solutions computed on a sphere in a box domain. The coarse grid, $\mathcal{G}_1$, in this case consists of three grids. A Cartesian grid covers the cube $[-2,2]^3$ with 41^3 grid points. The sphere of radius $\frac{1}{2}$ is covered with two overlapping orthographic patches, each with $33 \times 33 \times 13$ grid points. The grids $\mathcal{G}_2$ and $\mathcal{G}_4$ are, respectively, two and four times finer in each direction. The fine grid $\mathcal{G}_4$ has about 6.4 million grid points. PEC boundary conditions are taken on all boundaries. The results in the table show the maximum errors and estimated convergence rates. The convergence rates for the errors in $\mathbf{E}$ are close to 4. The convergence rate for $\delta_{\mathbf{E}}$ is approximately 3.5, which is still reasonable considering that this quantity is derived from derivatives of the computed variables.

TABLE 2

Convergence rates for a three-dimensional sphere in a box domain. The maximum errors in E_x, E_y, E_z and the nondimensionalized divergence $\delta_{\mathbf{E}}$ are shown at $t = 1$ for a trigonometric exact solution. The estimated convergence rate, $E \propto h^{\sigma}$, is also indicated.

Grid	$\|\mathrm{e}^{E_x}\|_\infty$	$\|\mathrm{e}^{E_y}\|_\infty$	$\|\mathrm{e}^{E_z}\|_\infty$	$\delta_{\mathbf{E}}$
$\mathcal{G}_1$	$1.1e{-3}$	$6.0e{-4}$	$1.4e{-3}$	$3.0e{-3}$
$\mathcal{G}_2$	$4.8e{-5}$	$1.9e{-5}$	$6.0e{-5}$	$2.1e{-4}$
$\mathcal{G}_4$	$3.7e{-6}$	$1.6e{-6}$	$4.6e{-6}$	$2.4e{-5}$
Rate σ	4.12	4.26	4.11	3.47

TABLE 3

The maximum errors at $t = 3$ for the computation of a plane wave scattering from a perfectly conducting sphere.

Grid	$\|\mathrm{e}^{E_x}\|_\infty$	$\|\mathrm{e}^{E_y}\|_\infty$	$\|\mathrm{e}^{E_z}\|_\infty$	$\delta_{\mathbf{E}}$
$\mathcal{G}_1$	$1.3e{-2}$	$8.1e{-3}$	$6.7e{-3}$	$3.9e{-3}$
$\mathcal{G}_2$	$9.3e{-4}$	$5.8e{-4}$	$4.8e{-4}$	$4.2e{-4}$
$\mathcal{G}_4$	$6.2e{-5}$	$3.9e{-5}$	$3.2e{-5}$	$5.4e{-5}$
Rate σ	3.86	3.86	3.85	3.09

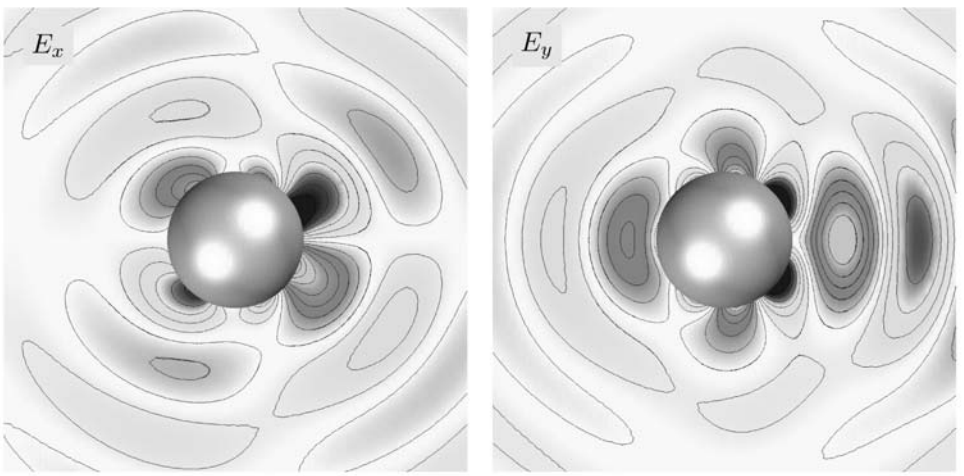


FIG. 6. *Scattering of a plane wave by a perfectly conducting sphere. The scattered fields for E_x and E_y are shown.*

10.2. Scattering by a PEC sphere. The scattering of an incident plane wave $\mathbf{E}^I = \left(e^{i(kx-\omega t)}, 0, 0\right)$ by a sphere of radius a is considered. The exact Mie series solution is well known [42]. The given incident field can be subtracted out, leaving the solution to the scattered field to be determined. A PEC boundary condition is imposed on the surface of the sphere, taking into account the incident field that has been removed. The initial conditions are taken as the exact solution for the scattered field at times $t = -\Delta t$ and $t = 0$. The exact solution is imposed on the outer boundaries of the domain. The solution is integrated to time $t = 3$. The grids, $\mathcal{G}_1$, $\mathcal{G}_2$, and $\mathcal{G}_4$, used for the sphere in a box are the same as those described in section 10.1. Table 3 presents the maximum errors and estimated convergence rates. The computed values of $\mathbf{E}$ are converging at rates close to 4. Figure 6 shows the scattered field for an incident wave with $ka = \pi$.

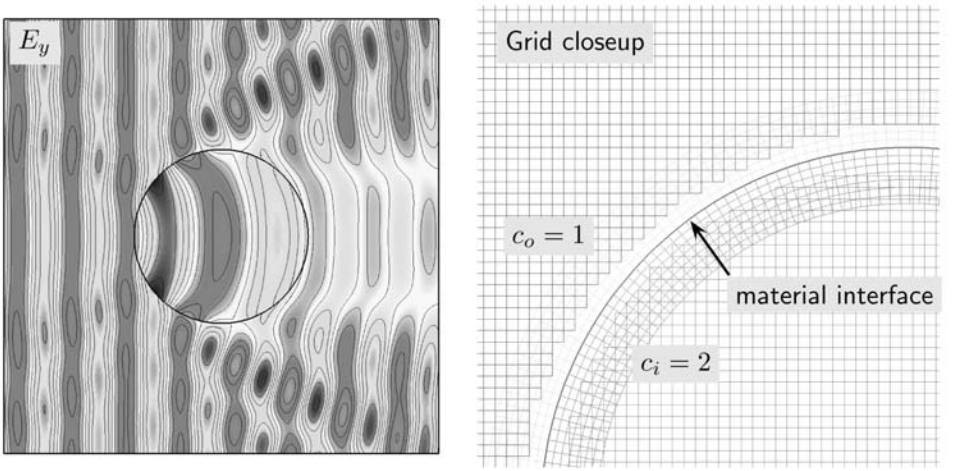


FIG. 7. *Scattering of a plane wave by a dielectric cylinder for* $ka = 4\pi$, $c_o = 1$, *and* $c_i = 2$. *The field component* E_y *is shown for the solution computed on grid* $\mathcal{G}_4$. *A closeup of the grid* $\mathcal{G}_2$ *near the interface is shown.*

10.3. Scattering of a plane wave by a dielectric cylinder. The two-dimensional computation of a plane wave hitting a dielectric (insulating) cylindrical disk possessing different material properties is used to test the accuracy of the material interface conditions presented in section 7. Consider a plane wave, $(E_x^I, E_y^I, H_z^I) = (0, \frac{\omega}{k}e^{i(kx-\omega t)}, e^{i(kx-\omega t)})$, incident upon a dielectric cylinder of radius a. Let ϵ_i be the electric permittivity for the region interior to the cylinder, let ϵ_o be the electric permittivity for the region outside the cylinder, and let $m = \sqrt{\epsilon_i/\epsilon_o} = c_o/c_i$ be the relative index of refraction for the dielectric. (The magnetic permeability μ is assumed to be the same in both materials.) The solution for the magnetic field in terms of the incident, scattered field and dielectric field is known as a series solution; see Balanis [2] or van de Hulst [42]. A closeup of the grid used in this computation is shown in Figure 7. The coarse grid, $\mathcal{G}_1$, for this domain consists of four component grids. A Cartesian grid covers the domain $[-1,1]^2$ with 81×81 grid points. An annular grid of inner radius $a = \frac{1}{2}$ and radial width .0875 covers the region immediately adjacent and exterior to the circle of radius $\frac{1}{2}$, centered at the origin. A second annular grid with the same radial width and grid spacing covers the region just inside the circle, and a second Cartesian grid lies in the central region. The refined versions of the grids are $\mathcal{G}_2$ and $\mathcal{G}_4$. The width of the annular grids are decreased by the same factor of 2 as the grids are refined so that 6 grid points remain in the radial direction. The initial conditions at $t = 0$ and $t = -\Delta t$ are taken as the exact solution. The boundary condition on the outer boundaries is also taken as the exact solution.

Table 4 shows convergence results for the scattering of a plane wave by the dielectric cylinder for an incident wave number of $ka = 4\pi$ with $c_o = 1$ and $c_i = 2$. The fourth-order accuracy of the method is evident. The E_y component of the computed solution is shown in Figure 7. Note that the tangential component of the field is continuous at the interface, but there is a jump in the normal component.

10.4. Computing eigenvalues of a disk and an L-shaped domain. The main focus of this paper has been on the solution of the time domain equations. In

TABLE 4
Maximum errors at $t = 1$ in computing the scattering of a plane wave by a dielectric cylinder. The incident wave number was $ka = 4\pi$, with $\epsilon = 1/4$ inside the cylinder and $\epsilon = 1$ in the region exterior to the cylinder.

Grid	$\|e^{E_x}\|_\infty$	$\|e^{E_y}\|_\infty$	$\|e^{H_z}\|_\infty$	$\delta_{\mathbf{E}}$
$\mathcal{G}_1$	$1.4e{-1}$	$2.9e{-1}$	$3.0e{-1}$	$6.7e{-2}$
$\mathcal{G}_2$	$1.0e{-2}$	$2.1e{-2}$	$2.2e{-2}$	$4.5e{-3}$
$\mathcal{G}_4$	$6.8e{-4}$	$1.4e{-3}$	$1.4e{-3}$	$2.9e{-4}$
Rate σ	3.86	3.87	3.88	3.92

this section the computation of the frequency domain eigenvalues and eigenvectors will be briefly considered. Of particular interest will be the behavior of the scheme in computing approximations to the highly singular solutions of a problem with a reentrant corner.

The eigenvalue problem for Maxwell's equations is posed as

$$-\Delta \mathbf{E} = \lambda \mathbf{E}, \qquad \mathbf{x} \in \Omega, \tag{10.3}$$

$$\mathbf{n} \times \mathbf{E} = 0, \quad \nabla \cdot \mathbf{E} = 0, \qquad \mathbf{x} \in \partial\Omega. \tag{10.4}$$

This formulation is sometimes called the "regularized" eigenvalue problem [18, 8] since the operator $\nabla \times \nabla$ has been replaced by the more [illegible] operator $-\Delta$. This problem is discretized following the approximations and boundary conditions given in section 9, resulting in the generalized matrix eigenvalue problem $Ax = \lambda Mx$. The smallest eigenvalues of this system are determined using ARPACK [31]. Note that the eigenvalue problem (10.3)–(10.4) admits solutions with nonzero $\nabla \cdot \mathbf{E}$ (corresponding to the Dirichlet eigenvalues of the $-\Delta$ operator). These eigenvalues are removed as a postprocessing step. Alternatively the original formulation (10.3)–(10.4) can be altered to eliminate these eigenvalues [17].

As a first example, consider the computation of the eigenvalues for the domain consisting of the unit disk, $\Omega = \{\mathbf{x} \in \mathbb{R}^2 \mid |\mathbf{x}| \le 1\}$. The exact eigenvalues and eigenvectors for this problem are easily determined. The coarse grid, $\mathcal{G}_1$, for this case consists of a Cartesian grid for the square $[-.7, .7]^2$ with 25×25 grid points, and an annular grid centered at the origin with inner radius $\frac{3}{4}$ and outer radius 1 and containing 61×7 grid points. Grids $\mathcal{G}_2$ and $\mathcal{G}_4$ have two and four times as many points in each direction. Figure 8 shows the coarse grid $\mathcal{G}_1$, the E_y component of the eigenvector for the smallest eigenvalue, and the relative errors in the first five eigenvalues for the three different grids. The relative error is defined as the difference between the computed value and the true value, divided by the true value. It is seen that the eigenvalues are computed very accurately with the values converging at a rate close to fourth-order.

As a second problem, the eigenvalues will be computed for the L-shaped domain $\Omega = [-1, 1]^2 - [-1, 0]^2$. This problem is quite difficult due to the highly singular nature of some of the eigenfunctions. The eigenfunction corresponding to the smallest eigenvalue, for example, becomes unbounded at the corner. It is known [8] that many straightforward finite-difference and finite element approximations to the eigenvalue problem for a problem with a reentrant corner will fail to approximate some of the singular eigenfunctions (those that become unbounded), and the corresponding eigenvalues will be missing entirely from the discretized equations. There are a number of proposed fixes to this problem including adding new degrees of freedom based on singular functions [8] or the use of a weighted regularization [5]. Note that both of

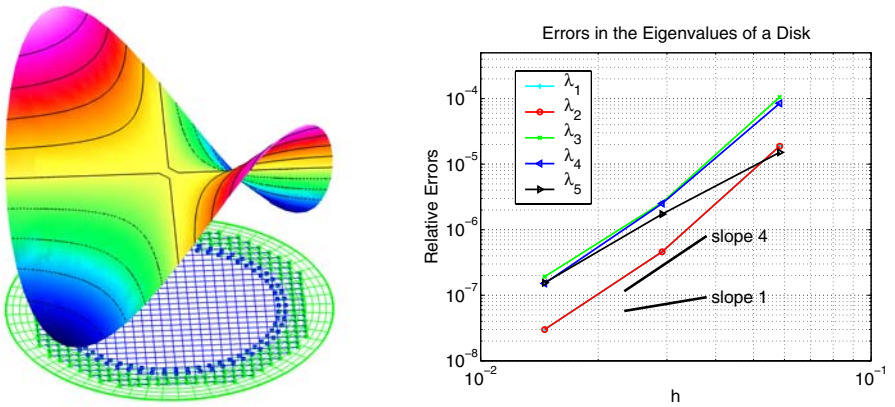


FIG. 8. *Solving the eigenvalue problem for the unit disk. Left: the coarse grid and the component E_y of the eigenvector for the smallest eigenvalue. Right: the relative errors in the smallest five eigenvalues for three different grid spacings.*

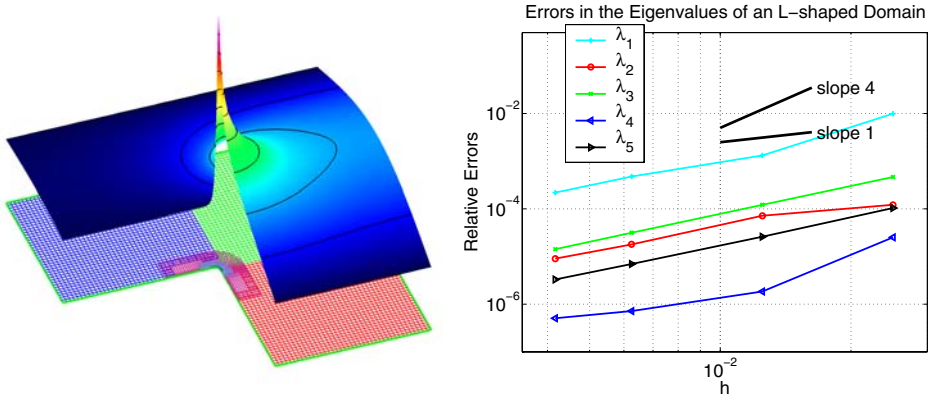


FIG. 9. *Solving the eigenvalue problem for an L-shaped domain. Left: the coarse grid $\mathcal{G}_1$ and the component E_x of the eigenvector for the smallest eigenvalue. The reentrant corner is rounded off slightly and refined. Right: the relative errors in the smallest five eigenvalues for three different grid spacings.*

these latter approaches could likely be adapted to the current discretization if so desired. The use of edge elements [33] and a sufficiently fine grid will not have these difficulties.

For the case here using overlapping grids, the corner problem will be dealt with by introducing a domain with a slightly rounded corner as shown in Figure 9. The radius of curvature of the rounded corner can be made as small as desired, given a sufficiently fine grid. This use of smooth boundary approximations to corners with large opening angles is one that the author generally recommends for other classes of problems as well. The coarse grid, $\mathcal{G}_1$, for the L-shaped domain consists of a total of six grids. There are three Cartesian grids for the three square domains $[-1,0]\times[0,1]$, $[0,1]^2$, and $[0,1]\times[-1,0]$, each with 41×41 points. A small boundary fitted grid with 64×14 points is added to define a smooth rounded corner. Two additional nested refinement grids with 80×17 and 100×21 are added to this boundary fitted grid (see

Figure 9). The boundary fitted grid is defined by an analytic function involving the logarithm of hyperbolic cosine functions [19]. Finer grids $\mathcal{G}_2$, $\mathcal{G}_4$, and $\mathcal{G}_6$ have two, four, and six times as many points in each direction.

Figure 9 shows the relative errors in computing the first 5 of the smallest eigenvalues for the L-shaped domain. The coarse grid $\mathcal{G}_1$ is shown along with the E_x component of the eigenvector corresponding to the smallest eigenvalue. The eigenvalues for the L-shaped domain correspond to the eigenvalues of the Neumann problem for the $-\Delta$ operator and have been computed very accurately[3] (see Table 5). The relative errors shown in the figure and table are for the computed eigenvalues of the rounded domain compared to these accurate values for the domain with a sharp corner. The results from Figure 9 indicate that all of the smallest eigenvalues can be computed with reasonable accuracy. The convergence rates are between first- and second-order accurate rather than fourth-order. This might be expected since the solutions and gradients are very large and not fully resolved. The eigenvalues of the rounded domain will also be slightly different from the domain with a sharp corner. It may be possible to improve the convergence rates by a more careful choice of grid or by incorporating knowledge of the singularity, following the approaches mentioned earlier.

The numerical values of the computed eigenvalues for the rounded corner are compared to the "exact" values for a sharp corner in Table 5. Note that the imaginary parts of these computed eigenvalues are all found to be zero (to double precision accuracy). The matrix system $Ax = \lambda Mx$ for the generalized eigenvalue problem will not in general be completely symmetric on an overlapping grid, and thus there may be some eigenvalues with an imaginary part. It is expected that some high-frequency eigenvalues will have an imaginary part; these will correspond to the weakly unstable modes for the time-dependent problem. A high-order dissipation is added to the time domain algorithm to remove this instability (discussed in section 9.2).

TABLE 5

Real and imaginary parts, $(\mathrm{Re}(\lambda), \Im(\lambda))$*, of the computed eigenvalues for the L-shaped domain with a rounded corner compared, the "exact" values for a sharp corner, and the relative errors. Results are for the coarse grid* $\mathcal{G}_1$*.*

	Computed	"Exact"	Rel-err
λ_1	$(1.4903, 0.0)$	1.47562182	9.93e−03
λ_2	$(3.5345, 0.0)$	3.53403137	1.22e−04
λ_3	$(9.8650, 0.0)$	9.86960440	4.67e−04
λ_4	$(9.8699, 0.0)$	9.86960440	2.53e−05
λ_5	$(11.391, 0.0)$	11.3894794	1.04e−04

10.5. Serial performance and memory usage. The performance of the **mx** solver on two problems, one two-dimensional and one three-dimensional, is given in Table 6. The two-dimensional problem is for a circle in a channel grid. The three-dimensional problem considered the computation in the pill-box geometry shown in Figure 10. The table indicates the amount of time that is spent in various parts of the code. In the two-dimensional case, close to 70% of the time is spent in the Fortran kernel that advances the solution on the interior points of the Cartesian grid. Advancing the solution on the curvilinear grids takes about 14%, while applying the boundary conditions on all grids takes about 11%. These results show that the overall

[3] www.maths.univ-rennes1.fr/∼dauge/benchmax.html

TABLE 6

*CPU time (in seconds) per step for various parts of the code and their percentage of the total CPU time per step. Results for the two-dimensional cylinder in a channel problem (*3.8 *million grid points) are given in the two columns on the left. Results for the three-dimensional pill-box problem (*10.2 *million grid points) are given in the two columns on the right. Computations were performed on a desktop with a* 2.2 *GHz Zeon processor.*

	2D cylinder		3D pill-box	
	s/step	%	s/step	%
Advance Cartesian grids	1.24	71.4	4.23	28.2
Advance curvilinear grids	.242	14.0	7.18	47.8
Boundary conditions	.184	10.6	2.23	14.8
Interpolation	.008	0.46	1.01	6.8
Other	.056	3.4	0.35	2.4
Total	1.73	100	15.0	100

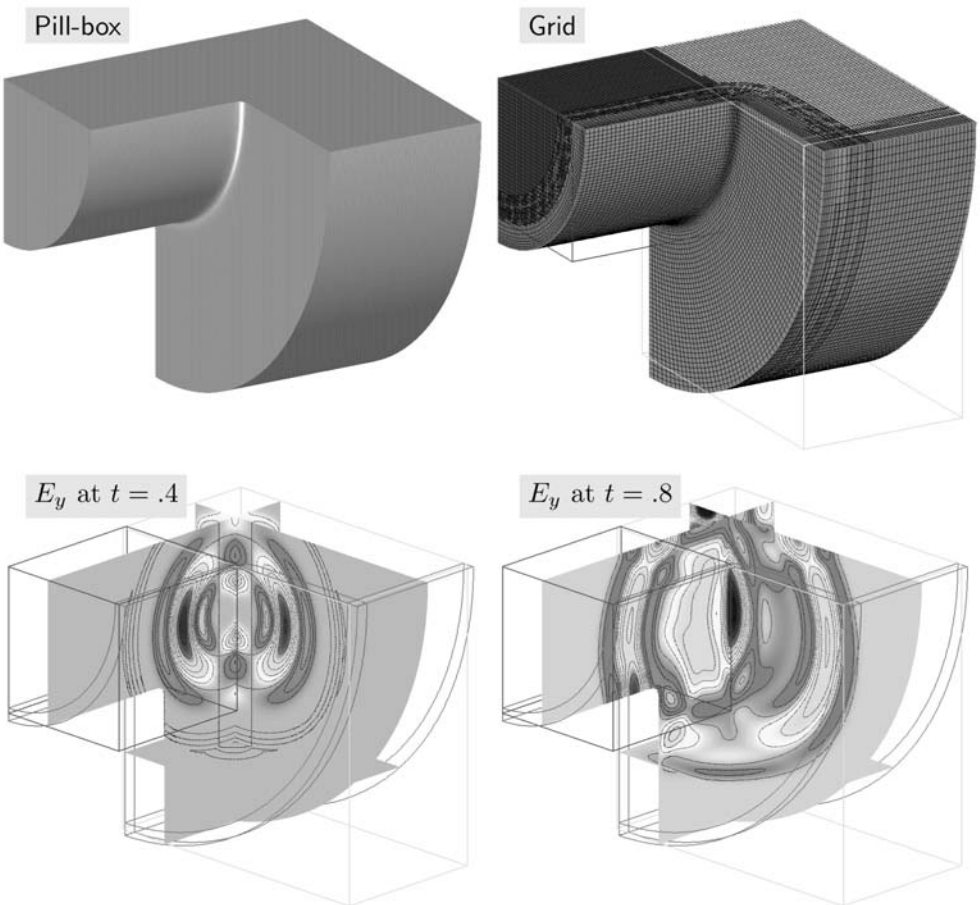


FIG. 10. *Propagation of a pulse in a pill-box cavity. Top left: the pill-box geometry. Top right: the overlapping grid for the geometry (coarse grid version). Bottom: the electric field component* E_y *at two times. Contours of the field are plotted on planes that cut through the computational domain. The block-boundaries of the grids are shown for reference.*

performance of the code is approaching that of a purely Cartesian grid solver, provided that a sufficiently high percentage of the grid points are Cartesian grid points.

In three dimensions, the advancement of the curvilinear grids takes the most time at about 48% with the Cartesian grids taking 28% and boundary conditions and interpolation taking a total of about 22%. In three dimensions the cost of advancing the curvilinear grid with a fourth-order approximation is on the order of 10 times slower per grid point than for a Cartesian grid. As the grids are refined the percentage of Cartesian grid points will increase since the number of Cartesian grid points scales like N^3, while the number of curvilinear grid points grows like N^2 (assuming that the curvilinear boundary fitted grids retain a fixed number of grid points in the normal direction as the grids are refined). Thus for sufficiently fine grids in three dimensions it can be expected that the time will be dominated by the work done on Cartesian grids.

Table 7 gives the computer memory usage of the **mx** code for solving Maxwell's equations, to fourth-order accuracy, on the three-dimensional pill-box example of Figure 10. The table indicates the amount of memory required in MBytes for a double precision (64-bit) computation and the number of double precision floating point numbers required per grid point, denoted by reals/point. The number of reals/point for this computation was about 12.6. Only the **E** field was computed in this case. About 18% of the memory was dedicated to the storage of the overlapping grid (grid metrics, masks, etc.). To put these numbers in context, note that if the Cartesian grids were treated as general curvilinear grids, then the storage of the Jacobian matrix $\partial\mathbf{x}/\partial\mathbf{r}$ on the entire grid would itself require 9 reals/point. The **mx** solver is therefore seen to be quite memory efficient.

TABLE 7

Memory usage for the three-dimensional pill-box problem solved in double precision with the fourth-order accurate approximation. The grid for this problem contained about 10.2 *million grid points with about* 4.2e5 *interpolation points. The number of double precision floating point numbers required per grid point was about* 12.6 *(reals/pt). About* 17.8% *of the memory was dedicated to the storage of the overlapping grid (grid metrics, masks, etc.).*

Memory usage, pill-box problem			
	Mbytes	Reals/pt	%
Overlapping grid	175.	2.3	17.8
Grid functions	779.	10.0	79.4
Other	26.	0.3	2.8
Total	980.	12.6	100.0

11. Conclusions. A scheme has been presented for the solution of the time-dependent Maxwell equations in complex geometries. The approach uses overlapping grids with a typical grid consisting of one or more background Cartesian grids overlapping body fitted curvilinear grids. Maxwell's equations are solved as a second-order vector wave equation. A three-level time-stepping scheme is used that can generate methods of orders 2, 4, 6, and higher. High-order accurate symmetric finite-difference approximations to the generalized Laplace operator have been developed. These approximations are symmetric on arbitrary logically rectangular curvilinear grids. Implementations of these operators in two and three dimensions for orders 2, 4, 6, and 8 are shown to give energy-preserving results when solving Maxwell's equations on a square and a box with randomly perturbed grid points. High-order accurate centered boundary conditions for PEC boundaries have been devised. High-order accurate centered boundary conditions have also been developed for material interfaces that

separate regions with constant values of the material properties. The interfaces are represented with overlapping grids, using boundary fitted grids on each side of the interface. These centered boundary conditions are based on compatibility conditions derived from the governing equations. The stability and accuracy of the fourth-order accurate PEC approximations are analyzed for a model problem on a periodic strip. The stability of the material interface conditions are analyzed for a one-dimensional model problem for a second- and a fourth-order approximation.

The approach has been implemented for computations on distributed memory multiprocessor machines. The arrays holding the solution variables and grid metrics on each component grid are distributed across one or more processors. An efficient parallel interpolation scheme has been developed for communicating the interpolation information where component grids overlap. Results from some parallel computations show that the algorithm scales well.

Numerical results for a number of two- and three-dimensional problems are used to verify the accuracy of the fourth-order accurate version of the method. The method of analytic solutions is used to generate [illegible] solutions to the forced equations and verify the convergence rates. The scattering of a plane wave by a three-dimensional sphere is computed and shown to converge to fourth-order accuracy compared to the exact series solution. The material interface boundary conditions are verified to be fourth-order accurate for the scattering of a plane wave by a dielectric cylinder. The frequency domain eigenvalue problem is discretized using the fourth-order accurate approximations. Eigenmodes and eigenvalues of a two-dimensional disk and an L-shaped domain are determined. By rounding the sharp reentrant corner of the L-shaped domain, the method is able to compute accurate approximations to the eigenvalues of the highly singular eigenmodes. Statistics presented from some representative computations show that the method is very efficient in both speed and memory, especially when the overlapping grid for the domain consists mainly of Cartesian grid points. In that case the performance of the method can approach that of a Cartesian grid solver.

The **mx** Maxwell solver has been implemented using the Overture framework that has support for adaptive mesh refinement, moving grids and fast multigrid solvers for elliptic equations [22, 21]. These capabilities of Overture may be useful for future extensions of the Maxwell solver. Far-field radiation conditions for boundaries of the computational domain that have been artificially truncated are an important consideration. The treatment of these far-field boundary conditions in conjunction with the present approach is left to a future publication. The general approach developed here can also be applied to other systems that take the form of a second-order wave equation, such as the equations of elasticity.

Acknowledgments. The author would like to thank Professor Heinz-Otto Kreiss for many useful discussions. Thanks also to Kyle Chand for help with the **mx** solver.

REFERENCES

[1] L. Anné, P. Joly, and Q. H. Tran, *Construction and analysis of higher order finite difference schemes for the 1D wave equation*, Comput. Geosci., 4 (2000), pp. 207–249.
[2] C. A. Balanis, *Advanced Engineering Electromagnetics*, John Wiley & Sons, New York, 1989.
[3] G. Chesshire and W. Henshaw, *Composite overlapping meshes for the solution of partial differential equations*, J. Comput. Phys., 90 (1990), pp. 1–64.
[4] G. C. Cohen, *Higher-Order Numerical Methods for Transient Wave Equations*, Springer, New York, 2002.
[5] M. Costable and M. Dauge, *Weighted regularization of Maxwell equations in polyhedral domains*, Numer. Math., 93 (2002), pp. 239–277.

[6] M. Dablain, *The application of high order differencing for the scalar wave equation*, Geophysics, 51 (1986), pp. 54–66.
[7] S. Dey and R. Mittra, *A locally conformal finite-difference time-domain (FDTD) algorithm for modeling three-dimensional perfectly conducting objects*, IEEE Microwave Guided Wave Lett., 7 (1997), pp. 273–275.
[8] A.-S. B.-B. Dhia, C. Hazard, and S. Lohrengel, *A singular field method for the solution of Maxwell's equations in polyhedral domains*, SIAM J. Appl. Math., 59 (1999), pp. 2028–2044.
[9] A. Ditkowksi, K. Dridi, and J. Hesthaven, *Convergent Cartesian grid methods for Maxwell's equations in complex geometries*, J. Comput. Phys., 170 (2001), pp. 39–80.
[10] T. A. Driscoll and B. Fornberg, *A block pseudospectral method for Maxwell's equations.* I. *One-dimensional case*, J. Comput. Phys., 140 (1998), pp. 47–65.
[11] T. A. Driscoll and B. Fornberg, *Block pseudospectral methods for Maxwell's equations* II: *Two-dimensional, discontinuous-coefficient case*, SIAM J. Sci. Comput., 21 (1999), pp. 1146–1167.
[12] B. Fornberg, *A Practical Guide to Pseudospectral Methods*, Cambridge University Press, Cambridge, UK, 1996.
[13] S. D. Gedney, F. S. Lansing, and D. L. Rascoe, *Full wave analysis of microwave monolithic circuit devices using a generalized Yee-algorithm based on an unstructured grid*, IEEE Trans. Microwave Theory Tech., 44 (1996), pp. 1393–1400.
[14] B. Gustafsson, H.-O. Kreiss, and J. Oliger, *Time Dependent Problems and Difference Methods*, John Wiley & Sons, New York, 1995.
[15] B. Gustafsson and E. Mossberg, *Time compact high order difference methods for wave propagation*, SIAM J. Sci. Comput., 26 (2004), pp. 259–271.
[16] B. Gustafsson and P. Walhund, *Time compact difference methods for wave propagation in discontinuous media*, SIAM J. Sci. Comput., 26 (2004), pp. 272–293.
[17] C. Hazard, *Numerical simulation of corner singularities: A paradox in Maxwell-like problems*, C. R. Mecanique, 330 (2002), pp. 57–68.
[18] C. Hazard and S. Lohrengel, *A singular field method for Maxwell's equations: Numerical aspects for* 2*D magnetostatics*, SIAM J. Numer. Anal., 40 (2002), pp. 1021–1040.
[19] W. Henshaw, *Mappings for Overture, a Description of the Mapping Class and Documentation for Many Useful Mappings*, Research report UCRL-MA-132239, Lawrence Livermore National Laboratory, Livermore, CA, 1998.
[20] W. Henshaw, H.-O. Kreiss, and L. Reyna, *A fourth-order accurate difference approximation for the incompressible Navier-Stokes equations*, Comput. & Fluids, 23 (1994), pp. 575–593.
[21] W. D. Henshaw, *On multigrid for overlapping grids*, SIAM J. Sci. Comput., 26 (2005), pp. 1547–1572.
[22] W. D. Henshaw and D. W. Schwendeman, *An adaptive numerical scheme for high-speed reactive flow on overlapping grids*, J. Comput. Phys., 191 (2003), pp. 420–447.
[23] J. Hesthaven and T. Warburton, *Nodal high-order methods on unstructured grids,* I. *Time-domain solution of Maxwell's equations*, J. Comput. Phys., 181 (2002), pp. 186–221.
[24] R. Holland, *Finite difference solution of Maxwell's equations in generalized nonorthogonal coordinates*, IEEE Trans. Nuclear Sci., 30 (1983), pp. 4589–4591.
[25] J. Jin, *The Finite Element Method in Electromagnetics*, John Wiley & Sons, New York, 1993.
[26] H. M. Jurgens and D. W. Zingg, *Numerical solution of the time-domain Maxwell equations using high-accuracy finite-difference methods*, SIAM J. Sci. Comput., 22 (2000), pp. 1675–1696.
[27] H.-O. Kreiss and N. A. Petersson, *A second order accurate embedded boundary method for the wave equation with Dirichlet data*, SIAM J. Sci. Comput., 27 (2006), pp. 1141–1167.
[28] H.-O. Kreiss, N. A. Petersson, and J. Ystrom, *Difference approximations for the second order wave equation*, SIAM J. Numer. Anal., 40 (2002), pp. 1940–1967.
[29] J. Lambert, *Computational Methods in Ordinary Differential Equations*, Cambridge University Press, New York, 1973.
[30] J. Lee, R. Palendech, and R. Mittra, *Modeling three-dimensional discontinuities in waveguides using nonorthogonal FDTD algorithm*, IEEE Trans. Microwave Theory Tech., 40 (1992), pp. 346–352.
[31] R. B. Lehoucq, D. C. Sorensen, and C. Yang, *ARPACK Users' Guide: Solution of Large-Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods*, SIAM, Philadelphia, 1998.
[32] N. K. Madsen, *Divergence preserving discrete surface integral method for Maxwell's curl equations using non-orthogonal unstructured grids*, J. Comput. Phys., 119 (1995), pp. 34–45.
[33] J. Nédélec, *Mixed finite elements in* R^3, Numer. Math., 35 (1980), pp. 315–341.

[34] D. Quinlan, A++/P++ *Class Libraries*, Research report LA-UR-95-3273, Los Alamos National Laboratory, Los Alamos, NM, 1995.
[35] L. G. M. Reyna, *Part* III. *On Composite Meshes*, Ph.D. thesis, Department of Applied Mathematics, California Institute of Technology, Pasadena, CA, 1982.
[36] P. Roache, *Code verification by the method of manufactured solutions*, ASME J. Fluids Eng., 124 (2002), pp. 4–10.
[37] G. Rodrigue and D. White, *A vector finite element time-domain method for solving Maxwell's equations on unstructured hexahedral grids*, SIAM J. Sci. Comput., 23 (2001), pp. 683–706.
[38] T. Rylander and A. Bondeson, *Application of stable FEM-FDTD hybrid to scattering problems*, IEEE Trans. Antennas and Propagation, 50 (2002), pp. 141–144.
[39] G. R. Shubin and J. B. Bell, *A modified equation approach to constructing fourth order methods for acoustic wave propagation*, SIAM J. Sci. Statist. Comput., 8 (1987), pp. 135–151.
[40] A. Sussman, G. Agrawal, and J. Saltz, *A Manual for the Multiblock PARTI Runtime Primitives, Revision* 4.1, Technical report CS-TR-3070.1, Department of Computer Science, University of Maryland, College Park, MD, 1993.
[41] A. Taflove and S. C. Hagness, *Computational Electrodynamics: The Finite-Difference Time-Domain Method*, Artech House, Boston, 2000.
[42] H. van de Hulst, *Light Scattering by Small Particles*, Dover, New York, 1957.
[43] B. Yang, D. Gottlieb, and J. Hesthaven, *Spectral simulations of electromagnetic wave scattering*, J. Comput. Phys., 134 (1997), pp. 216–230.
[44] K. Yee, *Numerical solution of initial boundary value problems involving Maxwell's equations in isotropic media*, IEEE Trans. Antennas and Propagation, 14 (1966), pp. 302–307.
[45] K. S. Yee, J. S. Chen, and A. H. Chang, *Conformal finite-difference time-domain (FDTD) with overlapping grids*, IEEE Trans. Antennas and Propagation, 40 (1992), pp. 1068–1075.

SIAM J. SCI. COMPUT.
Vol. 28, No. 5, pp. 1766–1797
© 2006 Society for Industrial and Applied Mathematics

AN UNSTAGGERED, HIGH-RESOLUTION CONSTRAINED TRANSPORT METHOD FOR MAGNETOHYDRODYNAMIC FLOWS*

JAMES A. ROSSMANITH†

Abstract. The ideal magnetohydrodynamic (MHD) equations are important in modeling phenomena in a wide range of applications, including space weather, solar physics, laboratory plasmas, and astrophysical fluid flows. Numerical methods for the MHD equations must confront the challenge of producing approximate solutions that remain accurate near shock waves and that satisfy a divergence-free constraint on the magnetic field. Failure to accomplish this often leads to unphysical solutions. In this paper, a high-resolution wave propagation method is developed that utilizes a novel constrained transport technique to keep the magnetic field divergence-free. This approach is based on directly solving the magnetic potential equation in conjunction with a new limiting strategy to obtain a nonoscillatory magnetic field. It is demonstrated in this work that an unstaggered definition of the divergence is the correct one to use in the case of wave propagation methods. Therefore, we solve the magnetic potential equation on the same grid as the MHD equations; hence the usual grid staggering that is found in constrained transport methods is eliminated. We demonstrate through truncation error analysis and direct numerical simulation that the resulting method is second order accurate in space and time for smooth solutions and nonoscillatory near shocks and other discontinuities. The resulting numerical method has been implemented as an extension to the CLAWPACK software package and can be freely downloaded from the Web.

Key words. high-resolution methods, constrained transport, magnetohydrodynamics, hyperbolic conservation laws, wave propagation methods

AMS subject classifications. 65M99, 76W05, 35L45

DOI. 10.1137/050627022

1. Introduction.

The ideal magnetohydrodynamic (MHD) equations model the dynamics of an electrically conducting fluid. In conservation form, the MHD system can be written as

$$\frac{\partial}{\partial t}\begin{bmatrix}\rho\\ \rho\vec{u}\\ \mathcal{E}\\ \vec{B}\end{bmatrix} + \vec{\nabla}\cdot\begin{bmatrix}\rho\vec{u}\\ \rho\vec{u}\otimes\vec{u} + \left(p+\frac{1}{2}|\vec{B}|^2\right)\mathcal{I} - \vec{B}\otimes\vec{B}\\ \vec{u}\left(\mathcal{E}+p+\frac{1}{2}|\vec{B}|^2\right) - \vec{B}(\vec{u}\cdot\vec{B})\\ \vec{u}\otimes\vec{B} - \vec{B}\otimes\vec{u}\end{bmatrix} = 0, \tag{1.1}$$

$$\vec{\nabla}\cdot\vec{B} = 0\,, \tag{1.2}$$

where

$$p = (\gamma-1)\left(\mathcal{E} - \frac{1}{2}\rho|\vec{u}|^2 - \frac{1}{2}|\vec{B}|^2\right). \tag{1.3}$$

Here ρ is the mass density, $\vec{u} = (u_1, u_2, u_3)^t$ is the velocity field, $\mathcal{E}$ is the total energy, $\vec{B} = (B_1, B_2, B_3)^t$ is the magnetic field, p is the thermal pressure, $(1/2)|\vec{B}|^2$ is the magnetic pressure, and γ is the ideal gas constant. Equation (1.3) is the [illegible] for an ideal gas and closes the system by relating the pressure to the other

*Received by the editors March 17, 2005; accepted for publication (in revised form) April 28, 2006; published electronically October 20, 2006. This work was supported in part by NSF grants DMS-9803442 and DMS-0409972.

http://www.siam.org/journals/sisc/28-5/62702.html

†Department of Mathematics, University of Wisconsin, 480 Lincoln Drive, Madison, WI 53706-1388 (rossmani@math.wisc.edu).

unknowns. See, for example, Chapter 4 of Gombosi [19] for a detailed derivation of these equations.

The ideal MHD equations form a system of hyperbolic conservation laws, (1.1), with a constraint, (1.2). Furthermore, any [illegible] solution to (1.1) automatically satisfies (1.2) provided that the initial magnetic field obeys the divergence-free constraint (see, for example, [24]). Most numerical discretizations of (1.1), however, can guarantee only that the divergence of the magnetic field is of the order of the numerical truncation error:

$$\vec{\nabla} \cdot \vec{B} = \mathcal{O}(\Delta x^p, \Delta y^p). \tag{1.4}$$

Furthermore, across shock waves the discrete divergence may become very large:

$$\vec{\nabla} \cdot \vec{B} = \mathcal{O}(\Delta x^{-1}, \Delta y^{-1})\,. \tag{1.5}$$

Brackbill and Barnes [8] showed that this error can lead to spurious solutions. It has been shown several times in the literature that for shock-capturing schemes, failure to satisfy the divergence-free constraint can lead to unphysical oscillations near shocks and other discontinuities. These oscillations can in turn lead to the production of negative pressures. See Tóth [38] for a review.

In order to overcome the difficulty associated with the divergence error, several methods have been introduced. The first approach used was the projection method of Brackbill and Barnes [8], which makes use of a Poisson solver in each time step in order to project a predicted magnetic field down to its divergence-free subspace. For example, this approach was recently used by Balbas, Tadmor, and Wu [2] in conjunction with various central schemes to evolve the fluid variables. Zachary, Malgolin, and Collela [43] proposed a method that splits the MHD equations into a conservative and an advective part; the advective part is moved to the right-hand side and treated as a source term. In Powell's method [34, 35] a source term proportional to the divergence of the magnetic field is added, which causes any divergence error to propagate with the fluid velocity. This source term was first used by Godunov [18] to symmetrize the MHD system. Dedner et al. [11] introduced an additional hyperbolic equation with damping in order to control the divergence error. Yee and Sjögreen [41] introduced a high-order method with special divergence filtering. Fey and Torrilhon [14] and Torrilhon [40] developed a multidimensional limiting procedure that selects which discrete divergence is satisfied in each grid cell. Aslan [1] introduced a fluctuation splitting scheme for MHD on unstructured grids; the divergence-free condition is handled through the same source term as in Powell's approach. In this work we focus on the [illegible] (CT) framework of Evans and Hawley [13], which is based on the method of Yee [42] for Maxwell's equations. In this approach the numerical discretization is carefully chosen so that a divergence-free constraint is automatically satisfied to machine precision. The overall method is based on modifying a cell-centered finite volume method (called the [illegible]) in such a way that it satisfies a divergence-free constraint. The base scheme we will use is the Riemann solver–based wave propagation method [27].

The CT method of Evans and Hawley [13] provides a general framework for solving the ideal MHD equations so that the magnetic field satisfies a discrete divergence-free condition. The basic idea of their approach is to introduce a staggered magnetic field at cell faces (edges in two dimensions) and a staggered electric field at cell edges (corners in two dimensions) in such a way that the overall update for the magnetic field conserves a discrete definition of the divergence. The framework has built into it some

degrees of freedom that can be exploited in order to devise a specific numerical update. Several such methods have been proposed in the literature [5, 6, 10, 32, 33, 36, 12]. Furthermore, unstaggered variants of some of these methods were introduced by Tóth [38].

In this work we present a new CT scheme that updates the magnetic potential in a fundamentally different way than that of existing methods. In line with the methods of Balsara [5] and Londrillo and Del Zanna [33], the method proposed here tries to achieve an update for the magnetic field that preserves not only a divergence-free property, but also a quantifiable measure of the total variation. The basic features of the proposed method are summarized below:

1. The evolution equation for the magnetic potential is written as a system of first order hyperbolic equations. This system reduces to a scalar advection equation in two space dimensions.
2. The magnetic potential is updated with a high-resolution upwind method modified from the method of LeVeque [26].
3. Limiters are applied in such a way as to minimize the oscillations in both the magnetic potential and the magnetic field. Mathematical justification is provided for the choice of this limiting procedure.
4. The method is completely unstaggered; this makes the method simpler and reduces the amount of averaging in the numerical scheme. Analysis of the Lorentz force is provided that gives mathematical justification for the choice of the discrete divergence that is conserved.

2. The 8-wave Riemann solver. The numerical methods considered in this paper are based on the solution of one-dimensional (1D) [illegible]. In particular, we make use of [illegible], Roe-type Riemann solvers [30], which require us to compute wave decompositions. The specific Riemann solvers we consider in this paper are based on the 8-wave Riemann solver of Powell [34, 35]. Note that we are only using the wave decomposition of Powell and are not including the source terms of [34, 35] in our evolution equations.

For smooth solutions, the MHD equations can be rewritten in the following quasi-linear form:

$$\frac{\partial U}{\partial t} + A_1 \frac{\partial U}{\partial x} + A_2 \frac{\partial U}{\partial y} = 0\,,$$

where $U = (\rho, u_1, u_2, u_3, p, B_1, B_2, B_3)$ is the vector of primitive variables. The matrices A_1 and A_2 are 8×8 diagonalizable matrices with real eigenvalues:

$$A_m = R_m \Lambda_m L_m\,,$$

where $m = 1, 2$. R_m is the matrix of right eigenvectors, Λ_m is the diagonal matrix of eigenvalues, and $L_m = R_m^{-1}$ is the matrix of left eigenvectors. The exact form of A_1 and A_2 as well as their right and left eigenvectors can be found in [35]. The matrices A_m are related to the flux Jacobians, $\partial f^m/\partial q$, through a similarity transform:

$$\frac{df^m}{dq} = S A_m S^{-1},$$

where $S = \frac{dq}{du}$ [35].

The eigenvalues of A_m represent the wave speeds of the eight waves present in the MHD system,

$$\begin{aligned}
\lambda^{1,8} &= u_m \mp c_f &&: \text{fast magnetosonic waves,}\\
\lambda^{2,7} &= u_m \mp c_a &&: \text{Alfvén waves,}\\
\lambda^{3,6} &= u_m \mp c_s &&: \text{slow magnetosonic waves,}\\
\lambda^{4} &= u_m &&: \text{entropy wave,}\\
\lambda^{5} &= u_m &&: \text{divergence wave,}
\end{aligned}$$

where

$$\begin{aligned}
a &\equiv \sqrt{\frac{\gamma p}{\rho}},\\
c_a &\equiv \sqrt{\frac{B_m^2}{\rho}},\\
c_f &\equiv \left\{ \frac{1}{2}\left[a^2 + \frac{|\vec{B}|^2}{\rho} + \sqrt{\left(a^2 + \frac{|\vec{B}|^2}{\rho} \right)^2 - 4a^2 \frac{B_m^2}{\rho}} \right] \right\}^{1/2},\\
c_s &\equiv \left\{ \frac{1}{2}\left[a^2 + \frac{|\vec{B}|^2}{\rho} - \sqrt{\left(a^2 + \frac{|\vec{B}|^2}{\rho} \right)^2 - 4a^2 \frac{B_m^2}{\rho}} \right] \right\}^{1/2}.
\end{aligned}$$

The eigenvalues are well ordered in the sense that

$$\lambda^1 \le \lambda^2 \le \lambda^3 \le \lambda^4 \le \lambda^5 \le \lambda^6 \le \lambda^7 \le \lambda^8 .$$

The fast and slow magnetosonic waves are genuinely nonlinear, while the remaining waves are linearly degenerate.

Although it is possible to compute the Roe averages for the ideal MHD equations using the strategy of Cargo and Callice [9], we will instead base our approximate Riemann solver on simple arithmetic averages of the primitive variables. These are computationally less expensive to evaluate than the Roe averages presented in [9] and give good accuracy in practice (see section 7).

3. The high-resolution wave propagation method. The CT schemes that will be discussed in sections 4 and 5 all require an underlying [illegible] for solving the ideal MHD equations. The base scheme considered in this work is the high-resolution wave propagation method of LeVeque [27] in conjunction with the flux-based wave decomposition approach of [4]. The wave propagation method is based on solving 1D Riemann problems and then using the solutions from these Riemann problems to obtain an update that is second order accurate for smooth solutions, numerically conservative, shock-capturing, and stable up to a CFL number of one. This method has in the past been successfully applied to several application problems, including acoustics [15, 17], linear elasticity [16], nonlinear elasticity [29], shallow water waves [28], incompressible fluid dynamics [25], combustion and detonation waves [22, 23], relativistic hydrodynamics [3], and numerical relativity [7]. We should note that although we make use of this particular base scheme, we argue in section 5 that the CT method proposed in this work can be used independent of this choice.

The wave propagation of LeVeque [27] is a numerical method for approximately solving hyperbolic conservation laws of the form

$$\frac{\partial q}{\partial t} + \frac{\partial}{\partial x} f^1(q) + \frac{\partial}{\partial y} f^2(q) = 0. \tag{3.1}$$

The method can be obtained by first constructing a second order accurate Taylor series in time,

$$\begin{aligned} q(x,y,t+\Delta t) &= q(x,y,t) + \Delta t \frac{\partial q}{\partial t}(x,y,t) + \frac{1}{2}\Delta t^2 \frac{\partial^2 q}{\partial t^2}(x,y,t) + \mathcal{O}(\Delta t^3) \\ &= q - \Delta t \left(f^1_x + f^2_y\right) + \frac{1}{2}\Delta t^2 \Big\{ \left(A f^1_x\right)_x + \left(B f^2_y\right)_y \\ &\quad + \left(A f^2_y\right)_x + \left(B f^1_x\right)_y \Big\} + \mathcal{O}(\Delta t^3), \end{aligned} \tag{3.2}$$

where $A = \partial f^1/\partial q$, $B = \partial f^2/\partial q$, and we have used the evolution equation to turn time derivatives into spatial derivatives. After discretizing in space, the general update in two dimensions is

$$\begin{aligned} Q^{n+1}_{ij} = Q^n_{ij} &- \frac{\Delta t}{\Delta x}\left[\mathcal{A}^-_1 \Delta Q_{i+1/2,j} + \mathcal{A}^+_1 \Delta Q_{i-1/2,j}\right] \\ &- \frac{\Delta t}{\Delta y}\left[\mathcal{A}^-_2 \Delta Q_{i,j+1/2} + \mathcal{A}^+_2 \Delta Q_{i,j-1/2}\right] \\ &- \frac{\Delta t}{\Delta x}\left[\tilde{F}^1_{i+1/2,j} - \tilde{F}^1_{i-1/2,j}\right] - \frac{\Delta t}{\Delta y}\left[\tilde{F}^2_{i,j+1/2} - \tilde{F}^2_{i,j-1/2}\right], \end{aligned} \tag{3.3}$$

where $\mathcal{A}^-\Delta Q$ and $\mathcal{A}^+\Delta Q$ are [illegible] and $\tilde{F}^1$ and $\tilde{F}^2$ are [illegible]. The wave propagation method is constructed so that the fluctuations contain corrections needed for a first order upwind update (approximates terms with one x or one y derivative) and $\tilde{F}^1$ and $\tilde{F}^2$ contain higher-resolution (approximates terms with two x or two y derivatives) and multidimensional corrections (approximates terms with cross derivatives) [27, 30].

3.1. First order upwind update. The fluctuations are constructed by solving locally linearized Riemann problems. At each cell interface $(x_{i-1/2}, y_j)$ we have to solve a 1D Riemann problem of the form

$$\frac{\partial q}{\partial t} + \tilde{A}_{i-1/2,j} \frac{\partial q}{\partial x} = 0, \tag{3.4}$$

$$q(x,0) = \begin{cases} Q_{i-1,j} & \text{if } x < x_{i-1/2}, \\ Q_{ij} & \text{if } x > x_{i-1/2}, \end{cases} \tag{3.5}$$

$$f^1(q,x) = \begin{cases} f^1_{i-1,j}(q) & \text{if } x < x_{i-1/2}, \\ f^1_{ij}(q) & \text{if } x > x_{i-1/2}, \end{cases} \tag{3.6}$$

where

$$\tilde{A}_{i-1/2,j} \approx \frac{\partial f^1}{\partial q}\left(q(\vec{x}_{i-1/2,j}, t), x_{i-1/2}, y_j\right) \tag{3.7}$$

is an approximate flux Jacobian that is considered to be constant in the Riemann problem. We denote the eigenvalues, right eigenvectors, and left eigenvectors of $\tilde{A}_{i-1/2,j}$ by $s^p_{i-1/2,j}$, $r^p_{i-1/2,j}$, and $\ell^p_{i-1/2,j}$.

In order to solve Riemann problem (3.4)–(3.5), the flux difference is decomposed along the eigenvectors of $\tilde{A}_{i-1/2,j}$. Based on these eigenvectors and eigenvalues, the fluctuations $\mathcal{A}_1^- \Delta Q_{i-1/2,j}$ and $\mathcal{A}_1^+ \Delta Q_{i-1/2,j}$ in (3.3) are assembled by summing all the components of the flux that are left- and right-going, respectively. Mathematically, we write

$$\Delta F^1_{i-1/2,j} = f^1_{ij}(Q_{ij}) - f^1_{i-1,j}(Q_{i-1,j}), \tag{3.8}$$

$$\mathcal{Z}^p_{i-1/2,j} = \left[\ell^p_{i-1/2,j} \cdot \Delta F^1_{i-1/2,j}\right] r^p_{i-1/2,j}, \tag{3.9}$$

$$\mathcal{A}_1^+ \Delta Q_{i-1/2,j} = \sum_{p:\, s^p_{i-1/2,j} > 0} \mathcal{Z}^p_{i-1/2,j} + \frac{1}{2} \sum_{p:\, s^p_{i-1/2,j} = 0} \mathcal{Z}^p_{i-1/2,j}, \tag{3.10}$$

$$\mathcal{A}_1^- \Delta Q_{i-1/2,j} = \sum_{p:\, s^p_{i-1/2,j} < 0} \mathcal{Z}^p_{i-1/2,j} + \frac{1}{2} \sum_{p:\, s^p_{i-1/2,j} = 0} \mathcal{Z}^p_{i-1/2,j}, \tag{3.11}$$

where we refer to $\mathcal{Z}^p_{i-1/2}$ as a [illegible] and $s^p_{i-1/2,j}$ as a [illegible]. An analogous procedure is carried out in the 2-direction in order to obtain $\mathcal{A}_2^- \Delta Q_{i,j-1/2}$ and $\mathcal{A}_2^+ \Delta Q_{i,j-1/2}$.

It is well known that linearized Riemann solvers can fail in the case where the exact solution to the Riemann problem is a transonic rarefaction (see [30] for more discussion). In order to avoid this difficulty we make use of an entropy fix similar to the one introduced by Harten and Hyman [21].

3.2. Correction fluxes. To obtain second order accuracy for smooth flows, [illegible] corrections of the following form are constructed:

$$\tilde{F}^1_{i-1/2,j} = \frac{1}{2} \sum_{p=1}^{M} \operatorname{sign}(s^p_{i-1/2,j}) \left(1 - \frac{\Delta t}{\Delta x} \left| s^p_{i-1/2,j} \right| \right) \mathcal{Z}^p_{i-1/2}\, \phi_{i-1/2,j}\,, \tag{3.12}$$

where ϕ is a total variation diminishing (TVD) wave limiter function. The wave limiter allows the method to switch between a second order method in smooth regions of the solution and a first order method near discontinuities [30]. To make this method stable up to a CFL number of one without dimensional splitting, [illegible] corrections based on solving transverse Riemann problems are added to the numerical fluxes. For more details on these corrections, we refer the reader to [26, 27, 30].

4. Constrained transport. Because the magnetic field satisfies a divergence-free condition, it can always be written as the curl of a magnetic potential:

$$\vec{B} = \vec{\nabla} \times \vec{\mathbb{A}}. \tag{4.1}$$

Using the magnetic potential, the evolution equations for the magnetic field in (1.1), this can be rewritten as follows:

$$\frac{\partial \vec{\mathbb{A}}}{\partial t} + \vec{\Omega} = 0, \tag{4.2}$$

where $\vec{\Omega}$ is the electric field, and

$$\vec{\Omega} = \vec{B} \times \vec{u} \tag{4.3}$$

is Ohm's law for a perfect conductor [19].

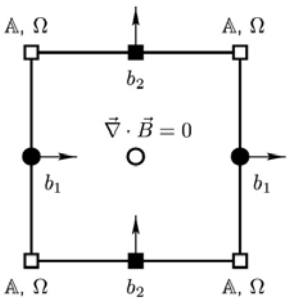


FIG. 4.1. *Grid staggering for the CT framework of Evans and Hawley* [13].

In 2 or 2.5 dimensions[1] we need only keep track of the z-components of the magnetic potential, $\mathbb{A}$, and the electric field, Ω, such that

$$\Omega = (\vec{B}\times\vec{u})\cdot\hat{k} = -u_1B_2 + u_2B_1, \tag{4.4}$$

and

$$B_1 = \frac{\partial\mathbb{A}}{\partial y} \qquad \text{and} \qquad B_2 = -\frac{\partial\mathbb{A}}{\partial x}.$$

Note that in 2.5 dimensions, the B_3 component of the magnetic field does not need to be computed from a potential, since the divergence of $\vec{B}$ in 2.5 dimensions involves only B_1 and B_2.

Independent of the dimension, the key property of the magnetic potential is that if it is known, a divergence-free magnetic field can be directly computed from it. The constrained transport method introduced by Evans and Hawley [13] tries to take advantage of this property on a discrete numerical grid.

4.1. Staggered constrained transport. Evans and Hawley [13] introduced the following numerical grid staggering (see Figure 4.1):

$$\begin{aligned}
\text{at cell centers:} &\quad \rho_{ij},\ \rho\vec{u}_{ij},\ \mathcal{E}_{ij},\ \vec{B}_{ij},\\
\text{at cell corners:} &\quad \mathbb{A}_{i-1/2,j-1/2},\ \Omega_{i-1/2,j-1/2},\\
\text{at left/right edges:} &\quad [b_1]_{i-1/2,j},\\
\text{at top/bottom edges:} &\quad [b_2]_{i,j-1/2}\,.
\end{aligned}$$

If the magnetic potential is known, then the magnetic field at the cell edges can be computed in the following way:

$$[b_1]_{i-1/2,j} = \frac{1}{\Delta y}\Big(\mathbb{A}_{i-1/2,j+1/2} - \mathbb{A}_{i-1/2,j-1/2}\Big), \tag{4.5}$$

$$[b_2]_{i,j-1/2} = \frac{1}{\Delta x}\Big(\mathbb{A}_{i-1/2,j-1/2} - \mathbb{A}_{i+1/2,j-1/2}\Big). \tag{4.6}$$

It can be easily shown that with this definition of the magnetic field, the following discrete divergence-free condition is satisfied to machine precision:

$$\Big[\vec{\nabla}\cdot\vec{B}\Big]_{ij} \equiv \frac{[b_1]_{i+1/2,j} - [b_1]_{i-1/2,j}}{\Delta x} + \frac{[b_2]_{i,j+1/2} - [b_2]_{i,j-1/2}}{\Delta y} = 0. \tag{4.7}$$

[1]By 2.5 dimensions we mean that all three components of $\vec{u}$ and/or $\vec{B}$ are nonzero, but that all these components are functions of only x, y, and t.

In order to update the fluid variables, however, we also need a cell-centered representation of the magnetic field. This can be obtained through simple linear interpolation:

$$[B_1]_{ij} = \frac{1}{2}\Big([b_1]_{i+1/2,j} + [b_1]_{i-1/2,j}\Big), \tag{4.8}$$

$$[B_2]_{ij} = \frac{1}{2}\Big([b_2]_{i,j+1/2} + [b_2]_{i,j-1/2}\Big). \tag{4.9}$$

Tóth [38] showed that with these definitions, the cell-centered magnetic field also satisfies a discrete divergence-free condition:

$$\begin{aligned}\Big[\vec{\nabla}\cdot\vec{B}\Big]_{i+1/2,j+1/2} &\equiv \frac{[B_1]_{i+1,j} + [B_1]_{i+1,j+1} - [B_1]_{ij} - [B_1]_{i,j+1}}{2\Delta x} \\ &+ \frac{[B_2]_{i,j+1} + [B_2]_{i+1,j+1} - [B_2]_{ij} - [B_2]_{i+1,j}}{2\Delta y} = 0.\end{aligned} \tag{4.10}$$

In order to obtain a complete numerical method for the MHD equations, an update for the magnetic potential that couples to the other variables must be provided.

The full constrained transport algorithm can be summarized as follows:

Step 0. Primary quantities: $\rho^n,\ \rho\vec{u}^{\,n},\ \mathcal{E}^n,\ b_1^n,\ b_2^n,\ B_3^n,\ \mathbb{A}^n$.

Step 1. Average edge magnetic field:

$$\Longrightarrow [B_1]_{ij}^n = \tfrac{1}{2}\Big([b_1]_{i-1/2,j}^n + [b_1]_{i+1/2,j}^n\Big),$$
$$\Longrightarrow [B_2]_{ij}^n = \tfrac{1}{2}\Big([b_2]_{i,j-1/2}^n + [b_2]_{i,j+1/2}^n\Big).$$

Step 2. Solve the MHD equations using the base scheme:

$$\Longrightarrow \rho^{n+1},\ \rho\vec{u}^{\,n+1},\ \mathcal{E}^{n+1},\ \vec{B}^{\star}.$$

Step 3. Construct electric field at cell corners: $\Omega_{i-1/2,j-1/2}^{n+1/2}$.

Step 4. Update the magnetic potential:

$$\Longrightarrow \mathbb{A}_{i-1/2,j-1/2}^{n+1} = \mathbb{A}_{i-1/2,j-1/2}^{n} - \Delta t\,\Omega_{i-1/2,j-1/2}^{n+1/2}.$$

Step 5. Compute $\vec{\nabla}\times\vec{\mathbb{A}}$:

$$\Longrightarrow [b_1]_{i-1/2,j}^{n+1} = \tfrac{1}{\Delta y}\Big(\mathbb{A}_{i-1/2,j+1/2}^{n+1} - \mathbb{A}_{i-1/2,j-1/2}^{n+1}\Big),$$
$$\Longrightarrow [b_2]_{i,j-1/2}^{n+1} = \tfrac{1}{\Delta x}\Big(\mathbb{A}_{i-1/2,j-1/2}^{n+1} - \mathbb{A}_{i+1/2,j-1/2}^{n+1}\Big).$$

In other words, the algorithm says that we begin with cell-centered quantities ρ^n, $\rho\vec{u}^n$, b_1^n, b_2^n, B_3^n, and $\mathbb{A}^n$. The magnetic field is averaged to cell centers so that the ideal MHD equations can be solved with a high-resolution finite volume method. After updating the cell-centered quantities, an electric field at cell corners is constructed. This electric field is then used to update the magnetic potential. Finally, the magnetic potential is differenced and we obtain the updated magnetic field values.

In the above description of CT each step is clearly defined, with the exception of Step 3; in this step there are many options for how the electric field is constructed. Several methods have been proposed in the literature [6, 10, 36]. See Tóth [38] for a review. Additional staggered CT methods that are closely related to these methods but do not conform to the algorithm as described above were developed by De Sterck [12], Londrillo and Del Zanna [32, 33], and Balsara [5].

4.2. Unstaggered CT. One of the main drawbacks of the Evans and Hawley [13] formulation coupled to a cell-centered finite volume method is that one needs a dual representation of the magnetic field. In an adaptive mesh refinement framework, especially in three space dimensions, this makes the method somewhat cumbersome. In order to overcome this, Tóth [38] introduced fully cell-centered versions of the Evans and Hawley approach. These approaches do not require a cell edge magnetic field, and instead of conserving the divergence defined as in (4.7) they conserve the following cell-centered definition of the divergence:

$$\left[\vec{\nabla}\cdot\vec{B}\right]_{ij} \equiv \frac{[B_1]_{i+1,j}-[B_1]_{i-1,j}}{2\Delta x}+\frac{[B_2]_{i,j+1}-[B_2]_{i,j-1}}{2\Delta y}=0. \tag{4.11}$$

This is accomplished through the following definition of the magnetic field:

$$[B_1]_{ij}=\frac{1}{2\Delta y}\Big(\mathbb{A}_{i,j+1}-\mathbb{A}_{i,j-1}\Big), \tag{4.12}$$

$$[B_2]_{ij}=\frac{1}{2\Delta x}\Big(\mathbb{A}_{i-1,j}-\mathbb{A}_{i+1,j}\Big). \tag{4.13}$$

In this case, Step 1 in the constrained algorithm described in section 4.1 is completely eliminated. The modified algorithm can be written as follows:

Step 0. Primary quantities: $\rho^n,\ \rho\vec{u}^{\,n},\ \mathcal{E}^n,\ B_1^n,\ B_2^n,\ B_3^n,\ \mathbb{A}^n$.

Step 1. Solve the MHD equations using the base scheme:
$\Longrightarrow \rho^{n+1},\ \rho\vec{u}^{\,n+1},\ \mathcal{E}^{n+1},\ \vec{B}^{\star}$.

Step 2. Construct electric field at cell centers: $\Omega_{ij}^{n+1/2}$.

Step 3. Update the magnetic potential:
$\Longrightarrow \mathbb{A}_{ij}^{n+1}=\mathbb{A}_{ij}^{n}-\Delta t\,\Omega_{ij}^{n+1/2}$.

Step 4. Compute $\vec{\nabla}\times\vec{\mathbb{A}}$:
$$\Longrightarrow [B_1]_{ij}^{n+1}=\tfrac{1}{2\Delta y}\Big(\mathbb{A}_{i,j+1}^{n+1}-\mathbb{A}_{i,j-1}^{n+1}\Big),$$
$$\Longrightarrow [B_2]_{ij}^{n+1}=\tfrac{1}{2\Delta x}\Big(\mathbb{A}_{i-1,j}^{n+1}-\mathbb{A}_{i+1,j}^{n+1}\Big).$$

Tóth [38] introduced two separate techniques for constructing the electric field in Step 3; one is a variant of Balsara and Spicer [6], the other a variant of Dai and Woodward [10]. Tóth [38] found that the variant of Balsara and Spicer [6] is less accurate than the original method, because it introduces additional averaging of the numerical fluxes. The variant of Dai and Woodward [10] has less averaging than the original method and was therefore found to be more accurate.

4.3. Computing the initial potential. CT schemes require an initial magnetic potential. In many cases, however, the initial conditions will only be given in terms of the magnetic field, and it may be difficult to obtain an analytic form for the initial $\mathbb{A}$. In this case, we can compute the magnetic potential by solving a Poisson equation of the form [24]

$$\vec{\nabla}\times\vec{B}=\vec{\nabla}\times\vec{\nabla}\times\vec{\mathbb{A}}=\vec{\nabla}\left(\vec{\nabla}\cdot\vec{\mathbb{A}}\right)-\nabla^2\vec{\mathbb{A}},$$
$$\Longrightarrow\quad \nabla^2\vec{\mathbb{A}}=-\vec{\nabla}\times\vec{B},$$

where we have assumed, without loss of generality, that the initial magnetic potential is solenoidal. Note that the magnetic potential at later times will in general not satisfy $\vec{\nabla} \cdot \vec{\mathbb{A}} = 0$. Once the initial magnetic potential has been determined, the initial magnetic field should be redefined as a finite difference of the magnetic potential so that it satisfies the discrete divergence-free condition.

5. A new CT method. The advantage of the CT framework is that it allows one to control the divergence of the magnetic field, which results in control over the Lorentz force along the magnetic field. This makes CT much more robust than standard cell-centered finite volume methods for the ideal MHD equations. The drawback of CT methods is that by constructing the electric field in the various ways proposed in the literature and then applying centered differences to compute the magnetic fields, one is not guaranteed that the resulting magnetic field satisfies any kind of TVD property. The staggered methods of Londrillo and Del Zanna [32, 33] and Balsara [5] address this issue through more careful reconstructions of the electric field. In this section we develop an alternative to these methods. The proposed method has the following basic features:

1. The evolution equation for the magnetic potential is written as a system of first order hyperbolic equations.
2. The magnetic potential is updated with a high-resolution upwind method.
3. Limiters are applied in such a way as to minimize the oscillations in both the magnetic potential and the magnetic field.
4. The method is naturally unstaggered.

5.1. The magnetic potential equation. The equation for the magnetic potential, (4.2), can be rewritten as a partial differential equation (PDE) by applying Ohm's law (4.3) and the definition of the magnetic potential (4.1). In two dimensions this PDE is a scalar advection equation of the form

$$\frac{\partial \mathbb{A}}{\partial t} + u_1 \frac{\partial \mathbb{A}}{\partial x} + u_2 \frac{\partial \mathbb{A}}{\partial y} = 0\,. \tag{5.1}$$

In three dimensions the magnetic potential satisfies a system of PDEs:

$$\frac{\partial \vec{\mathbb{A}}}{\partial t} + N_1(\vec{u}) \frac{\partial \vec{\mathbb{A}}}{\partial x} + N_2(\vec{u}) \frac{\partial \vec{\mathbb{A}}}{\partial y} + N_3(\vec{u}) \frac{\partial \vec{\mathbb{A}}}{\partial z} = 0, \tag{5.2}$$

where N_1, N_2, and N_3 are 3×3 matrices that depend only on the fluid velocity $\vec{u}$. If we assume that the fluid velocity is given, then both (5.1) and (5.2) are linear, variable coefficient, hyperbolic PDEs.

5.2. The MPACT scheme. In this section we propose a new CT scheme that is based on directly solving (5.1) or (5.2) with a properly chosen high-resolution method. It is natural to solve the magnetic potential equation on the same numerical grid as the MHD equations, and therefore we will use an unstaggered representation of the magnetic field. The proposed CT scheme can be summarized as follows:

Step 0. Primary quantities: $\rho^n,\ \rho\vec{u}^{\,n},\ \mathcal{E}^n,\ \vec{B}^{\,n},\ \mathbb{A}^n$.

Step 1. Solve the MHD equations using the base scheme:
$\Longrightarrow \rho^{n+1},\ \rho\vec{u}^{\,n+1},\ \mathcal{E}^{n+1},\ \vec{B}^{\star}$.
(NOTE: $\vec{B}^{\star}$ is discarded and need not be stored.)

Step 2. Time average the velocity field:
$$\Longrightarrow \vec{u}^{\,n+1/2} = \tfrac{1}{2}\left(\vec{u}^{\,n} + \vec{u}^{\,n+1}\right).$$

Step 3. Solve the magnetic potential equation (5.1) using $\vec{u}^{\,n+1/2}$: $\mathbb{A}^{n+1}$.

Step 4. Compute $\vec{\nabla} \times \vec{\mathbb{A}}$:
$$\Longrightarrow [B_1]_{ij}^{n+1} = \tfrac{1}{2\Delta y}\left(\mathbb{A}_{i,j+1}^{n+1} - \mathbb{A}_{i,j-1}^{n+1}\right),$$
$$\Longrightarrow [B_2]_{ij}^{n+1} = \tfrac{1}{2\Delta x}\left(\mathbb{A}_{i-1,j}^{n+1} - \mathbb{A}_{i+1,j}^{n+1}\right).$$

In other words, this algorithm tells us that in Step 1 all of the primitive variables are updated self-consistently. The only problem is that the resulting magnetic field is not divergence-free. In order to correct this problem, we also advance the magnetic potential forward in time (Step 3). When we advance the magnetic potential forward in time, the new velocity field is no longer an unknown since we have already computed it self-consistently. Therefore, we can solve for the magnetic potential simply by advecting it along the computed velocity field. Once we have the magnetic potential at the new time level, a new magnetic field is computed from the potential in such a way that it satisfies a discrete divergence-free condition (Step 4). The coupling between the fluid variables and the magnetic potential comes from the fact that in the update for the fluid variables we need to use the magnetic field, which comes from the potential.

We refer to this method as the MPACT (magnetic potential advection constrained transport) scheme. Definition (4.11) of the divergence of the magnetic field is exactly conserved by this method. Note that we must use the intermediate value of the velocity, $\vec{u}^{\,n+1/2}$, in Step 3 in order to guarantee full second order accuracy. Also note that the only step in the above algorithm that requires further explanation is Step 3.

In section 5.3 we explain how one can solve the magnetic potential equation (5.1) to second order accuracy in the wave propagation framework. In section 5.4 we devise a new limiting strategy in order to obtain both a magnetic potential and a magnetic field that remains approximately TVD. In section 6 quantitative arguments are presented that explain why this method produces better approximations to the ideal MHD equations than the base scheme alone. This method is tested on several two-dimensional (2D) examples in section 7.

5.3. A modified wave propagation method. Although the magnetic potential equation is not a conservation law, it can still be solved using the wave propagation method of LeVeque [27]. However, LeVeque [27] showed that the wave propagation method is only formally first order accurate for variable coefficient problems written in quasilinear form—the standard wave program method is second order accurate for conservation laws. We present in this section a modified wave propagation method that is formally second order for variable coefficient, linear advection. Extension to linear hyperbolic systems and to three dimensions is straightforward.

In Step 3 of the MPACT scheme we are faced with solving the following equation:

$$\frac{\partial q}{\partial t} + u_1(x,y)\,\frac{\partial q}{\partial x} + u_2(x,y)\,\frac{\partial q}{\partial y} = 0\,, \tag{5.3}$$

where u_1 and u_2 are time averaged onto time level $t = t^{n+1/2}$. Just as in the wave

propagation method, we construct a second order Taylor expansion in time:

$$\begin{aligned} q(x,y,t+\Delta t) &= q(x,y,t) + \Delta t \frac{\partial q}{\partial t}(x,y,t) + \frac{1}{2}\Delta t^2 \frac{\partial^2 q}{\partial t^2}(x,y,t) + \mathcal{O}(\Delta t^3) \\ &= q - \Delta t \left(u_1 q_x + u_2 q_y\right) + \frac{1}{2}\Delta t^2 \Big\{ u_1 \left(u_1 q_x\right)_x + u_2 \left(u_2 q_y\right)_y \\ &\quad + u_1 \left(u_2 q_y\right)_x + u_2 \left(u_1 q_x\right)_y \Big\} + \mathcal{O}(\Delta t^3), \end{aligned} \tag{5.4}$$

where we have used the evolution equation to turn time derivatives into spatial derivatives. We will try to approximate the terms in this expansion with a modified wave propagation method of the form

$$\begin{aligned} Q_{ij}^{n+1} = Q_{ij}^n &- \frac{\Delta t}{\Delta x}\left[\mathcal{A}_1^- \Delta Q_{i+1/2,j} + \mathcal{A}_1^+ \Delta Q_{i-1/2,j}\right] \\ &- \frac{\Delta t}{\Delta y}\left[\mathcal{A}_2^- \Delta Q_{i,j+1/2} + \mathcal{A}_2^+ \Delta Q_{i,j-1/2}\right] \\ &- \frac{\Delta t}{\Delta x}\left[\tilde{F}^{1-}_{i+1/2,j} - \tilde{F}^{1+}_{i-1/2,j}\right] - \frac{\Delta t}{\Delta y}\left[\tilde{F}^{2-}_{i,j+1/2} - \tilde{F}^{2+}_{i,j-1/2}\right], \end{aligned} \tag{5.5}$$

where the $\mathcal{A}^{\pm}\Delta Q$ terms contain first order upwind corrections (approximates terms with one x or one y derivative) and the $\tilde{F}$ terms contain high-resolution (approximates terms with two x or two y derivatives) and multidimensional corrections (approximates terms with cross derivatives). What distinguishes this method from the wave propagation method of [27] is that for each cell interface, $(x_{i-1/2}, y_j)$, we define two distinct correction fluxes, $\tilde{F}^{1-}_{i-1/2,j}$ and $\tilde{F}^{1+}_{i-1/2,j}$. $\tilde{F}^{1-}_{i-1/2,j}$ is associated with the solution in grid cell (x_{i-1}, y_j), while $\tilde{F}^{1+}_{i-1/2,j}$ is associated with the solution in grid cell (x_i, y_j).

In the modified method, we take first order corrections of the form

$$\mathcal{A}_1^- \Delta Q_{i-1/2,j} = [s_{i-1,j}]^- \, \mathcal{W}_{i-1/2,j}, \tag{5.6}$$

$$\mathcal{A}_1^+ \Delta Q_{i-1/2,j} = [s_{ij}]^+ \, \mathcal{W}_{i-1/2,j}, \tag{5.7}$$

where

$$[s]^- = \begin{cases} s & \text{if} \quad s < 0, \\ 0 & \text{otherwise,} \end{cases} \qquad [s]^+ = \begin{cases} s & \text{if} \quad s > 0, \\ 0 & \text{otherwise,} \end{cases} \tag{5.8}$$

$$s_{i-1/2,j} = [u_1]_{i-1/2,j}\,, \tag{5.9}$$

$$\mathcal{W}_{i-1/2,j} = (Q_{ij} - Q_{i-1,j})\,. \tag{5.10}$$

We refer to $\mathcal{W}$ as a [illegible] and s as a [illegible]. The high-resolution correction fluxes are given by

$$\tilde{F}^{1-}_{i-1/2,j} = \frac{1}{2}\left|s_{i-1,j}\right|\left(1 - \frac{\Delta t}{\Delta x}\frac{s_{i-1/2,j}}{\left|s_{i-1,j}\right|}\, s_{i-1,j}\right)\mathcal{W}_{i-1/2,j}, \tag{5.11}$$

$$\tilde{F}^{1+}_{i-1/2,j} = \frac{1}{2}\left|s_{ij}\right|\left(1 - \frac{\Delta t}{\Delta x}\frac{s_{i-1/2,j}}{\left|s_{ij}\right|}\, s_{ij}\right)\mathcal{W}_{i-1/2,j}\,, \tag{5.12}$$

where $s_{i-1/2,j}$ can be taken as the arithmetic average of s_{ij} and $s_{i-1,j}$. Finally, in order to achieve full second order accuracy and stability up to a CFL number of one,

multidimensional corrections must be included. These multidimensional corrections can be written as

$$\begin{aligned}\tilde{F}^{1-}_{i+1/2} = \tilde{F}^{1-}_{i+1/2} - \frac{1}{2}\frac{\Delta t}{\Delta y}\Big(& [u_1]^+_i \left(u_2^+ \Delta Q - 2\tilde{F}^{2+}\right)_{i,j-1/2} \\ & + [u_1]^-_i \left(u_2^+ \Delta Q - 2\tilde{F}^{2+}\right)_{i+1,j-1/2} \\ & + [u_1]^+_i \left(u_2^- \Delta Q + 2\tilde{F}^{2-}\right)_{i,j+1/2} \\ & + [u_1]^-_i \left(u_2^- \Delta Q + 2\tilde{F}^{2-}\right)_{i+1,j+1/2}\Big),\end{aligned} \tag{5.13}$$

where $\tilde{F}^{2-}$ are the high-resolution fluxes in the 2-direction.

CLAIM. [illegible] $\Delta t = c_1 \Delta x = c_2 \Delta y$ [illegible] c_1 [illegible] c_2 [illegible] $q(x,y,t)$ [illegible] (x,y,t) [illegible] $\mathcal{O}(\Delta t^2)$

This claim is proved in the appendix.

5.4. A new limiting strategy for obtaining a nonoscillatory magnetic field. The aim of this paper is to develop a CT method that produces a nonoscillatory magnetic field. The modified wave propagation method developed so far offers a new approach for updating the magnetic potential. However, if standard wave limiters are applied to the magnetic potential, unphysical oscillations can still be introduced into the magnetic field. In order to overcome this problem we must introduce a new limiting strategy.

The [illegible] limiting strategy of LeVeque [27], adapted to the modified wave propagation method presented in this work, results in the following correction fluxes.

[illegible] "

$$\tilde{F}^{1-}_{i-1/2} = \frac{1}{2}\left|s_{i-1}\right|\left(1 - \frac{\Delta t}{\Delta x}\frac{s_{i-1/2}}{\left|s_{i-1}\right|}\, s_{i-1}\right)\mathcal{W}_{i-1/2}\,\phi_{i-1/2}, \tag{5.14}$$

$$\tilde{F}^{1+}_{i-1/2} = \frac{1}{2}\left|s_{i}\right|\left(1 - \frac{\Delta t}{\Delta x}\frac{s_{i-1/2}}{\left|s_{i}\right|}\, s_{i}\right)\mathcal{W}_{i-1/2}\,\phi_{i-1/2}, \tag{5.15}$$

$$\phi_{i-1/2} = \phi\left(\frac{\mathcal{W}_{I-1/2}\cdot\mathcal{W}_{i-1/2}}{\mathcal{W}_{i-1/2}\cdot\mathcal{W}_{i-1/2}}\right), \tag{5.16}$$

where $\phi(\theta)$ is a TVD limiter function and $\mathcal{W}_{I-1/2}$ is chosen from the upwind direction based on $s_{i-1/2}$ [30]. This approach has the following effect:

[illegible] $\Longrightarrow$ [illegible]

Since we are more concerned with obtaining a nonoscillatory derivative, we propose the following approach:

[illegible] $\Longrightarrow$ [illegible]

We note that such a procedure will yield accurate results only if the variable that is being updated remains continuous. If the variable is discontinuous, then limiting based on the first derivative will be too aggressive, yielding an overly diffusive approximation. Such a limiting strategy may work in our case, however, since the magnetic

potential is always continuous, while the magnetic field is discontinuous wherever the potential fails to be differentiable.

Applying the wave-difference limiting strategy results in the following correction fluxes.

$$\tilde{F}^{1-}_{i-1/2} = \frac{1}{2}\left|s_{i-1}\right|\left(1 - \frac{\Delta t}{\Delta x}\frac{s_{i-1/2}}{\left|s_{i-1}\right|}\, s_{i-1}\right)\mathcal{W}_{i-1/2}\,\phi_{i-1}, \tag{5.17}$$

$$\tilde{F}^{1+}_{i-1/2} = \frac{1}{2}\left|s_{i}\right|\left(1 - \frac{\Delta t}{\Delta x}\frac{s_{i-1/2}}{\left|s_{i}\right|}\, s_{i}\right)\mathcal{W}_{i-1/2}\,\phi_{i}, \tag{5.18}$$

$$\phi_i = \phi\left(\frac{\Delta\mathcal{W}_I \cdot \Delta\mathcal{W}_i}{\Delta\mathcal{W}_i \cdot \Delta\mathcal{W}_i}\right), \tag{5.19}$$

where $\Delta\mathcal{W}_i = \mathcal{W}_{i+1/2} - \mathcal{W}_{i-1/2}$ and $\Delta\mathcal{W}_I$ is chosen from the upwind direction based on s_i. As we demonstrate in an example below, this approach has the desired property of adding numerical diffusion whenever there is a "large" change in $\Delta\mathcal{W}$ even if there is a "small" change in $\mathcal{W}$. In particular, we can prove the following result.

CLAIM.

$$\frac{\partial q}{\partial t} + \frac{\partial q}{\partial x} = 0,$$

1.

$$B^n_{i+1/2} \equiv \frac{Q^n_{i+1} - Q^n_i}{\Delta x},$$

$$\text{TV}(B^{n+1}_{i+1/2}) \equiv \sum_{i=-\infty}^{\infty} |B^{n+1}_{i+1/2} - B^{n+1}_{i-1/2}| \le \text{TV}(B^n_{i+1/2}) \le \cdots \le \text{TV}(B^0_{i+1/2}).$$

2.

$$B^n_i \equiv \frac{1}{2}\left(B^n_{i+1/2} + B^n_{i-1/2}\right) = \frac{Q^n_{i+1} - Q^n_{i-1}}{2\Delta x},$$

$$\text{TV}(B^{n+1}_i) \equiv \sum_{i=-\infty}^{\infty} |B^{n+1}_{i+1} - B^{n+1}_i| \le \text{TV}(B^0_{i+1/2}).$$

The numerical update using the wave propagation method with the wave-difference limiting approach takes the following form for the simple constant coefficient advection equation:

$$Q^{n+1}_i = Q^n_i - \nu\left(Q^n_i - Q^n_{i-1}\right) - \frac{1}{2}\nu(1-\nu)\phi\left(\theta^n_i\right)\left(Q^n_{i+1} - 2Q^n_i + Q_{i-1}\right),$$

where

$$\theta^n_i = \frac{Q^n_i - 2Q^n_{i-1} + Q^n_{i-2}}{Q^n_{i+1} - 2Q^n_i + Q^n_{i-1}}.$$

1. If we take the above numerical update for Q_{i+1} and Q_i, subtract them from each other, and divide by Δx, we obtain the following update for $B_{i+1/2}$:

$$B^{n+1}_{i+1/2} = B^n_{i+1/2} - \nu\left(B^n_{i+1/2} - B^n_{i-1/2}\right) - \frac{1}{2}\nu(1-\nu)\left[\phi\left(\theta^n_i\right)\left(B^n_{i+3/2} - B^n_{i+1/2}\right) - \phi\left(\theta^n_{i-1}\right)\left(B^n_{i+1/2} - B^n_{i-1/2}\right)\right],$$

where

$$\theta^n_{i+1} = \frac{B^n_{i+1/2} - B^n_{i-1/2}}{B^n_{i+3/2} - B^n_{i+1/2}}.$$

This update can be rewritten in the following form:

$$B^{n+1}_{i+1/2} = B^n_{i+1/2} - C^n_{i-1/2}\left(B^n_{i+1/2} - B^n_{i-1/2}\right),$$

where

$$C^n_{i-1/2} = \nu + \frac{1}{2}\nu(1-\nu)\left(\frac{\phi(\theta^n_{i+1})}{\theta^n_{i+1}} - \phi(\theta^n_i)\right).$$

We know from the theorem of Harten [20] that the total variation of $B_{i+1/2}$ is nonincreasing if $0 \le C^n_{i-1/2} \le 1$ for all i. We know from the work of Sweby [37] that the standard TVD limiters (i.e., minmod, superbee, monotonized centered, and van Leer) satisfy this constraint. Therefore, we have shown part (1) of the claim.

2. We note that

$$\begin{aligned}\mathrm{TV}(B^{n+1}_i) &\equiv \sum_{i=-\infty}^{\infty} |B^{n+1}_{i+1} - B^{n+1}_i| \\ &= \frac{1}{2}\sum_{i=-\infty}^{\infty} |B^{n+1}_{i+3/2} + B^{n+1}_{i+1/2} - B^{n+1}_{i+1/2} - B^{n+1}_{i-1/2}| \\ &\le \frac{1}{2}\sum_{i=-\infty}^{\infty} |B^{n+1}_{i+3/2} - B^{n+1}_{i+1/2}| + \frac{1}{2}\sum_{i=-\infty}^{\infty} |B^{n+1}_{i+1/2} - B^{n+1}_{i-1/2}| = \mathrm{TV}(B^{n+1}_{i+1/2}).\end{aligned}$$

Using the result from part (1) it follows that

$$\mathrm{TV}(B^{n+1}_i) \le \mathrm{TV}(B^{n+1}_{i+1/2}) \le \mathrm{TV}(B^n_{i+1/2}) \le \cdots \le \mathrm{TV}(B^0_{i+1/2}). \qquad \square$$

In order to test this new limiting strategy, let us consider the linear advection equation,

$$\frac{\partial q}{\partial t} + \frac{\partial q}{\partial x} = 0, \tag{5.20}$$

on the interval $0 \le x \le 1$ with periodic boundary conditions. We take as initial conditions the following continuous and piecewise linear function:

$$q(x,0) = \begin{cases} 0 & \text{if} \quad x \le 0.25, \\ (x - 0.25)/0.075 & \text{if} \quad 0.25 \le x \le 0.4, \\ 2 & \text{if} \quad 0.4 \le x \le 0.6, \\ (0.75 - x)/0.075 & \text{if} \quad 0.6 \le x \le 0.75, \\ 0 & \text{if} \quad 0.75 \le x. \end{cases} \tag{5.21}$$

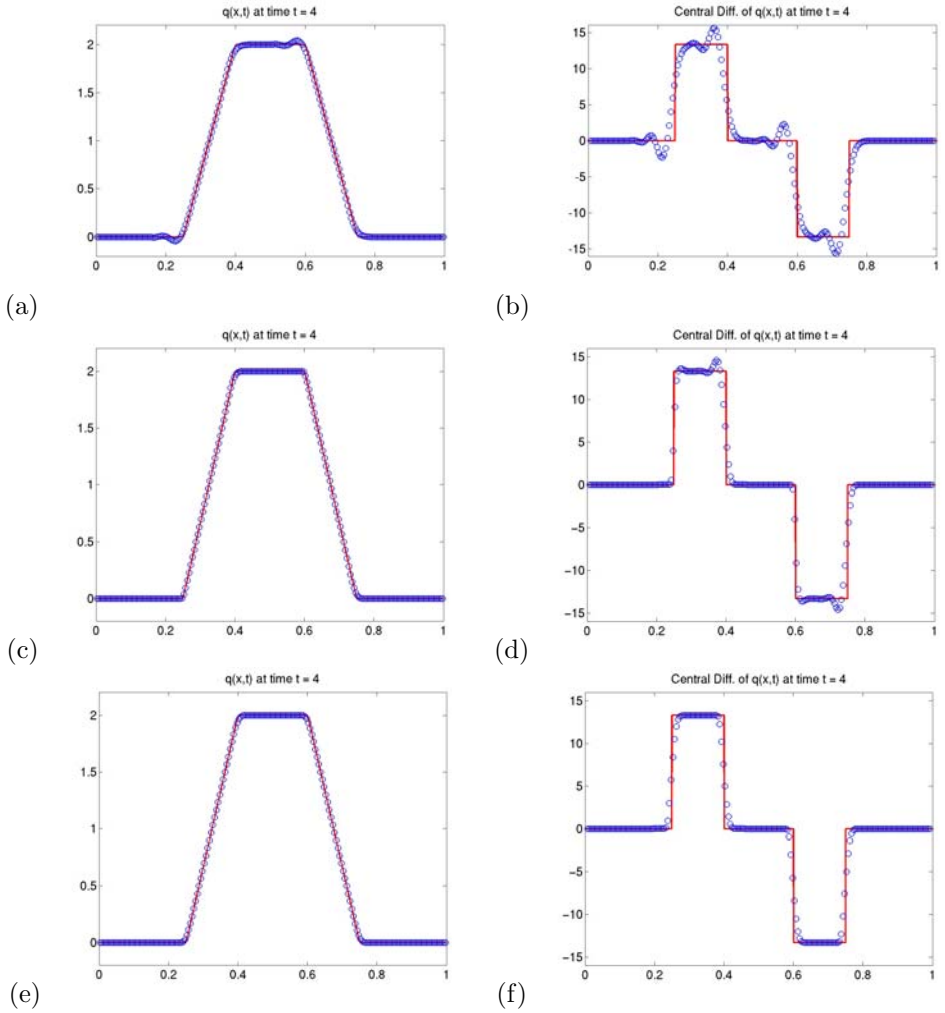


FIG. 5.1. *The solution to the* 1*D advection equation and its derivative on a periodic domain after four revolutions; the derivative is computed using a centered finite difference. In plots* (a) *and* (b) *no limiters are used (Lax–Wendroff method). In plots* (c) *and* (d) *the monotonized centered (MC) limiter is applied to the waves, while in plots* (e) *and* (f) *the MC limiter is applied to wave differences.*

Because the boundary conditions are periodic the solution returns to its initial state whenever t is an integer. The quantity that we would like to compute is the derivative of q, which we approximate using centered differences:

$$\frac{dq}{dx}(x_i, t^n) \approx \frac{Q^n_{i+1} - Q^n_{i-1}}{2\Delta x}. \tag{5.22}$$

In Figure 5.1 we plot $q(x,t)$ and the central difference of $q(x,t)$ after four full revolutions on a grid with 200 points using (a)–(b) no limiters, (c)–(d) the standard wave limiting, and (e)–(f) the new wave-difference limiting. As expected, the wave limiting approach does not limit the oscillations in the x-derivative of $q(x,t)$, while the wave-difference limiting approach produces a sharp and oscillation-free profile. Both limiting strategies produce sharp and oscillation-free $q(x,t)$ profiles.

5.5. Conservation of magnetic field. It is clear that the method we are proposing conserves mass, momentum, and energy, since these quantities are updated via flux differencing. In order to have confidence that our method produces correct jump conditions across shock waves, we should also verify that the magnetic field is conserved (as it would be in the base scheme). In order to show this we note that the total amount of $B_1(x,y,t)$ in the computational domain at time $t=t^n$ is given by

$$\begin{aligned} T_1^n &= \Delta x \Delta y \sum_{i=1}^{N}\sum_{j=1}^{M} [B_1]_{ij}^n = \frac{\Delta x}{2}\sum_{i=1}^{N}\sum_{j=1}^{M}\left(\mathbb{A}_{i,j+1}^n - \mathbb{A}_{i,j-1}^n\right) \\ &= \frac{\Delta x}{2}\sum_{i=1}^{N} \mathbb{A}_{i,M+1}^n + \mathbb{A}_{i,M}^n - \mathbb{A}_{i,1}^n - \mathbb{A}_{i,0}^n \\ &\equiv \Delta x \sum_{i=1}^{N} \mathbb{A}_{i,M+1/2}^n - \Delta x \sum_{i=1}^{N} \mathbb{A}_{i,1/2}^n, \end{aligned} \tag{5.23}$$

where N and M are the number of computational grid cells in the x- and y-directions, respectively. Similarly, the total amount of $B_2(x,y,t)$ in the computational domain at time $t=t^n$ is given by

$$T_2^n = \Delta x \Delta y \sum_{j=1}^{M}\sum_{i=1}^{N} [B_2]_{ij}^n = \cdots \equiv \Delta y \sum_{j=1}^{M} \mathbb{A}_{1/2,j}^n - \Delta y \sum_{j=1}^{M} \mathbb{A}_{N+1/2,j}^n. \tag{5.24}$$

Therefore, the only way to modify the total amount of B_1 and B_2 in the computational domain is through loss/gain across the boundary; B_1 and B_2 are numerically conserved. In other words, the central differencing of the magnetic potential that is used to obtain the magnetic field acts to produce an update for the magnetic field that is in flux differencing form.

5.6. Boundary conditions for the magnetic potential. In the examples we present in section 7, we will use two types of boundary conditions: (1) extrapolation to model inflow/outflow and (2) periodic boundary conditions. In both cases we add to the computational domain two layers of ghost cells, which are used to construct numerical fluxes on the computational boundaries. For conserved variables, it will suffice in our examples to use zeroth order extrapolation (i.e., simply copying solution values from the outermost computational cells into the neighboring ghost cells). In order to specify a magnetic potential that is consistent with a zeroth order extrapolated magnetic field, we must in fact apply linear extrapolation to the magnetic potential; for example, on the left boundary we take

$$\mathbb{A}_{0,j} = 2\mathbb{A}_{1,j} - \mathbb{A}_{2,j} \quad \text{and} \quad \mathbb{A}_{-1,j} = 3\mathbb{A}_{1,j} - 2\mathbb{A}_{2,j},$$

where $j=-1,0,1,2,\ldots,M,M+1,M+2$ and M is the number of regular grid cells in the x-direction. This is simply due to the fact that a constant potential means a zero strength magnetic field, while a linear potential means a constant strength magnetic field.

Periodic boundary conditions on the conserved variable can again be handled by copying the appropriate computational grid values into the ghost cells; for example, on the left boundary we take

$$Q_{0,j} = Q_{N,j} \quad \text{and} \quad Q_{-1,j} = Q_{N-1,j},$$

where $j = -1, 0, 1, 2, \ldots, M, M+1, M+2$ and N and M are the number of regular grid points in the x- and y-directions, respectively. If the magnetic field is periodic, then the potential will in general be comprised of a linear part corresponding to the mean magnetic field plus a periodic part corresponding to the zero-mean part of the magnetic field:

$$\mathbb{A}(x, y) = \langle B_1 \rangle y - \langle B_2 \rangle x + \bar{\mathbb{A}}(x, y),$$

where $\vec{B} = (\langle B_1 \rangle, \langle B_2 \rangle)$ is the mean magnetic field. For periodic boundary conditions, the linear part is linearly extrapolated, while the periodic part is handled using the same boundary conditions as the conserved variables.

6. Analysis of the Lorentz force. In this section we take a closer look at the Lorentz force in both the base scheme and the newly proposed MPACT scheme. The idea here is to be able to quantify the improvements that the cell-centered CT method provides over those of just the base scheme. The analysis presented here is based on the work of Tóth [39]; a similar analysis was carried out by Londrillo and Del Zanna [33] for their scheme.

In the conservative form of the ideal MHD equations (1.1), the Lorentz force shows up as part of the momentum flux:

$$\vec{F} \equiv \vec{\nabla} \cdot \left(\frac{1}{2} |\vec{B}|^2 \mathcal{I} - \vec{B} \otimes \vec{B} \right) \equiv \vec{\nabla} \cdot \overleftrightarrow{T}. \tag{6.1}$$

If the magnetic field satisfies the divergence-free condition (1.2), then one can easily show that the projection of the Lorentz force along the magnetic field must identically vanish for all space and time:

$$\vec{F} \cdot \vec{B} \equiv 0. \tag{6.2}$$

Numerical discretizations of the ideal MHD equations in general only satisfy constraint (6.2) to the truncation error of the method. Across shocks this can mean that $\vec{F} \cdot \vec{B}$ is very large, which in turn can lead to the production of unphysical oscillations and the eventual failure of the method.

6.1. Estimating $\vec{F} \cdot \vec{B}$. We now try to quantify the magnitude of the projection of the Lorentz force along the magnetic field in the context of the methods discussed in this work. The wave propagation method can be rewritten in standard finite volume form:

$$Q_{ij}^{n+1} = Q_{ij}^n - \frac{\Delta t}{\Delta x} \left(F^1_{i+1/2,j} - F^1_{i-1/2,j} \right) - \frac{\Delta t}{\Delta y} \left(F^2_{i,j+1/2} - F^2_{i,j-1/2} \right), \tag{6.3}$$

where

$$F^1_{i-1/2,j} = \frac{1}{2} \left(f^1(Q^n_{ij}) + f^1(Q^n_{i-1,j}) \right) + \Phi^1_{i-1/2,j}, \tag{6.4}$$

$$F^2_{i,j-1/2} = \frac{1}{2} \left(f^2(Q^n_{ij}) + f^2(Q^n_{i,j-1}) \right) + \Phi^2_{i,j-1/2}. \tag{6.5}$$

In fact, the above formula is not restricted to the wave propagation method; for example, the corrections could have the following form (see LeVeque [30]):

$$\text{Godunov:} \quad \Phi^1_{i-1/2,j} \equiv \frac{1}{2} \left(\mathcal{A}_1^- \Delta Q_{i-1/2,j} - \mathcal{A}_1^+ \Delta Q_{i-1/2,j} \right),$$

$$\text{Lax–Friedrichs:} \quad \Phi^1_{i-1/2,j} \equiv \frac{1}{2} \frac{\Delta x}{\Delta t} \left(Q^n_{ij} - Q^n_{i-1,j} \right).$$

Independent of the choice of these corrections, the discrete projection of the Lorentz force along the magnetic field can be written as

$$\begin{aligned}\left[\vec{F}\cdot\vec{B}\right]_{ij} \equiv\ & [B_1]_{ij}\left(\frac{T^{11}_{i+1/2,j}-T^{11}_{i-1/2,j}}{\Delta x}+\frac{T^{12}_{i,j+1/2}-T^{12}_{i,j-1/2}}{\Delta y}\right)\\ &+\ [B_2]_{ij}\left(\frac{T^{21}_{i+1/2,j}-T^{21}_{i-1/2,j}}{\Delta x}+\frac{T^{22}_{i,j+1/2}-T^{22}_{i,j-1/2}}{\Delta y}\right).\end{aligned}$$

If we use in the above formula the full numerical construction of the fluxes, T^{ab}, according to formulas (6.4) and (6.5), the analysis of the Lorentz force becomes rather complicated. Instead, we note that in order to compute the dominant contributions in the discrete form of $\vec{F}\cdot\vec{B}$, using only the first terms in the numerical fluxes suffices and the corrections Φ may be omitted:

$$T^{ab}_{i-1/2,j} \approx \frac{1}{2}\left(T^{ab}(\vec{B}^n_{ij})+T^{ab}(\vec{B}^n_{i-1,j})\right), \tag{6.6}$$

$$T^{ab}_{i,j-1/2} \approx \frac{1}{2}\left(T^{ab}(\vec{B}^n_{ij})+T^{ab}(\vec{B}^n_{i,j-1})\right), \tag{6.7}$$

where

$$\overleftrightarrow{T} = \begin{bmatrix}\frac{1}{2}\left(B_2^2-B_1^2\right) & -B_1B_2\\ -B_1B_2 & \frac{1}{2}\left(B_1^2-B_2^2\right)\end{bmatrix}. \tag{6.8}$$

The omission of Φ is justified, because for first order schemes Φ acts to add numerical diffusion to numerical flux that would otherwise be unstable (see, for example, Chapter 4 of [30]). Therefore, the dominant contribution to $\vec{F}\cdot\vec{B}$ will come from the "uncorrected" fluxes. Furthermore, we are concerned only with first order schemes since we would like to understand the behavior of $\vec{F}\cdot\vec{B}$ across shocks.

Having made this simplification, we are now ready to compute a discrete form of $\vec{F}\cdot\vec{B}$. Following Tóth [39], we define the following quantities:

$$\begin{aligned}[\mu_x U]_{ij} &\equiv \frac{1}{2}\left(U_{i+1,j}+U_{i-1,j}\right),\\ [\mu_y U]_{ij} &\equiv \frac{1}{2}\left(U_{i,j+1}+U_{i,j-1}\right),\\ [\delta_x U]_{ij} &\equiv U_{i+1,j}-U_{i-1,j},\\ [\delta_y U]_{ij} &\equiv U_{i,j+1}-U_{i,j-1}.\end{aligned} \tag{6.9}$$

Using these definitions along with the simplified definitions (6.6)–(6.8), one can show that

$$\begin{aligned}\vec{F}\cdot\vec{B} \approx & -\left(B_1\mu_xB_1+B_2\mu_yB_2\right)\left[\frac{\delta_xB_1}{2\Delta x}+\frac{\delta_yB_2}{2\Delta y}\right]\\ &+B_2\left(\mu_yB_2-\mu_xB_2\right)\frac{\delta_xB_1}{2\Delta x}+B_1\left(\mu_xB_1-\mu_yB_1\right)\frac{\delta_yB_2}{2\Delta y}\\ &+\left(B_1\mu_xB_2-B_2\mu_xB_1\right)\frac{\delta_xB_2}{2\Delta x}+\left(B_2\mu_yB_1-B_1\mu_yB_2\right)\frac{\delta_yB_1}{2\Delta y},\end{aligned} \tag{6.10}$$

where we have dropped all ij subscripts. We note that this is the same expression that Tóth [39] derived, except that our expression has an extra factor of $1/2$ in each term and our definitions (6.9) are fully cell-centered versions of those given by Tóth.

TABLE 6.1

Numerical evaluation of the projection of the Lorentz force along the magnetic field. This data shows that for the base scheme $\vec{F}\cdot\vec{B}$ is relatively large and $\mathcal{O}\left(\Delta x^{-1}, \Delta y^{-1}\right)$, while for the MPACT *scheme this quantity is much smaller and $\mathcal{O}(1)$.*

	Base		MPACT	
$\Delta x, \Delta y$	$\|\vec{F}\cdot\vec{B}\|_1$	$\|\vec{\nabla}\cdot\vec{B}\|_1$	$\|\vec{F}\cdot\vec{B}\|_1$	$\|\vec{\nabla}\cdot\vec{B}\|_1$
0.020	0.2152e+00	0.2722e+00	0.3989e−01	0.2820e−14
0.010	0.2925e+00	0.3504e+00	0.3870e−01	0.1110e−14
0.006	0.3937e+00	0.4047e+00	0.3996e−01	0.8498e−14
0.005	0.5695e+00	0.6575e+00	0.3744e−01	0.0000e+00
0.004	0.9613e+00	0.1298e+01	0.3977e−01	0.1272e−13

Across a shock, we can estimate $\vec{F}\cdot\vec{B}$ as follows:

$$\vec{F}\cdot\vec{B} \sim \mathcal{O}(1)\left[\frac{\delta_x B_1}{2\Delta x} + \frac{\delta_y B_2}{2\Delta y}\right] + \mathcal{O}(\Delta x, \Delta y)\cdot\left\{\frac{\delta_x B_1}{2\Delta x}, \frac{\delta_x B_2}{2\Delta x}, \frac{\delta_y B_1}{2\Delta y}, \frac{\delta_y B_2}{2\Delta y}\right\}.$$

Furthermore, finite difference approximations to the derivatives of $\vec{B}$ across a shock will in general be of the order $\mathcal{O}\left(\Delta x^{-1}, \Delta y^{-1}\right)$. This implies that we can further estimate $\vec{F}\cdot\vec{B}$ as follows:

$$\vec{F}\cdot\vec{B} \sim \begin{cases} \mathcal{O}(1) & \text{if} \quad \frac{\delta_x B_1}{2\Delta x} + \frac{\delta_y B_2}{2\Delta y} = 0, \\ \mathcal{O}\left(\Delta x^{-1}, \Delta y^{-1}\right) & \text{otherwise.} \end{cases} \tag{6.11}$$

Therefore, in both the base scheme and the MPACT scheme, $\vec{F}\cdot\vec{B} \neq 0$. However, in the MPACT scheme the projection of the Lorentz force is always bounded independent of the grid resolution, while in the base scheme this term blows up like $1/(\Delta x, \Delta y)$. For a fixed resolution one would expect the MPACT scheme to have a much smaller $\vec{F}\cdot\vec{B}$ error, since the first term in (6.10) vanishes.

Finally, we would like to point out that Tóth [39] introduced a scheme for which $\vec{F}\cdot\vec{B} \equiv 0$ on the discrete level. The main difficulty with that scheme is that it is fully second order and has no mechanism for flux limiting. Although the MPACT scheme presented in this work only satisfies $\vec{F}\cdot\vec{B} \sim \mathcal{O}(1)$ for discontinuous data, we believe that for many problems this is good enough. As evidence for this we note that for standard 1D high-resolution schemes, which are known to work well for 1D ideal MHD, expression (6.10) also does not identically vanish:

$$\vec{F}\cdot\vec{B} \approx B_1\left(\mu_x B_2 - B_2\right)\frac{\delta_x B_2}{2\Delta x}.$$

For discontinuous data this expression is again $\mathcal{O}(1)$.

6.2. Numerical example. In order to numerically verify the above statements, we consider the 30° rotated Riemann problem that is described in detail in section 7.2. At $t = 0.3$ and $y = \Delta y/2$ (the $y =$ constant slice nearest the center), the 1-norm of the cell-centered divergence and the 1-norm of $\vec{F}\cdot\vec{B}$ as defined in (6.10) are computed for various values of $\Delta x = \Delta y$. Both the base scheme and the MPACT scheme are tested. The results of this computation are shown in Table 6.1; the numerical calculations validate the conclusions of formula (6.11).

7. MHD examples. In this section the MPACT method is applied to six separate 2D test problems for the ideal MHD equations. The first example is used to test

TABLE 7.1
Convergence study on a smooth Alfvén wave problem. Note that the errors shown here are truncated to three significant digits for the sake of brevity; however, the R_N values are computed with the full errors.

	Base		MPACT	
N	δ_N	R_N	δ_N	R_N
16	5.58e−2	–	1.48e−1	–
32	1.41e−2	1.98	3.80e−2	1.96
64	3.71e−3	1.93	9.60e−3	1.99
128	9.36e−4	1.99	2.34e−3	2.00

the order of accuracy, the second examines the shock-capturing ability, and the third examines the numerical dissipation of the MPACT method. The final three examples test the MPACT scheme on fully 2D problems. Unless otherwise noted, all computations are run with a CFL number of 0.95, the MC limiter (see [30]), and a gas constant of $\gamma = 5/3$.

7.1. Smooth Alfvén wave problem. We begin by numerically verifying the order of accuracy of the base scheme and the MPACT scheme on the smooth Alfvén problem proposed in [38]. This problem consists of a circularly polarized Alfvén wave that propagates at angle $\alpha = \tan^{-1}(0.5) \approx 25.5651^\circ$ relative to the horizontal axis. The computational domain is $(x, y) \in [0, 1/\cos\alpha] \times [0, 1/\sin\alpha]$ with double-periodic boundary conditions. The initial Alfvén wave propagates at a constant speed towards the origin and returns to its initial state whenever t is an integer.

Each method is run on grids of size $N \times 2N$, where $N = 16, 32, 64$, and 128. These grid resolutions are chosen so that $\Delta x/\Delta y = 1$. All of these simulations are run out to $t = 2$. As in [38], for each value of N we compute a relative error, δ_N, and an approximation to the order of accuracy, R_N, using the following definitions:

$$\begin{aligned} \delta_N &= \frac{1}{4}(E_N(u_\perp) + E_N(u_3) + E_N(B_\perp) + E_N(B_3)), \\ E_N(v) &= \frac{\sum_{i=1}^{N}\sum_{j=1}^{2N} |v_{ij} - v_{\text{exact}}|}{\sum_{i=1}^{N}\sum_{j=1}^{2N} |v_{\text{exact}}|}, \\ R_N &= \log_2\left(\delta_{N/2}/\delta_N\right). \end{aligned}$$

The results of these simulations are summarized in Table 7.1. Both methods are second order accurate; the MPACT scheme is slightly less accurate due to the additional step in computing the magnetic field.

7.2. Rotated Riemann problem. Next we test the constrained transport methods on a problem with discontinuities. In particular, we consider a 1D shock tube problem that has been rotated by an angle of $\alpha = 30^\circ$. The computational domain is $(x, y) \in [-1.2, 1.2] \times [-1, 1]$ with extrapolation boundary conditions on all conserved variables on all sides. On the top and bottom boundaries we don't extrapolate in the direction normal to the boundary (as is customary); instead, we extrapolate in the direction parallel to the shock interface. The initial data is

$$\left[\rho, u_\perp, u_{||}, u_3, p, B_\perp, B_{||}, B_3\right] = \begin{cases} \left[1, -0.4, 0, 0, 1, 0.75, 1, 0\right] & \text{if} \quad \xi < 0, \\ \left[0.2, -0.4, 0, 0, 0.1, 0.75, -1, 0\right] & \text{if} \quad \xi > 0, \end{cases}$$

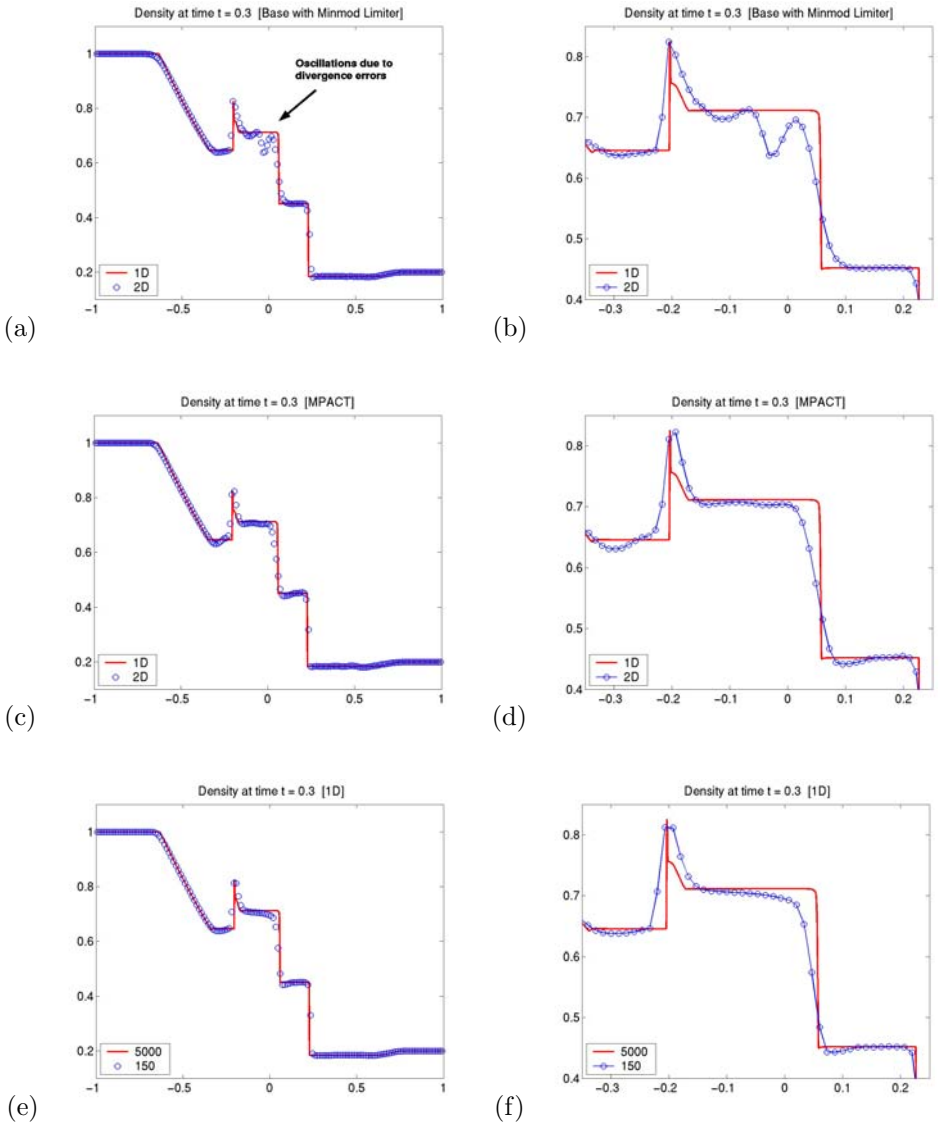


FIG. 7.1. *Solution to a* 30° *rotated Riemann problem. The solutions are obtained using* (a)–(b) *the base scheme,* (c)–(d) *the* MPACT *scheme, and* (e)–(f) *the* 1*D wave propagation method. The solid line in each plot is a highly resolved (*5000 *points) solution of the* 1*D MHD equations. For each method we plot both the entire solution from* −1 *to* 1 *as well as a blowup of the solution near the origin.*

where $\xi = x\cos\alpha + y\sin\alpha$, $u_\perp$ is the component of the velocity perpendicular to the shock interface, and $u_{||}$ is the component of velocity parallel to the shock interface.

The results on a 180×150 grid are shown in Figures 7.1 and 7.2. In these figures we plot the density at $j = 75$, which corresponds to $y = -0.066\bar{6}$. Also shown in all of these plots is a highly resolved (5000 grid points) solution of the 1D MHD equations. We see in panels (a)–(b) of Figure 7.1 that the base scheme produces oscillations because it fails to satisfy a discrete divergence-free condition. In order to be sure that these oscillations are due to divergence errors and not inadequate

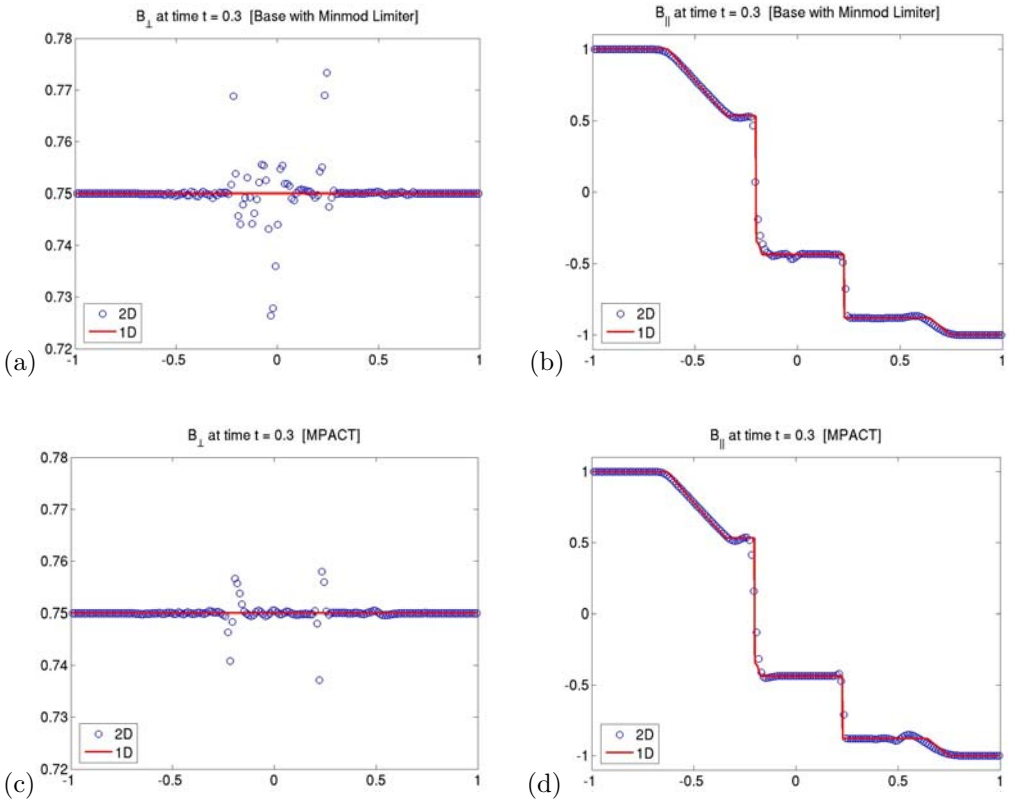


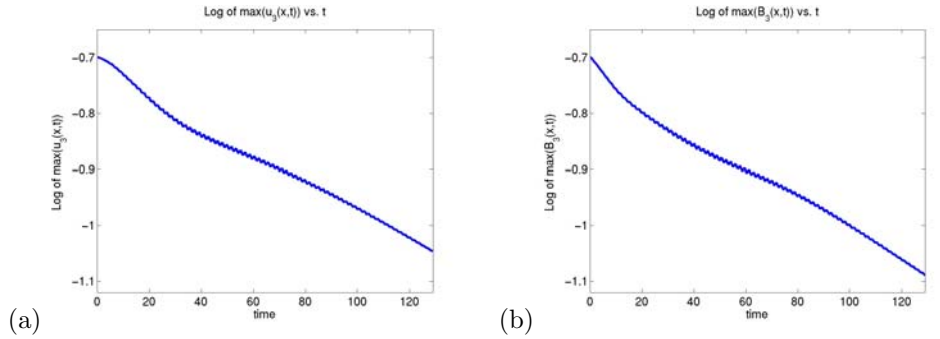
FIG. 7.2. *Magnetic field solutions to a* 30° *rotated Riemann problem (the corresponding density plots are shown in Figure* 7.1*). The solutions are obtained using* (a)–(b) *the base scheme and* (c)–(d) *the* MPACT *scheme. In these plots we show the components of the magnetic field perpendicular* ($B_\perp$) *and parallel* ($B_{||}$) *to the shock interface.*

limiting, the diffusive minmod limiter was used. Very similar results, which are not shown here, were also obtained with the base scheme using other standard limiters. As shown in panels (c)–(d), the MPACT scheme produces much cleaner results. In the MPACT scheme we have used the MC limiter for all conserved variables and the van Leer limiter for the magnetic potential. In this same figure we also show in (e)–(f) a truly 1D calculation with the same resolution as the other two schemes and the MC limiter; we note that both the 1D approach and the MPACT scheme produce very similar results. We note that the largest unphysical oscillations occur near the origin (where the initial discontinuity is placed); this is due to the fact that the ideal MHD equations produce the following "evolution" equation for the divergence of the magnetic field:

$$\frac{\partial}{\partial t}\left(\vec{\nabla}\cdot\vec{B}\right) = 0.$$

Finally, in Figure 7.2 we show the corresponding magnetic field components for this rotated Riemann problem. In these plots it is again evident that the base scheme produces larger errors near the origin than the MPACT scheme.

7.3. Numerical dissipation and long-term decay of Alfvén waves. Next we consider a problem proposed by Balsara [5], which tries to quantify the amount of



(a) (b)

FIG. 7.3. *Log-linear plot of the maximum value of the z-components of* (a) *velocity and* (b) *magnetic fields. In the exact solution the maximum value should remain constant, but numerical dissipation causes the amplitude to decay. The numerical dissipation produced by the* MPACT *scheme on the linearly degenerate Alfvén waves is comparable to several of the schemes tested by Balsara* [5].

numerical dissipation introduced by a numerical scheme for ideal MHD. Specifically, the test problem measures the decay of the amplitude of a linearly degenerate Alfvén wave that propagates at a shallow angle to the grid ($\tan(1/6) \approx 9.462°$). See Balsara [5] for a description of the initial conditions.

The MPACT scheme was run on the computational domain $[-3,3] \times [-3,3]$ on a 120×120 grid until time $t = 129$. The maximum values of u_3 and B_3 should remain constant in time for the exact solution, but decay due to numerical dissipation. These quantities are plotted versus time in Figure 7.3. The numerical dissipation produced by the MPACT scheme on the linearly degenerate Alfvén waves is comparable to several of the schemes tested by Balsara [5].

7.4. Rotor problem. The first truly multidimensional problem we attempt is the rotor problem considered in both [6] and [38]. The computational domain is $(x, y) \in [0, 1] \times [0, 1]$ with zeroth order extrapolation boundary conditions on all four sides. We make use of the MC limiter with a CFL number of 0.85.

The results of a computation using the MPACT scheme on a 256×256 grid are shown in Figure 7.4. We plot contour lines of the (a) density, (b) thermal pressure, (c) Mach number, and (d) magnetic pressure. These results are in good agreement with the computations of Balsara and Spicer [6] and Tóth [38].

7.5. Orszag–Tang vortex. Next we consider a problem that appears several times in the literature, including [10, 38, 43]. The computational domain in this case is $(x, y) \in [0, 2\pi] \times [0, 2\pi]$ with double-periodic boundary conditions. In this problem we again use the MC limiter, but with a reduced CFL number of 0.75. This slightly reduced CFL number is required because in the solution shock waves propagate along the boundary; however, in CLAWPACK [31] the computed CFL number does not include the tangential wave speed directly on the boundaries.

A Schlieren plot of the thermal pressure at time $t = 3.14$ predicted by the MPACT scheme on a 256×256 grid is shown in Figure 7.5(a). A slice of this solution at $y = 1.9321$ is shown in Figure 7.5(b). Although each paper displays the solution at slightly different times and resolutions, our results compare well with those given in [10, 38, 43].

Let us also take this opportunity to show what can go wrong if no CT is used. With a grid resolution of 256×256, the wave propagation method without CT (i.e., the base scheme) produces a negative pressure at time $t = 0.633$. The solution at

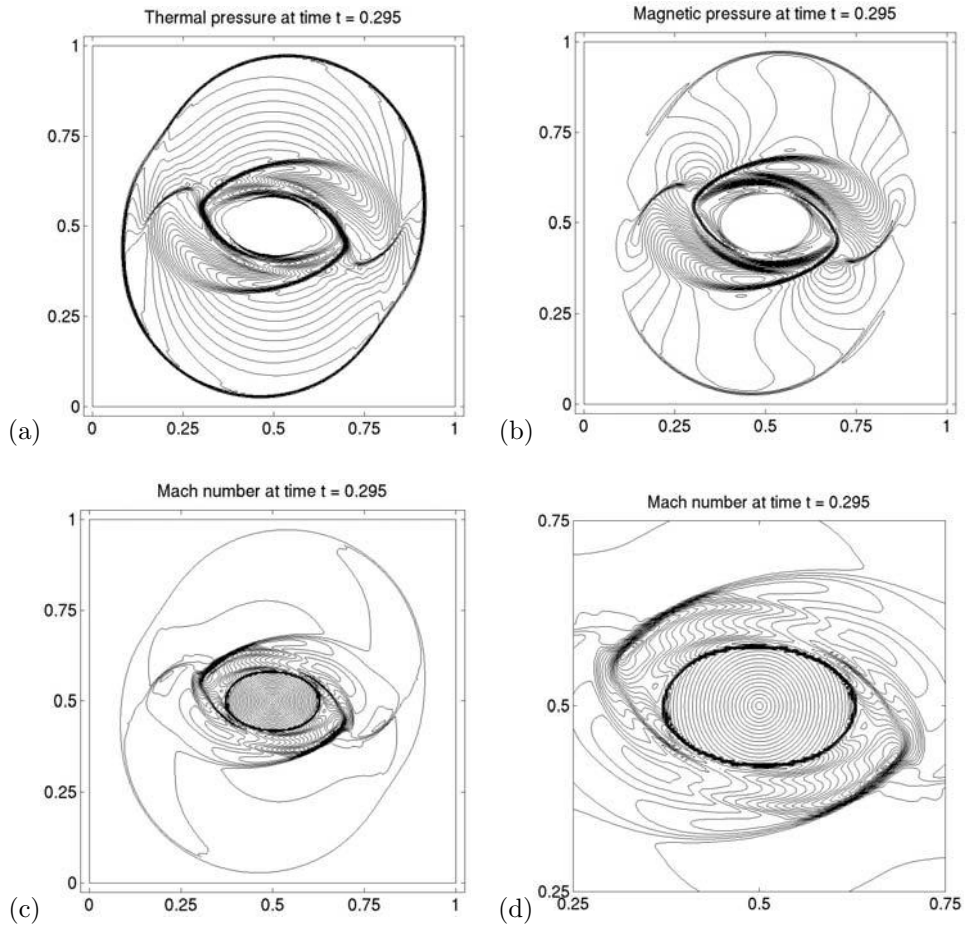


FIG. 7.4. *The* (a) *thermal pressure,* (b) *magnetic pressure,* (c) *Mach number, and* (d) *close-up view of the Mach number at* $t = 0.295$ *for the rotor problem. The solution was obtained with the* MPACT *scheme on a* 256×256 *grid. In following with Tóth* [38]*,* 30 *contour lines are shown for each graph in the ranges* $0.007 < p < 0.776$*,* $0 < |\vec{u}|/c_s < 3.64$*, and* $0.007 < |\vec{B}|^2/2 < 0.702$*.*

this time is shown in Figure 7.6 along with the CT solution at the same time. One clearly sees from these plots that keeping the divergence small is crucial for numerical stability. Note that we are using the same limiters, CFL numbers, etc., in these two calculations.

7.6. Cloud-shock interaction. Finally, we consider a slight modification of the cloud-shock interaction problem from Tóth [38] and Dai and Woodward [10]. In their formulation, the cloud is propagating at a constant speed toward the shock, while in our setup the bubble is initially stationary. The computational domain is $(x, y) \in [0, 1] \times [0, 1]$ with constant inflow conditions at the left boundary and outflow boundary conditions on the three remaining sides. In this problem we use the more diffusive minmod limiter [30] and a slightly reduced CFL number of 0.75. Here we have to use a slightly reduced CFL number in order to avoid producing negative pressures when the shock and cloud first come into contact. The initial conditions

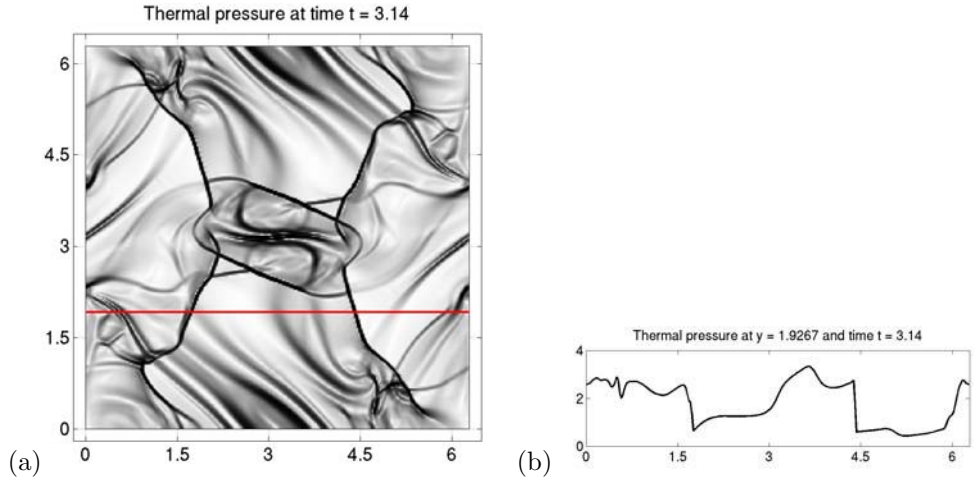


FIG. 7.5. *The thermal pressure at* $t = 3.14$ *for the Orszag–Tang vortex. Shown are* (a) *a Schlieren plot over the entire domain and* (b) *a horizontal slice at* $y = 1.9267$. *The solution was obtained with the* MPACT *scheme on a* 256×256 *grid.*

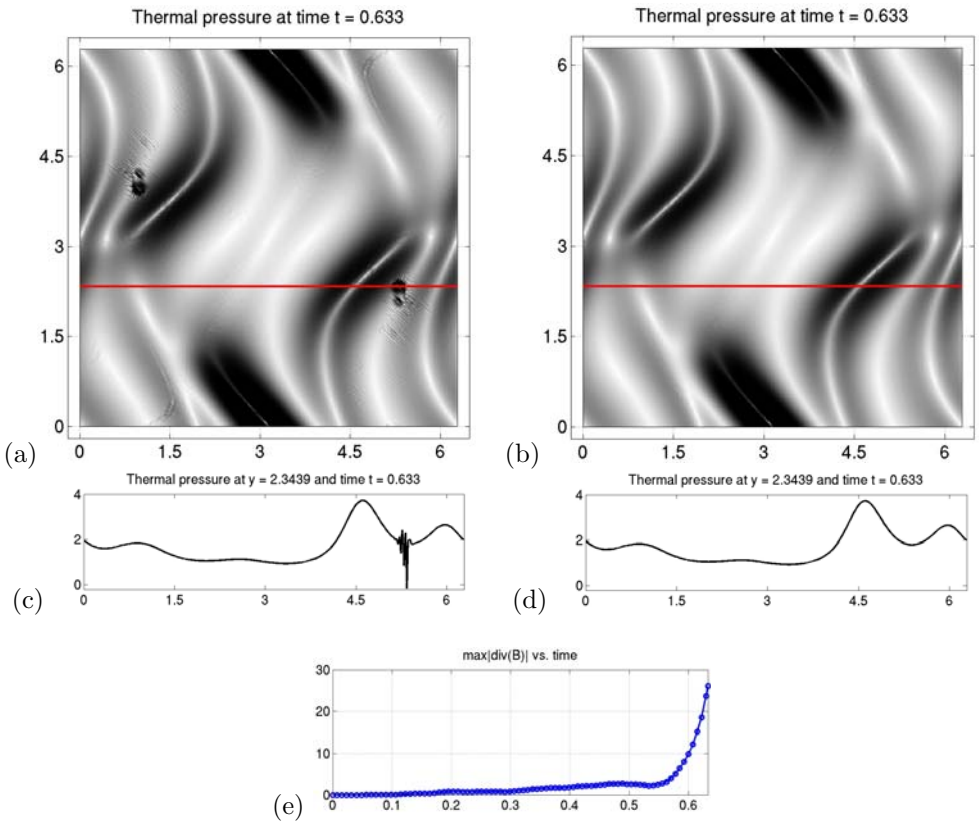


FIG. 7.6. *The thermal pressure at* $t = 0.633$ *for the Orszag–Tang vortex on a* 256×256 *grid. Plots* (a) *and* (c) *show the solution computed with the wave propagation method without CT (i.e., the base scheme). Plots* (b) *and* (d) *show the solution computed with the proposed CT method. Shown in* (e) *is* $\max |\vec{\nabla} \cdot \vec{B}|$ *as a function of time for the base scheme calculation—note the sharp increase in the divergence near* $t = 0.633$. *These plots demonstrate how unphysical oscillations can appear in calculations that do not explicitly control the divergence of the magnetic field.*

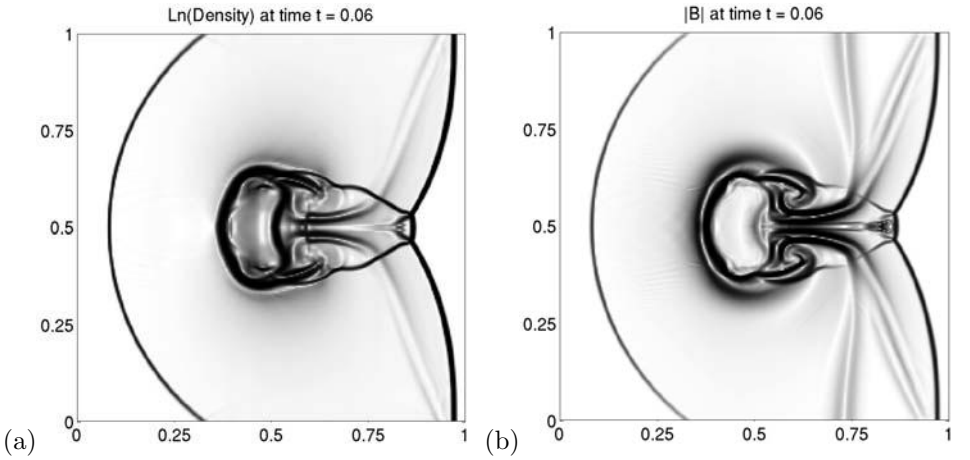


FIG. 7.7. *Schlieren plots of the* (a) *natural log of the density and* (b) *magnetic field at* $t = 0.06$ *for the cloud-shock interaction problem. The solution was obtained with the* MPACT *scheme on a* 256×256 *grid.*

consist of a shock located at $x = 0.05$,

$$\Big[\rho, u_1, u_2, u_3, p, B_1, B_2, B_3\Big]_{t=0} = \begin{cases} \Big[3.86859, 11.2536, 0, 0, 167.345, 0, 2.1826182, -2.1826182\Big] & \text{if } x < 0.05, \\ \Big[1, 0, 0, 0, 1, 0, 0.56418958, 0.56418958\Big] & \text{if } x > 0.05, \end{cases}$$

and a circular cloud of density $\rho = 10$ and radius $r = 0.15$ centered at $(x, y) = (0.25, 0.5)$. The cloud is in hydrostatic equilibrium with the fluid to the right of the shock. The initial magnetic potential is given by

$$\mathbb{A}(x, y, 0) = \begin{cases} -2.1826182x + 0.080921431 & \text{if} \quad x \leq 0.05, \\ -0.56418958x & \text{if} \quad x \geq 0.05. \end{cases}$$

The results of a computation using the MPACT scheme on a 256×256 grid are shown in Figure 7.7. We show Schlieren plots of the (a) log of the density and (b) magnetic field lines. Our results compare well with the solutions presented in [38]. With this test problem we are able to demonstrate the robustness of the MPACT scheme on a violent shock-cloud interaction.

8. Conclusions. We have introduced a new CT method for solving the ideal MHD equations. This method is comprised of four steps which must be carried out during each computational time step:

1. Solve the ideal MHD equations in conservative form using a high-resolution finite volume method.
2. Compute an intermediate time velocity field through linear interpolation in time.
3. Solve the magnetic potential equation.
4. Compute a divergence-free magnetic field from the magnetic potential using centered finite differences.

In this paper, the ideal MHD equations in Step 1 are solved using the wave propagation of LeVeque [27] with normal and transverse Riemann solvers based on the 8-wave decomposition described by Powell [34]. The magnetic potential equation in Step 3 is solved using a modified wave propagation method that ensures that smooth magnetic potentials are computed to second order accuracy. The centered finite differences in Step 4 of this method can introduce spurious oscillations in the magnetic field even if standard TVD limiters are applied in the magnetic potential update. To overcome this difficulty, a novel limiting strategy based on applying TVD limiters to [illegible] was introduced. The resulting MPACT scheme has all of the following properties:

1. Smooth solutions are approximated to second order accuracy in space and time.
2. The method is numerically conservative.
3. The magnetic field satisfies a discrete divergence-free condition.
4. Shock waves and other discontinuities are accurately captured.

This approach was successfully tested on several 2D problems.

The numerical method proposed in this work has been implemented as an extension to the CLAWPACK software package [31] and can be freely downloaded from the Web.

Appendix. Proof of second order accuracy for the modified WP method.

CLAIM. [illegible] $\Delta t = c_1 \Delta x = c_2 \Delta y$ [illegible] c_1 [illegible] c_2 [illegible] $q(x, y, t)$ [illegible] (x, y, t) [illegible] 5.3 [illegible] $\mathcal{O}(\Delta t^2)$ [illegible]. Under the assumption that $q(x, y, t)$ is smooth in x, y, and t, the quantity $q(x, y, t + \Delta t)$ can be expanded in a Taylor series in the following way:

$$
\begin{aligned}
q(x, y, t + \Delta t) &= q + \Delta t\, q_t + \frac{\Delta t^2}{2}\, q_{tt} + \mathcal{O}(\Delta t^3) \\
&= q - \Delta t\, u_1\, q_x + \frac{\Delta t^2}{2} \left[u_1 \left(u_1\, q_x + u_2\, q_y \right)_x \right] \\
&\quad - \Delta t\, u_2\, q_y + \frac{\Delta t^2}{2} \left[u_2 \left(u_2\, q_y + u_1\, q_x \right)_y \right] + \mathcal{O}(\Delta t^3)\,,
\end{aligned}
\tag{A.1}
$$

where $q = q(x, y, t)$, $u_1 \equiv u_1(x, y, t + \Delta t/2)$, and $u_2 \equiv u_2(x, y, t + \Delta t/2)$. In the above expression, derivatives in time were replaced by derivatives in space through the use of (5.3). Showing that the modified wave propagation method is $\mathcal{O}(\Delta t^2)$ accurate becomes equivalent to showing that each term in (A.1) is approximated to $\mathcal{O}(\Delta t^2)$. Furthermore, through symmetry this is equivalent to showing that the terms

$$
q^\star = q - \Delta t\, u_1\, q_x + \frac{\Delta t^2}{2} \left[u_1 \left(u_1\, q_x + u_2\, q_y \right)_x \right] \tag{A.2}
$$

are approximated to $\mathcal{O}(\Delta t^2)$ by the part of the algorithm that is responsible for constructing fluctuations and fluxes in the x-direction.

The part of the algorithm that is responsible for modeling the terms in expression (A.2) can be written as

$$
Q_i^\star = Q_i^n - \frac{\Delta t}{\Delta x} \left[\mathcal{A}_1^- \Delta Q_{i+1/2} + \mathcal{A}_1^+ \Delta Q_{i-1/2} \right] - \frac{\Delta t}{\Delta x} \left[\tilde{F}^{1-}_{i+1/2} - \tilde{F}^{1+}_{i-1/2} \right]. \tag{A.3}
$$

We now plug in definitions (5.6), (5.7), (5.11), and (5.12) to obtain that

$$\begin{aligned} Q_i^\star = Q_i^n &- \frac{1}{2}\frac{\Delta t}{\Delta x}\left[\left(2\left[s_i\right]^- + |s_i|\right)\mathcal{W}_{i+1/2} + \left(2\left[s_i\right]^+ - |s_i|\right)\mathcal{W}_{i-1/2}\right] \\ &+ \frac{1}{2}\frac{\Delta t^2}{\Delta x^2}\left[u_1\right]_i\left[s_{i+1/2}\mathcal{W}_{i+1/2} - s_{i-1/2}\mathcal{W}_{i-1/2}\right] - \frac{\Delta t}{\Delta x}\left[\tilde{\tilde{F}}^{1-}_{i+1/2} - \tilde{\tilde{F}}^{1+}_{i-1/2}\right], \end{aligned} \tag{A.4}$$

where the $\tilde{\tilde{F}}$'s contain the transverse corrections. We now make use of the identities

$$\left[s_i\right]^+ + \left[s_i\right]^- = s_i, \tag{A.5}$$

$$\left[s_i\right]^+ - \left[s_i\right]^- = |s_i|, \tag{A.6}$$

$$s_{i+1/2}\mathcal{W}_{i+1/2} = \left[u_1\right]_{i+1/2}\left(Q_{i+1} - Q_i\right) \tag{A.7}$$

and reduce (A.4) to

$$\begin{aligned} Q_i^\star = Q_i^n &- \frac{1}{2}\frac{\Delta t}{\Delta x}s_i\left(\mathcal{W}_{i+1/2} + \mathcal{W}_{i-1/2}\right) \\ &+ \frac{1}{2}\frac{\Delta t^2}{\Delta x^2}\left[u_1\right]_i\left(\left[u_1\right]_{i+1/2}\left(Q_{i+1} - Q_i\right) - \left[u_1\right]_{i-1/2}\left(Q_i - Q_{i-1}\right)\right) \\ &- \frac{\Delta t}{\Delta x}\left[\tilde{\tilde{F}}^{1-}_{i+1/2} - \tilde{\tilde{F}}^{1+}_{i-1/2}\right], \end{aligned} \tag{A.8}$$

$$\begin{aligned} \implies Q_i^\star = Q_i^n &- \frac{1}{2}\frac{\Delta t}{\Delta x}\left[u_1\right]_i\left(Q_{i+1} - Q_{i-1}\right) \\ &+ \frac{1}{2}\frac{\Delta t^2}{\Delta x^2}\left[u_1\right]_i\left(\left[u_1\right]_{i+1/2}\left(Q_{i+1} - Q_i\right) - \left[u_1\right]_{i-1/2}\left(Q_i - Q_{i-1}\right)\right) \\ &- \frac{\Delta t}{\Delta x}\left[\tilde{\tilde{F}}^{1-}_{i+1/2} - \tilde{\tilde{F}}^{1+}_{i-1/2}\right] + \mathcal{O}\left(\Delta t^3\right), \end{aligned} \tag{A.9}$$

$$\begin{aligned} \implies Q_i^\star = Q_i^n &- \Delta t\left[u_1 q_x\right]_i + \frac{\Delta t^2}{2}\left[u_1\left(u_1 q_x\right)_x\right]_i \\ &- \frac{\Delta t}{\Delta x}\left[\tilde{\tilde{F}}^{1-}_{i+1/2} - \tilde{\tilde{F}}^{1+}_{i-1/2}\right] + \mathcal{O}\left(\Delta t^3\right). \end{aligned} \tag{A.10}$$

Finally, we turn our attention to the transverse corrections. After collecting contributions from all the necessary grid cells, the transverse flux correction can be written as

$$\begin{aligned} \tilde{\tilde{F}}^{1-}_{i+1/2} = -\frac{1}{2}\frac{\Delta t}{\Delta y}\Bigg(&\left[u_1\right]_i^+\left(u_2^+\Delta Q - 2\tilde{F}^{2+}\right)_{i,j-1/2} \\ &+ \left[u_1\right]_i^-\left(u_2^+\Delta Q - 2\tilde{F}^{2+}\right)_{i+1,j-1/2} \\ &+ \left[u_1\right]_i^+\left(u_2^-\Delta Q + 2\tilde{F}^{2-}\right)_{i,j+1/2} \\ &+ \left[u_1\right]_i^-\left(u_2^-\Delta Q + 2\tilde{F}^{2-}\right)_{i+1,j+1/2}\Bigg). \end{aligned} \tag{A.11}$$

We note that

$$\begin{aligned} u_2^+\Delta Q - 2\tilde{F}^{2+} &= u_2^-\Delta Q + \mathcal{O}(\Delta t^2), \\ u_2^-\Delta Q + 2\tilde{F}^{2-} &= u_2^+\Delta Q + \mathcal{O}(\Delta t^2) \end{aligned}$$

through the application of identities (A.5) and (A.6). Using this result in conjunction with a Taylor expansion, the terms in expression (A.11) can be rewritten as follows:

$$\begin{aligned}
\left(u_2^+ \Delta Q - 2\tilde{F}^{2+}\right)_{i,j-1/2} &= u_2^- \Delta Q_{i+1/2,j} + \mathcal{O}(\Delta t^2), \\
\left(u_2^+ \Delta Q - 2\tilde{F}^{2+}\right)_{i+1,j-1/2} &= u_2^- \Delta Q_{i+1/2,j} + \mathcal{O}(\Delta t^2), \\
\left(u_2^- \Delta Q + 2\tilde{F}^{2-}\right)_{i,j+1/2} &= u_2^+ \Delta Q_{i+1/2,j} + \mathcal{O}(\Delta t^2), \\
\left(u_2^- \Delta Q + 2\tilde{F}^{2-}\right)_{i+1,j+1/2} &= u_2^+ \Delta Q_{i+1/2,j} + \mathcal{O}(\Delta t^2)\,.
\end{aligned}$$

Note that the error here is $\mathcal{O}(\Delta t^2)$ because the Taylor expansions are accurate to $\mathcal{O}(\Delta t)$ and $u_2^{\pm}\Delta Q \mp 2\tilde{F}^{2\pm} = \mathcal{O}(\Delta t)$. Using these results, the transverse flux correction (A.11) can be rewritten as

$$\begin{aligned}
\tilde{\tilde{F}}^{1-}_{i+1/2} &= -\frac{1}{2}\frac{\Delta t}{\Delta y}\left([u_1]_i^- + [u_1]_i^+\right)\left(u_2^- \Delta Q_{i+1/2,j} + u_2^+ \Delta Q_{i+1/2,j}\right) + \mathcal{O}(\Delta t^2) \\
&= -\frac{1}{2}\frac{\Delta t}{\Delta y}\,[u_1]_i\, u_2 \Delta Q_{i+1/2,j} + \mathcal{O}(\Delta t^2)\,.
\end{aligned}$$

Similarly,

$$\tilde{\tilde{F}}^{1+}_{i-1/2} = -\frac{1}{2}\frac{\Delta t}{\Delta y}\,[u_1]_i\, u_2 \Delta Q_{i-1/2,j} + \mathcal{O}(\Delta t^2)\,.$$

Plugging these results into (A.10) yields

$$\begin{aligned}
Q_i^\star &= Q_i^n - \Delta t\,[u_1 q_x]_i + \frac{\Delta t^2}{2}\,[u_1\,(u_1 q_x)_x]_i \\
&\quad + \frac{1}{2}\frac{\Delta t^2}{\Delta x \Delta y}\,[u_1]_i\left(u_2 \Delta Q_{i+1/2,j} - u_2 \Delta Q_{i-1/2,j}\right) + \mathcal{O}\left(\Delta t^3\right) \\
&= Q_i^n - \Delta t\,[u_1 q_x]_i + \frac{\Delta t^2}{2}\left[u_1\,(u_1 q_x + u_2 q_y)_x\right]_i + \mathcal{O}\left(\Delta t^3\right)\,.
\end{aligned} \tag{A.12}$$

Comparing expressions (A.12) and (A.1) shows that the 1-step error of the modified wave propagation method is $\mathcal{O}\left(\Delta t^3\right)$, and therefore the local truncation error is $\mathcal{O}\left(\Delta t^2\right)$. ∎

Acknowledgment. The author would like to thank the anonymous reviewers for their valuable comments.

REFERENCES

[1] N. Aslan, *A visual fluctuation splitting scheme for magnetohydrodynamics with a new sonic fix and Euler limit*, J. Comput. Phys., 197 (2004), pp. 1–27.

[2] J. Balbas, E. Tadmor, and C.C. Wu, *Non-oscillatory central schemes for one- and two-dimensional MHD equations*, J. Comput. Phys., 201 (2004), pp. 261–285.

[3] D.S. Bale, *Wave Propagation Algorithms on Curved Manifolds with Applications to Relativistic Hydrodynamics*, Ph.D. thesis, University of Washington, Seattle, WA, 2002.

[4] D.S. Bale, R.J. LeVeque, S. Mitran, and J.A. Rossmanith, *A wave propagation method for conservation laws and balance laws with spatially varying flux functions*, SIAM J. Sci. Comput., 24 (2002), pp. 955–978.

[5] D.S. Balsara, *Second-order-accurate schemes for magnetohydrodynamics with divergence-free reconstruction*, Astrophys. J. Suppl., 151 (2004), pp. 149–184.
[6] D.S. Balsara and D. Spicer, *A staggered mesh algorithm using high order Godunov fluxes to ensure solenoidal magnetic fields in magnetohydrodynamic simulations*, J. Comput. Phys., 149 (1999), pp. 270–292.
[7] J.M. Bardeen and L.T. Buchman, *Numerical tests of evolution systems, gauge conditions, and boundary conditions for* $1D$ *colliding gravitational plane waves*, Phys. Rev. D, 65 (2002).
[8] J.U. Brackbill and D.C. Barnes, *The effect of nonzero* $\nabla \cdot b$ *on the numerical solution of the magnetohydrodynamic equations*, J. Comput. Phys., 35 (1980), pp. 426–430.
[9] P. Cargo and G. Callice, *Roe matrices for ideal MHD and systematic construction of Roe matrices for systems of conservation laws*, J. Comput. Phys., 136 (1997), pp. 446–466.
[10] W. Dai and P.R. Woodward, *A simple finite difference scheme for multidimensional magnetohydrodynamic equations*, J. Comput. Phys., 142 (1998), pp. 331–369.
[11] A. Dedner, F. Kemm, D. Kröner, C.-D. Munz, T. Schnitzer, and M. Wesenberg, *Hyperbolic divergence cleaning for the MHD equations*, J. Comput. Phys., 175 (2002), pp. 645–673.
[12] H. De Sterck, *Multi-dimensional upwind constrained transport on unstructured grids for "shallow water" magnetohydrodynamics*, in Proceedings of the 15th AIAA Computational Fluid Dynamics Conference, AIAA 2001-2623, Anaheim, CA, 2001.
[13] C. Evans and J.F. Hawley, *Simulation of magnetohydrodynamic flow: A constrained transport method*, Astrophys. J., 332 (1988), pp. 659–677.
[14] M. Fey and M. Torrilhon, *A constrained transport upwind scheme for divergence-free advection*, in Hyperbolic Problems: Theory, Numerics, and Applications, T.Y. Hou and E. Tadmor, eds., Springer, Berlin, 2003, pp. 529–538.
[15] T.R. Fogarty, *High-Resolution Finite Volume Methods for Acoustics in a Rapidly-Varying Heterogeneous Medium*, Master's thesis, University of Washington, Seattle, WA, 1997.
[16] T.R. Fogarty, *Finite Volume Methods for Acoustics and Elasto-plasticity with Damage in a Heterogeneous Medium*, Ph.D. thesis, University of Washington, Seattle, WA, 2002.
[17] T.R. Fogarty and R.J. LeVeque, *High-resolution finite volume methods for acoustics in periodic and random media*, J. Acoust. Soc. Amer., 106 (1999), pp. 1–12.
[18] S.K. Godunov, *Symmetric form of the magnetohydrodynamic equations*, Numer. Methods Mech. Continuum Medium, 1 (1972), pp. 26–34.
[19] T.I. Gombosi, *Physics of the Space Environment*, Cambridge University Press, Cambridge, UK, 1998.
[20] A. Harten, *High resolution schemes for hyperbolic conservation laws*, J. Comput. Phys., 49 (1983), pp. 357–393.
[21] A. Harten and J.M. Hyman, *Self-adjusting grid methods for one-dimensional hyperbolic conservation laws*, J. Comput. Phys., 50 (1983), pp. 235–269.
[22] C. Helzel, *Numerical Approximation of Conservation Laws with Stiff Source Terms for the Modelling of Detonation Waves*, Ph.D. thesis, Otto-von-Guericke-Universität Magdeburg, Magdeburg, Germany, 2000.
[23] C. Helzel, R.J. LeVeque, and G. Warnecke, *A modified fractional step method for the accurate approximation of detonation waves*, SIAM J. Sci. Comput., 22 (2000), pp. 1489–1510.
[24] J.D. Jackson, *Classical Electrodynamics*, 3rd ed., Wiley, New York, 1999.
[25] L. Lee, *An Immersed Interface Method for Incompressible Navier–Stokes Equations*, Ph.D. thesis, University of Washington, Seattle, WA, 2002.
[26] R.J. LeVeque, *High-resolution conservative algorithms for advection in incompressible flow*, SIAM J. Numer. Anal., 33 (1996), pp. 627–665.
[27] R.J. LeVeque, *Wave propagation algorithms for multi-dimensional hyperbolic systems*, J. Comput. Phys., 131 (1997), pp. 327–335.
[28] R.J. LeVeque, *Balancing source terms and flux gradients in high-resolution Godunov methods: The quasi-steady wave-propagation algorithm*, J. Comput. Phys., 146 (1998), pp. 346–365.
[29] R.J. LeVeque, *Finite volume methods for nonlinear elasticity in heterogeneous media*, Internat. J. Numer. Methods Fluids, 20 (2002), pp. 93–104.
[30] R.J. LeVeque, *Finite Volume Methods for Hyperbolic Problems*, Cambridge University Press, Cambridge, UK, 2002.
[31] R.J. LeVeque, clawpack *software*, available from http://www.amath.washington.edu/~claw.
[32] P. Londrillo and L. Del Zanna, *High-order upwind schemes for multidimensional magnetohydrodynamics*, Astrophys. J., 530 (2000), pp. 508–524.
[33] P. Londrillo and L. Del Zanna, *On the divergence-free condition in Godunov-type schemes*

for ideal magnetohydrodynamics: The upwind constrained transport method, J. Comput. Phys., 195 (2004), pp. 17–48.

[34] K.G. Powell, *An Approximate Riemann Solver for Magnetohydrodynamics (That Works in More Than One Dimension)*, Tech. Report 94-24, ICASE, Langley, VA, 1994.

[35] K.G. Powell, P.L. Roe, T.J. Linde, T.I. Gombosi, and D.L. De Zeeuw, *A solution-adaptive upwind scheme for ideal magnetohydrodynamics*, J. Comput. Phys., 154 (1999), pp. 284–309.

[36] D.S. Ryu, F. Miniati, T.W. Jones, and A. Frank, *A divergence-free upwind code for multidimensional magnetohydrodynamic flows*, Astrophys. J., 509 (1998), pp. 244–255.

[37] P.K. Sweby, *High resolution schemes using flux limiters for hyperbolic conservation laws*, SIAM J. Numer. Anal., 21 (1984), pp. 995–1011.

[38] G. Tóth, *The $\nabla \cdot B = 0$ constraint in shock-capturing magnetohydrodynamics codes*, J. Comput. Phys., 161 (2000), pp. 605–652.

[39] G. Tóth, *Conservative and orthogonal discretization for the Lorentz force*, J. Comput. Phys., 182 (2002), pp. 346–354.

[40] M. Torrilhon, *Locally divergence-preserving upwind finite volume-schemes for magnetohydrodynamic equations*, SIAM J. Sci. Comput., 26 (2005), pp. 1166–1191.

[41] H.C. Yee and B. Sjögreen, *Non-linear filtering and limiting in high order methods for ideal and non-ideal MHD*, J. Sci. Comput., 27 (2006), pp. 507–521.

[42] K. Yee, *Numerical solutions of initial boundary value problems involving Maxwell's equations in isotropic media*, IEEE Trans. Antennas and Propagation, 14 (1966), pp. 302–307.

[43] A.L. Zachary, A. Malgoli, and P. Colella, *A higher-order Godunov method for multidimensional ideal magnetohydrodynamics*, SIAM J. Sci. Comput., 15 (1994), pp. 263–284.

SIAM J. SCI. COMPUT.
Vol. 28, No. 5, pp. 1798–1811
© 2006 Society for Industrial and Applied Mathematics

A RATIONAL SPECTRAL COLLOCATION METHOD WITH ADAPTIVELY TRANSFORMED CHEBYSHEV GRID POINTS*

T. W. TEE† AND LLOYD N. TREFETHEN†

Abstract. A spectral collocation method based on rational interpolants and adaptive grid points is presented. The rational interpolants approximate analytic functions with exponential accuracy by using prescribed barycentric weights and transformed Chebyshev points. The locations of the grid points are adapted to singularities of the underlying solution, and the locations of these singularities are approximated by the locations of poles of Chebyshev–Padé approximants. Numerical experiments on two time-dependent problems, one with finite time blow-up and one with a moving front, indicate that the method far outperforms the standard Chebyshev spectral collocation method for problems whose solutions have singularities in the complex plane close to $[-1, 1]$.

Key words. rational spectral collocation method, adaptive grid points, barycentric interpolation formula, complex singularities, Chebyshev–Padé approximation, blow-up problem, the Burgers equation

AMS subject classifications. 65M70, 65M50, 41A20, 41A21

DOI. 10.1137/050641296

1. Introduction.

The Chebyshev spectral method (the phrase spectral method is synonymous with spectral collocation method in this article) is a method for the numerical solution of differential equations on a bounded nonperiodic interval, which may be assumed without loss of generality to be $[-1, 1]$. The basis of the method is to approximate the solution of a differential equation by a polynomial which interpolates data u_k at Chebyshev points $x_k = \cos(k\pi/N)$, $k = 0, 1, \ldots, N$, where the data $\{u_k\}$ are determined by requiring that the polynomial interpolant satisfy the differential equation exactly at the points $\{x_k\}$. As N increases, the approximation error decays at a rate which depends on the smoothness of the underlying solution. If the solution can be continued as an analytic function in a closed ellipse with foci ± 1, semimajor axis length L, and semiminor axis length l, then the error decays exponentially at the rate $O((L+l)^{-N})$ [29]. However, if the solution has singularities in the complex plane close to $[-1, 1]$, so that $L + l \approx 1$, then this convergence rate will be too slow for the method to be effective.

One cannot simply alter the points and expect a better convergence rate, because polynomials which interpolate at Chebyshev points already give near best polynomial approximations. Instead, one must consider spectral methods based on other types of global interpolants, such as rational functions. Unlike polynomials, rational functions can interpolate at any set of points without the drawback of the Runge phenomenon.

The purpose of this article is to present a new spectral method based on rational interpolants and adaptive grid points, and to show that this method far outperforms the Chebyshev spectral method for problems whose solutions have singularities in the complex plane close to $[-1, 1]$.

Spectral methods based on rational interpolants have been developed by Berrut, Baltensperger, and Mittelmann [12]. A common feature of their methods is the rep-

*Received by the editors September 27, 2005; accepted for publication (in revised form) May 30, 2006; published electronically October 24, 2006.
http://www.siam.org/journals/sisc/28-5/64129.html
†Oxford University Computing Laboratory, Wolfson Building, Parks Road, Oxford OX1 3QD, United Kingdom (wynn.tee@comlab.ox.ac.uk, LNT@comlab.ox.ac.uk).

resentation of rational interpolants in barycentric form. The barycentric form of a rational function r which interpolates data $f_0, f_1, \ldots, f_N$ at points $x_0, x_1, \ldots, x_N$ is

$$r(x) = \frac{\displaystyle\sum_{k=0}^{N} \frac{w_k}{x - x_k} f_k}{\displaystyle\sum_{k=0}^{N} \frac{w_k}{x - x_k}}, \tag{1.1}$$

where $w_0, w_1, \ldots, w_N$ are called barycentric weights. An advantage of representing a rational interpolant in barycentric form is that its derivatives can be evaluated easily using differentiation formulae derived by Baltensperger and Berrut [1, 2], instead of using the quotient rule repeatedly. The pth derivative of r evaluated at x_j can be written in the form $r^{(p)}(x_j) = \sum_{k=0}^{N} D_{jk}^{(p)} u_k$, where the entries of $D^{(1)}$ and $D^{(2)}$, the first and second order differentiation matrices, are given by

$$D_{jk}^{(1)} = \begin{cases} \dfrac{w_k}{w_j(x_j - x_k)} & \text{if } j \neq k, \\ -\displaystyle\sum_{i=0,\ i\neq k}^{N} D_{ji}^{(1)} & \text{if } j = k, \end{cases} \tag{1.2}$$

$$D_{jk}^{(2)} = \begin{cases} 2D_{jk}^{(1)} \left(D_{jj}^{(1)} - \dfrac{1}{x_j - x_k} \right) & \text{if } j \neq k, \\ -\displaystyle\sum_{i=0,\ i\neq k}^{N} D_{ji}^{(2)} & \text{if } j = k \end{cases} \tag{1.3}$$

for $j, k = 0, 1, \ldots, N$. For every set of points $\{x_k\}$, there is a unique set of barycentric weights $\{w_k\}$, up to a multiplicative constant, for which (1.1) represents a polynomial interpolant. For example, if $\{x_k\}$ are the Chebyshev points, then (1.1) represents a polynomial interpolant if and only if $w_0 = c/2$, $w_k = (-1)^k c$, $k = 1, \ldots, N-1$, and $w_N = (-1)^N c/2$ for some nonzero constant c [17]. Thus,

Baltensperger, Berrut, and Noël exploited this property, modifying a polynomial which interpolates at Chebyshev points into a rational interpolant which approximates analytic functions with exponential accuracy, by applying a conformal map to the points and leaving the barycentric weights unchanged [4]. Baltensperger and Berrut developed a spectral method based on that rational interpolant, in which they used the conformal map of Kosloff and Tal-Ezer [24] to distribute grid points more evenly over the computational interval [3, 11]. Berrut had shown in earlier work that if that rational interpolant has arbitrary points, then it cannot have poles in the interpolation interval, which is a crucial property for approximating continuous functions [9].

Berrut and Mittelmann took the alternative approach, modifying a polynomial which interpolates at Chebyshev points into a rational interpolant with preassigned poles, by changing the barycentric weights using a formula discovered by Berrut [10] and leaving the points unchanged [13]. They developed a spectral method based on that rational interpolant, in which they computed the locations of preassigned poles by minimizing a residual error iteratively [14, 15].

Berrut and Mittelmann also developed a spectral method based on a combination of the two previous rational interpolants, in which they used a modified version of the conformal map of Bayliss and Turkel [8] to adapt grid points to multiple fronts, and computed the parameters of the map and the locations of poles simultaneously by minimizing a residual error iteratively [16]. However, this computationally expensive optimization step renders their method inefficient for time-dependent problems.

Spectral methods based on adaptive grid points have, so far, required that the underlying problem be transformed into new coordinates, where the transformation maps predefined grid points in a transformed coordinate into irregular grid points in a physical coordinate. The aim is to transform a problem which has a rapidly varying solution into one which has a slowly varying solution. The transformation is usually achieved by means of a parametrized map whose parameters are chosen to minimize an error functional [6, 7, 22]. A related method is that of an adaptive finite difference method coupled with a spectral method, in which a transformation serving the same purpose as that just described is constructed from the grid points of an adaptive finite difference solution [26, 27].

In this article, we present a new adaptive rational spectral method which, unlike existing adaptive spectral methods, does not require that the underlying problem be transformed into new coordinates, and unlike existing rational spectral methods, takes into account and locates a priori unknown singularities of the underlying solution. In section 2, we explain how a conformal map which maps the singularities of a function farther into the complex plane can be used to improve the Chebyshev spectral method. In section 3, we explain how Chebyshev–Padé approximation can be used to approximate the locations of these singularities. In section 4, we explain how the new method can be implemented from ideas discussed in sections 2 and 3. In section 5, we present the results of numerical experiments on two time-dependent problems. Finally in section 6, we present some conclusions.

2. Enlarging the ellipse of analyticity. The basis of our adaptive rational spectral method is the following theorem, which was proved in [4], on the exponential convergence of rational functions which interpolate at transformed Chebyshev points.

THEOREM 1. D_1 D_2 $\mathbb{C}$ $J = [-1, 1]$ I $g : D_1 \to D_2$ $g(J) = I$ E_ρ ± 1 ρ $f : D_2 \to \mathbb{C}$ $f \circ g : D_1 \to \mathbb{C}$ E_ρ g $E_\sigma \supseteq E_\rho$

$$r_N(x) = \frac{\displaystyle\sum_{k=0}^{N}{}' \frac{(-1)^k}{x - x_k} f(x_k)}{\displaystyle\sum_{k=0}^{N}{}' \frac{(-1)^k}{x - x_k}}, \qquad x_k = g(\cos(k\pi/N)), \tag{2.1}$$

$k = 0$ $k = N$

$$|r_N(x) - f(x)| = O((L + l)^{-N}) \tag{2.2}$$

$x \in [-1, 1]$

If the solution u of a differential equation can be continued as an analytic function in a neighborhood of $[-1, 1]$ in the complex plane, then the error in approximating u by the Chebyshev spectral method will decay exponentially at a rate which depends

on the size of the largest ellipse with foci ± 1 in which u is analytic. To be brief, this largest ellipse will be called the [illegible] of u.

Theorem 1 suggests that one should choose a conformal map g for which the ellipse of analyticity of $u \circ g$ is larger than the ellipse of analyticity of u, and apply g in a spectral method based on rational interpolants of the form (2.1), to obtain an approximation of u which is more accurate than that obtained using the Chebyshev spectral method with the same number of grid points. Alternatively, one can obtain an equally accurate approximation of u by using the Chebyshev spectral method to approximate $u \circ g$ directly, but this approach requires the tedious transformation of the underlying differential equation into new coordinates. To be precise, if x is the physical coordinate and $y = g^{-1}(x)$ is the transformed coordinate, where g^{-1} is the inverse of g, then the first few derivatives of u have to be transformed according to

$$\frac{du}{dx} = \frac{1}{g'(y)} \frac{du}{dy}, \tag{2.3}$$

$$\frac{d^2u}{dx^2} = \frac{1}{[g'(y)]^2} \frac{d^2u}{dy^2} - \frac{g''(y)}{[g'(y)]^3} \frac{du}{dy}, \tag{2.4}$$

$$\frac{d^3u}{dx^3} = \frac{1}{[g'(y)]^3} \frac{d^3u}{dy^3} - \frac{3g''(y)}{[g'(y)]^4} \frac{d^2u}{dy^2} - \frac{g'''(y)g'(y) - 3[g''(y)]^2}{[g'(y)]^5} \frac{du}{dy}, \tag{2.5}$$

and similarly for higher derivatives. These transformations are not required when one approximates u by a spectral method based on rational interpolants of the form (2.1).

How do we choose g? If only real solutions are of interest, then we should choose g such that g^{-1} maps the two complex conjugate singularities lying on the ellipse of analyticity of u farther away from $[-1, 1]$. There are many possibilities, but for the time being, consider the case where u is analytic everywhere in the complex plane except along two lines extending from $0 \pm \varepsilon \mathrm{i}$ to $0 \pm \infty \mathrm{i}$, as shown in the region I_1 of Figure 1. Then

1. $h_1(z) = -z\varepsilon^{-1}\mathrm{i}$ rotates and scales I_1 to produce I_2,
2. $h_2(z) = \sin^{-1}(z)$ maps I_2 to the vertical strip I_3, and
3. $h_3(z) = -z/\sin^{-1}(\varepsilon^{-1}\mathrm{i})$ rotates and scales I_3 to produce I_4.

The conformal map g which maps I_4 to I_1 is the inverse of $h_3 \circ h_2 \circ h_1$ and due to the identity $\sin^{-1}(z) = -\mathrm{i}\sinh^{-1}(\mathrm{i}z)$, can be written in the form

$$g(z) = \varepsilon \sinh(z \sinh^{-1}(\tfrac{1}{\varepsilon})). \tag{2.6}$$

Figure 1 shows the adaptive nature of g, in that it maps points which are clustered near the boundaries of $[-1, 1]$ into points which are clustered near the singular lines of u. Table 1 shows that as ε decreases geometrically, the number of grid points required for the Chebyshev spectral method to approximate u to 10 digits of accuracy increases geometrically, whereas for a spectral method based on rational interpolants of the form (2.1) and the conformal map (2.6), this number increases only algebraically.

The derivation of (2.6) can be generalized to cases where the two singular lines begin at $\delta \pm \varepsilon \mathrm{i}$ instead of $0 \pm \varepsilon \mathrm{i}$. The conformal map g becomes

$$g(z) = \delta + \varepsilon \sinh\left[\left(\sinh^{-1}(\tfrac{1-\delta}{\varepsilon}) + \sinh^{-1}(\tfrac{1+\delta}{\varepsilon})\right)\tfrac{z-1}{2} + \sinh^{-1}(\tfrac{1-\delta}{\varepsilon})\right]. \tag{2.7}$$

The derivation of (2.7) makes no assumptions about the type of singularities of u, so (2.7) is applicable for all types of singularities, such as poles and branch points, as long as they are confined to the two singular lines.

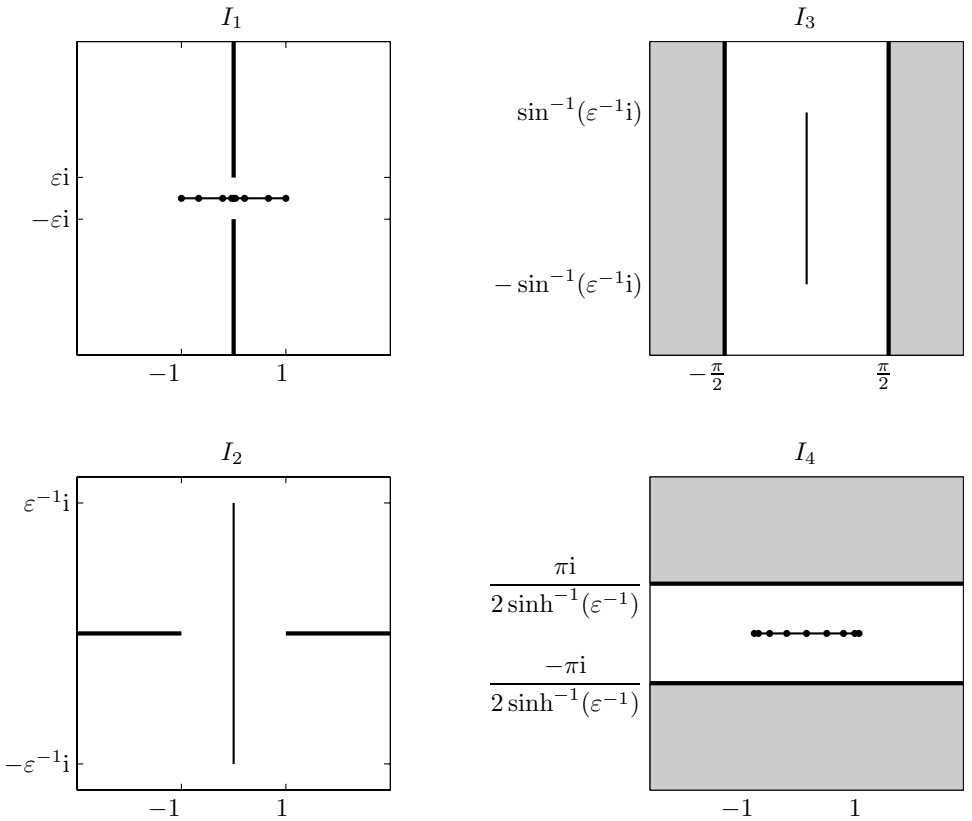


FIG. 1. *Sequence of transformation images to be read from top to bottom, then from left to right. In the first panel, the thick lines represent singular lines, the thin line represents* $[-1, 1]$*, and the white background represents the region of analyticity* I_1*. In the fourth panel, the dots represent Chebyshev points.*

TABLE 1

Comparison of theoretical convergence rates for the Chebyshev spectral method and the adaptive rational spectral method. If a convergence rate is $O(K^{-N})$*, then* N_{min} *is the smallest integer* N *for which* K^{-N} *is less than* 10^{-10}*.*

ε	Chebyshev spectral method		Adaptive rational spectral method	
	Convergence rate	$N_{\min}$	Convergence rate	$N_{\min}$
1	$O(2.4142^{-N})$	27	$O(3.8258^{-N})$	18
0.1	$O(1.1050^{-N})$	231	$O(1.6528^{-N})$	46
0.01	$O(1.0100^{-N})$	2303	$O(1.3395^{-N})$	79
0.001	$O(1.0010^{-N})$	23026	$O(1.2277^{-N})$	113
0.0001	$O(1.0001^{-N})$	230259	$O(1.1711^{-N})$	146

3. Locating the singularities. The parameters δ and ε required in (2.7) are usually not known analytically, so it is necessary to approximate them using some singularity location technique. The technique must be computationally cheap for our adaptive rational spectral method to be more efficient than existing rational spectral methods. The technique which we chose is based on Chebyshev–Padé approximation. It is conceptually similar to the technique presented by Weideman in [30], where the locations of singularities are approximated by the locations of poles of Fourier–Padé approximants.

If f is Lipschitz continuous on $[-1,1]$, then it has a Chebyshev expansion

$$f(x) = \sum_{k=0}^{\infty} a_k T_k(x), \tag{3.1}$$

where $a_k = \frac{2}{\pi}\int_{-1}^{1}(1-x^2)^{-\frac{1}{2}} f(x)\,T_k(x)\,dx$ (a_0 is half the value given by this formula) and $T_k(x) = \cos(k\cos^{-1}(x))$. The scalars $\{a_k\}$ are called Chebyshev coefficients and the functions $\{T_k\}$ are the Chebyshev polynomials. The linear $[m,n]$ Chebyshev–Padé approximation of f is defined as the rational function

$$r_{m,n}(x) \equiv \frac{p(x)}{q(x)} = \frac{\displaystyle\sum_{k=0}^{m} b_k T_k(x)}{1 + \displaystyle\sum_{k=1}^{n} c_k T_k(x)} \tag{3.2}$$

of numerator degree at most m and denominator degree at most n, such that the Chebyshev expansion of $fq - p$ satisfies

$$f(x)q(x) - p(x) = O(T_{m+n+1}(x)). \tag{3.3}$$

The coefficients $\{b_k\}$ and $\{c_k\}$ can be determined by substituting (3.1) and (3.2) into (3.3), matching terms with Chebyshev polynomials of equal degree, and finally solving the system of linear equations

$$\begin{pmatrix} 1 & & & -d_{0,1} & \cdots & -d_{0,n} \\ & \ddots & & \vdots & \vdots & \vdots \\ & & 1 & -d_{m,1} & \cdots & -d_{m,n} \\ & & & -d_{m+1,1} & \cdots & -d_{m+1,n} \\ & & & \vdots & \vdots & \vdots \\ & & & -d_{m+n,1} & \cdots & -d_{m+n,n} \end{pmatrix} \begin{pmatrix} b_0 \\ \vdots \\ b_m \\ c_1 \\ \vdots \\ c_n \end{pmatrix} = \begin{pmatrix} a_0 \\ \vdots \\ a_m \\ a_{m+1} \\ \vdots \\ a_{m+n} \end{pmatrix}, \tag{3.4}$$

where $\sum_{j=0}^{\infty} d_{j,k}T_j(x) = \sum_{j=0}^{\infty} a_j T_j(x)T_k(x)$ for $\{b_k\}$ and $\{c_k\}$ as required. It should be noted that $\{c_k\}$ can be computed independently of $\{b_k\}$, in which case the system of linear equations to be solved is only of size n. The poles of $r_{m,n}$ can be determined by computing the eigenvalues of the companion matrix [5, 18]

$$C = \begin{pmatrix} 0 & 1 & & & & \\ \frac{1}{2} & 0 & \frac{1}{2} & & & \\ & \ddots & \ddots & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & & & \frac{1}{2} & 0 & \frac{1}{2} \\ \dfrac{c_0}{-2c_n} & \dfrac{c_1}{-2c_n} & \cdots & \dfrac{c_{n-3}}{-2c_n} & \dfrac{c_{n-2}}{-2c_n} + \dfrac{1}{2} & \dfrac{c_{n-1}}{-2c_n} \end{pmatrix}, \tag{3.5}$$

where $c_0 = 1$. This is a matrix eigenvalue problem of size n. It can be deduced from (3.4) that $r_{m,n}$ always exists and is unique provided that $c_n \neq 0$.

How do we determine δ and ε? The aim is to approximate them by the real and imaginary parts of the poles of a Chebyshev–Padé approximant. If we can evaluate

or approximate the solution u of a differential equation at $M+1$ Chebyshev points, typically scaled and translated to a subinterval of $[-1,1]$ rather than $[-1,1]$ itself, then we can compute approximations to the Chebyshev coefficients $a_0,\dots,a_M$ of u by applying the fast Fourier transform to these values of u. The poles of an $[m,2]$ Chebyshev–Padé approximation of u, where m is a small integer which we take to be $M/2$ rounded to the nearest integer, can then be computed by applying the techniques described in the preceding paragraph. The locations of these poles approximate the locations of the two complex conjugate singularities lying on the ellipse of analyticity of u [28], and hence δ and ε. The overall work involves solving a 2×2 system of linear equations and computing the eigenvalues of a 2×2 matrix, both of which are trivial computationally. In practice, the ill-conditioning of (3.4) is not a problem, because we need only know δ and ε to one or two digits of accuracy.

4. Implementation. How do we put together a spectral method based on the ideas discussed in sections 2 and 3? If we can evaluate or approximate the solution u of a differential equation at grid points $x_0^n, x_1^n, \dots, x_N^n$ and time t^n, then we approximate u by the rational interpolant

$$r_N^n(x,t) = \frac{\displaystyle\sum_{k=0}^{N}{}' \frac{(-1)^k}{x-x_k^n}\, r_N^n(x_k^n,t)}{\displaystyle\sum_{k=0}^{N}{}' \frac{(-1)^k}{x-x_k^n}}, \tag{4.1}$$

where $r_N^n(x_k^n,t^n)\approx u(x_k^n,t^n)$. The data $\{r_N^n(x_k^n,t^{n+1})\}$ approximating $\{u(x_k^n,t^{n+1})\}$ can be determined by discretizing the underlying differential equation using a temporal discretization method of choice and requiring that r_N^n satisfy the discretized equation exactly at the grid points $\{x_k^n\}$ and at any intermediate time steps, where the spatial derivatives of r_N^n can be evaluated as the product of differentiation matrices and data vectors, as mentioned in section 1. In MATLAB, the first and second order differentiation matrices can be constructed by calling `bcmatrix`:

```
% BCMATRIX constructs the first and second order differentiation matrices, D1
% and D2, corresponding to the barycentric weights w and grid points x. Note
% that w and x must be column vectors of the same size.
function [D1,D2] = bcmatrix(w,x)
  N = length(x)-1; ii = (1:N+2:(N+1)^2)';
  Dw = repmat(w',N+1,1) ./ repmat(w,1,N+1) - eye(N+1);
  Dx = repmat(x ,1,N+1) - repmat(x',N+1,1) + eye(N+1);
  D1 = Dw ./ Dx;
  D1(ii) = 0; D1(ii) = - sum(D1,2);
  D2 = 2*D1 .* (repmat(D1(ii),1,N+1) - 1./Dx);
  D2(ii) = 0; D2(ii) = - sum(D2,2);
```

There is no difference between what we have done so far and what we would do with the Chebyshev spectral method. The difference arises when we adapt grid points for the next time step, for which we do the following.

ALGORITHM 1. [illegible]

1. [illegible] $r_N^n(x,t^{n+1})$ [illegible] $M+1$ [illegible] $[-1,1]$ [illegible] δ
2. [illegible] $r_N^n(x,t^{n+1})$ [illegible] 1 [illegible]
3. [illegible] δ [illegible] ε [illegible] 2 [illegible]

4. [illegible] $x_k^{n+1} = g(\cos(k\pi/N)) \quad k = 0, 1, \ldots, N$ [illegible] g [illegible] (2.7) [illegible] δ [illegible] ε
5. [illegible] $r_N^n(x, t^{n+1})$ [illegible] $\{x_k^{n+1}\}$ [illegible]

In practice, we start with a grid of Chebyshev points and only begin adapting the grid points when ε falls below a threshold indicating how close the true singularities of u in the complex plane are to $[-1, 1]$. In our numerical experiments, we take this threshold to be 0.5. It is necessary to make a few remarks about steps 1–3.

The number $M+1$ of Chebyshev points in step 1 should be enough for $r_N^n(x, t^{n+1})$ to be approximated to a prescribed accuracy in the subinterval centered at δ. The width of the subinterval should be sufficiently small for M to be small, too, and yet sufficiently large for the real part of the true singularities of $u(x, t^{n+1})$ to lie within it. In our numerical experiments, we take M to be the smallest power of 2 for which the Chebyshev coefficient of largest subindex in step 2 is less than 10^{-6} in absolute value and we take the subinterval to be $[\delta - \zeta, \delta + \zeta]$, where $\zeta = \min(10\varepsilon, 1 - |\delta|)$. A typical value of M is 128. In MATLAB, step 1 (and step 5) can be implemented by calling `bcinterp`:

```
% BCINTERP evaluates the barycentric interpolation formula to find the values
% ff of a rational interpolant with barycentric weights w, grid points x, and
% grid data f, at the points xx. Note that w, x, and f must be column vectors
% of the same size.
function ff = bcinterp(w,x,f,xx)
  [mask,index] = ismember(xx,x);
  invmask = (mask==0);
  xxx = xx(invmask);
  ff = zeros(length(xx),1);
  numer = zeros(length(xxx),1);
  denom = zeros(length(xxx),1);
  for i=1:length(x)
    temp = w(i)./(xxx-x(i));
    numer = numer + temp*f(i);
    denom = denom + temp;
  end
  ff(invmask) = numer./denom;
  ff(mask) = f(index(mask));
```

and step 2 can be implemented by calling `chebpoly`:

```
% CHEBPOLY uses the fast Fourier transform to find M+1 Chebyshev coefficients
% A of a function, given its values U at M+1 Chebyshev points. The operation
% is applied to each column of U.
function A = chebpoly(U)
  M = size(U,1)-1;
  U(M+2:2*M,:) = U(M:-1:2,:);
  A = real(ifft(U)); A = A(1:M+1,:); A(2:M,:) = 2*A(2:M,:);
```

The new ε in step 3 should be multiplied by a safety factor to ensure that (2.7) accounts for the true singularities of $u(x, t^{n+1})$. In our numerical experiments, we take this safety factor to be 0.75. In MATLAB, step 3 can be implemented by calling `chebpade`, which in turn calls `chebeval` and `chebpoly`:

```
% CHEBPADE computes the poles z of an [m,n] Chebyshev-Pade approximation of a
% function with Chebyshev coefficients a. Note that a must be a column vector
% with the coefficients in the order a0, a1, a2, ...
function z = chebpade(m,n,a)
  N = length(a)-1;                          % number of coefficients less one
  E = chebeval([zeros(1,n);eye(N+n,n)]);
```

```
  F = repmat(chebeval([a;zeros(n,1)]),1,n);
  D = -chebpoly(E.*F);
  c = [1;D(m+2:m+n+1,:)\a(m+2:m+n+1)];     % denominator coefficients
  if (n > 1)                               % denominator is at least quadratic
    C = diag(0.5*ones(n-1,1),-1);          % companion matrix
    C = C + diag([1;0.5*ones(n-2,1)],1);
    C(n,1:n) = -0.5*c(1:n)'/c(n+1);
    C(n,n-1) = C(n,n-1) + 0.5;
    z = eig(C);                            % poles
  else                                     % denominator is linear
    z = -c(1)/c(2);                        % poles
  end

% CHEBEVAL uses the fast Fourier transform to find the values U of a function
% at N+1 Chebyshev points, given its N+1 Chebyshev coefficients A. The
% operation is applied to each column of A.
function U = chebeval(A)
  N = size(A,1)-1;
  A(2:N,:) = A(2:N,:)/2; A(N+2:2*N,:) = A(N:-1:2,:);
  U = real(fft(A)); U = U(1:N+1,:);
```

5. Numerical experiments. We now present the results of numerical experiments on two time-dependent problems. The problems are discretized temporally using the adaptive Runge–Kutta 5(4) method of Dormand and Prince [20] with an error tolerance of 10^{-10} (we did not use the `ode45` solver of MATLAB, even though it implements the same method, because it does not give us the option of pausing at the end of each time step to adapt the grid points), and solutions are computed using the Chebyshev spectral method and our adaptive rational spectral method with various $N < 128$. The approximation errors are calculated as the maximum absolute value of the difference between the various approximations and their $N = 128$ counterparts evaluated at 10001 equispaced points.

5.1. Blow-up problem. The first problem is the Frank–Kamenetskii or Gelfand equation

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + e^u, \quad -1 < x < 1, \tag{5.1}$$

with the initial condition $u(x, 0) = 0$ and boundary conditions $u(-1, t) = u(1, t) = 0$. The equation arises in mathematical models of exothermic chemical reactions in solids [25]. The solution of this problem is symmetric with a peak at $x = 0$ which blows up to infinity in finite time due to the nonlinear exponential term, a phenomenon known in the literature as thermal runaway. The solution has logarithmic singularities along the line $\mathrm{Re}(x) = 0$ near the blow-up time T [19], but this time is not known to high precision. The numerical experiment was run until $t = 3.544664598$. The results are summarized in Figure 2.

The first row of the figure shows the solution u approximated using the adaptive rational spectral method with $N = 56$. The grid points are shown as dots. The logarithmic blow-up effect can be seen in the fact that when time moves forward by two digits, the height of u increases by almost a fixed amount. The equidistribution of grid points along the approximation curves enables the method to resolve the tall sharp peak.

The second row shows the poles whose locations approximate the locations of the two complex conjugate singularities lying on the ellipse of analyticity of u. The poles

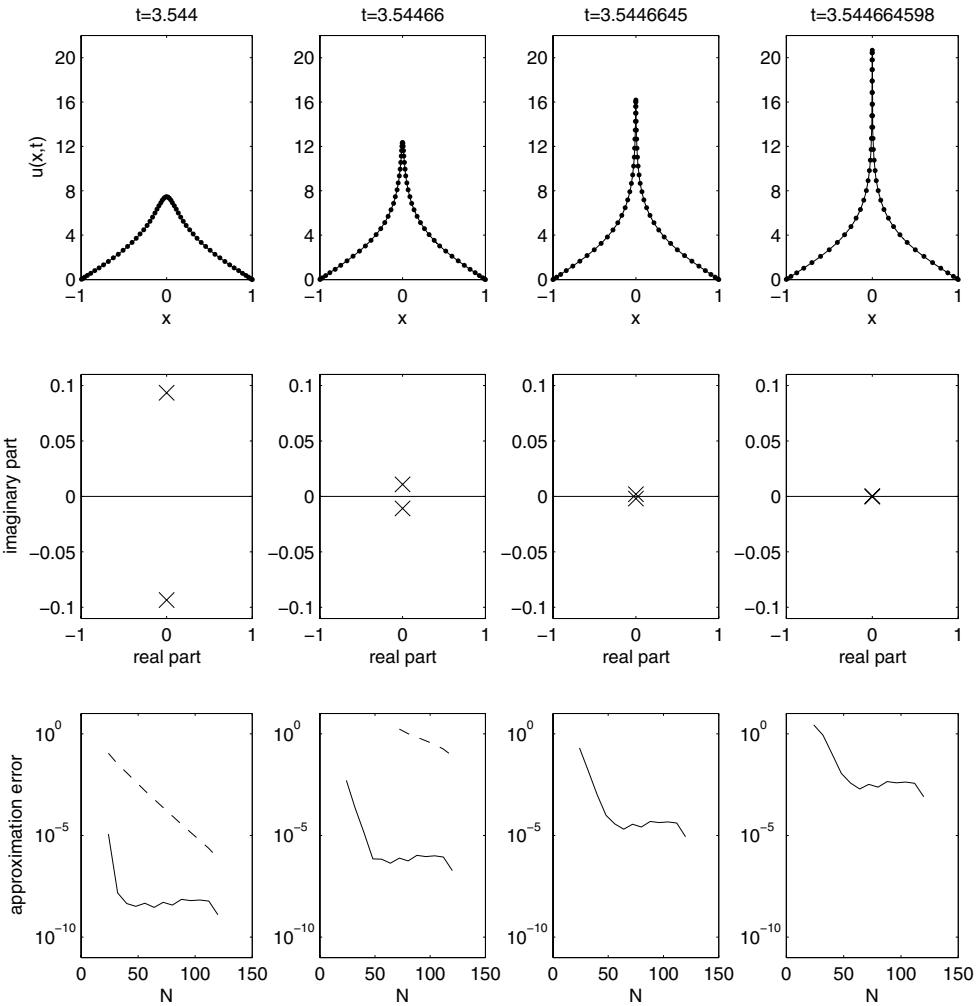


FIG. 2. *The first row shows approximations of the solution of* (5.1) *computed using the adaptive rational spectral method with* $N = 56$. *Dots represent grid points. The second row shows the poles computed using the Chebyshev–Padé singularity location technique. The third row shows log-linear plots of approximation error against* N. *Dashed lines correspond to the Chebyshev spectral method and solid lines to the adaptive rational spectral method. Dashed lines are absent where the Chebyshev spectral method failed to produce a result.*

appear along the line $\mathrm{Re}(x) = 0$ in agreement with the theory. They move closer to each other as the peak grows taller and coalesce at the blow-up time [30].

The third row shows plots of approximation error against N on a log-linear scale for the Chebyshev spectral method and the adaptive rational spectral method. As the height of u increases from about 8 to 20, the maximum number of accurate digits for the adaptive rational spectral method decreases in steps of about 2, reflecting the loss of absolute accuracy in the evaluation of e^u as its magnitude grows in steps of about 2 digits and its relative accuracy remains at about 16 digits. Apart from this quirk, the method requires about 50 grid points to produce results which are accurate to the limit of temporal errors. In contrast, the Chebyshev spectral method fails to produce any result beyond $t = 3.54466$ for $N \leq 128$.

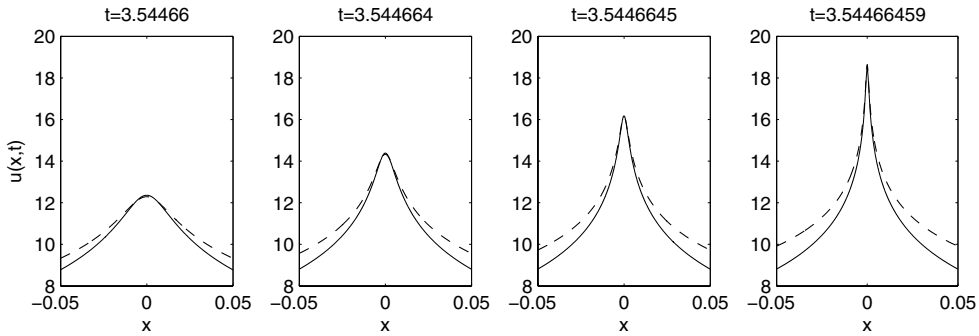


FIG. 3. *Shapes of the solution of* (5.1) *near the blow-up point and blow-up time. Dashed curves correspond to the asymptotic approximation* (5.2)*, and solid curves to the adaptive rational spectral method with* $N = 56$.

TABLE 2
The solution of (5.1) *evaluated at* $x = 0$ *approximated using the adaptive rational spectral method.*

N	$t = 3.544$	$t = 3.54466$	$t = 3.5446645$	$t = 3.544664598$
32	7.460488435	12.36019369	16.16163618	19.81771341
48	7.460488439	12.36040241	16.17636012	20.65913394
64	7.460488439	12.36040263	16.17643386	20.66847337
80	7.460488439	12.36040250	16.17642809	20.66802983
96	7.460488436	12.36040214	16.17641126	20.66650626
112	7.460488436	12.36040219	16.17641359	20.66671875

We also include Figure 3, which compares shapes of a computed approximation of u near the blow-up time with the asymptotic approximation [19]

$$u(x,t) = \xi - \log(1+\eta^2) - \frac{5}{2}\frac{\log(\xi)}{\xi}\frac{\eta^2}{1+\eta^2} + O\Big(\frac{1}{\xi}\Big), \tag{5.2}$$

where $\xi = -\log(T-t)$, $\eta = x/\sqrt{4\xi(T-t)}$, and $T = 3.544664598$. The shapes match only near $x = 0$, because (5.2) is valid only as $x \to 0$ and $t \to T$. In addition, we include Table 2, which lists the height of u approximated using the adaptive rational spectral method with various N, and hope that it will serve as a reference for further study of this blow-up problem.

5.2. Moving front problem. The second problem is the Burgers equation

$$\frac{\partial u}{\partial t} = \nu\frac{\partial^2 u}{\partial x^2} - u\frac{\partial u}{\partial x}, \quad 0 < x < 1, \tag{5.3}$$

with the initial condition $u(x,0) = \sin(2\pi x) + \frac{1}{2}\sin(\pi x)$ and boundary conditions $u(0,t) = u(1,t) = 0$. This problem serves as a common test problem for adaptive grid methods (see [23] and the references therein). Its solution develops a steep front which moves towards $x = 1$ but, due to the homogeneous boundary conditions, also decays with time. The width of the front is $O(\nu)$. The numerical experiment was run until $t = 0.90$ with $\nu = 10^{-3}$. The results are summarized in Figure 4.

The first row of the figure shows the solution u approximated using the adaptive rational spectral method with $N = 56$. The grid points are shown as dots. The clustering of grid points where the approximation curves have steep gradients enables the method to resolve the thin steep front.

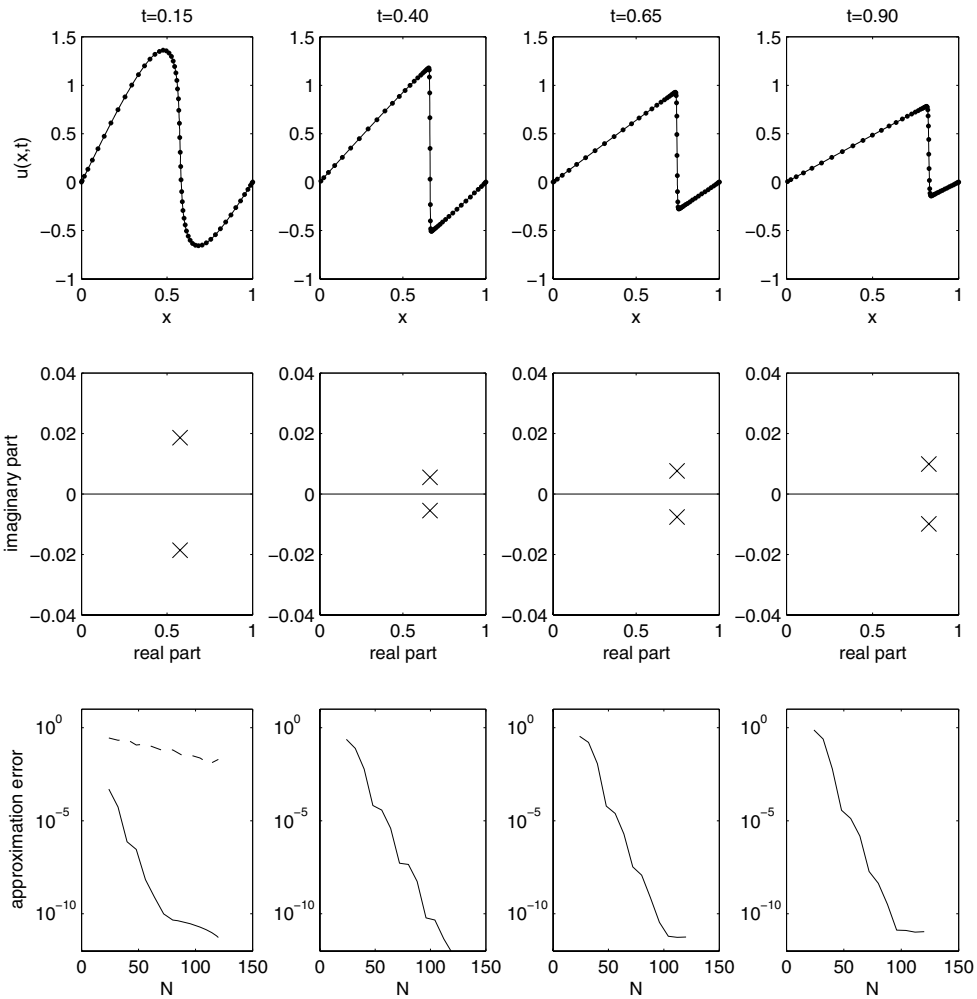


FIG. 4. *Like Figure* 2 *but for* (5.3) *with* $\nu = 10^{-3}$.

The second row shows the poles whose locations approximate the locations of the two complex conjugate singularities lying on the ellipse of analyticity of u. The poles move closer to each other when the front steepens, and apart again when the front decays. They also move towards $\mathrm{Re}(x) = 1$ in accordance with the front. In the absence of viscosity, they would coalesce when a shock is formed [30].

The third row shows plots of approximation error against N on a log-linear scale for the Chebyshev spectral method and the adaptive rational spectral method. The latter requires about 100 grid points to produce results which are accurate to the limit of temporal errors. In contrast, the Chebyshev spectral method fails to produce any result beyond $t = 0.15$ for $N \leq 128$.

We also vary ν and include Figure 5, which shows plots of approximation error against N on a log-linear scale for the adaptive rational spectral method with various ν. Each time ν decreases geometrically by a factor of 10, the number of grid points required for the method to produce results accurate to the limit of temporal errors increases by an approximately constant amount of 50.

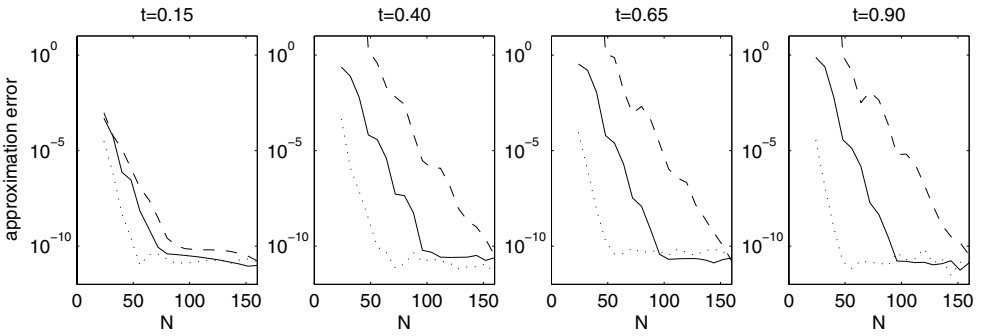


FIG. 5. *Log-linear plots of approximation error against N for the solution of* (5.3) *approximated using the adaptive rational spectral method. Dotted lines correspond to* $\nu = 10^{-2}$*, solid lines to* $\nu = 10^{-3}$*, and dashed lines to* $\nu = 10^{-4}$*.*

6. Conclusion. We have presented an adaptive rational spectral method which far outperforms the Chebyshev spectral method for problems whose solutions have singularities in the complex plane close to $[-1, 1]$. It combines ideas from existing rational spectral methods [3, 11] with an adaptive technique for locating singularities [30] using Chebyshev–Padé approximation. The method can be improved in various ways. For example, one can derive maps analogous to (2.7) for problems on periodic intervals or problems with multiple complex conjugate singularities. We are in the process of doing this using the framework of Schwarz–Christoffel mapping [21]. We are also studying the maps used in other adaptive spectral methods and hope to compare the effectiveness of those maps with ours. In addition, we are investigating further improvements and applications to some of the innumerable problems of scientific computing whose solutions involve fronts, boundary layers, or other singularities.

REFERENCES

[1] R. Baltensperger and J.-P. Berrut, *The errors in calculating the pseudospectral differentiation matrices for Čebyšev–Gauss–Lobatto points*, Comput. Math. Appl., 37 (1999), pp. 41–48.

[2] R. Baltensperger and J.-P. Berrut, Errata to *The errors in calculating the pseudospectral differentiation matrices for Čebyšev–Gauss–Lobatto points*, Comput. Math. Appl., 38 (1999), p. 119.

[3] R. Baltensperger and J.-P. Berrut, *The linear rational collocation method*, J. Comput. Appl. Math., 134 (2001), pp. 243–258.

[4] R. Baltensperger, J.-P. Berrut, and B. Noël, *Exponential convergence of a linear rational interpolant between transformed Chebyshev points*, Math. Comp., 68 (1999), pp. 1109–1120.

[5] Z. Battles and L. N. Trefethen, *An extension of MATLAB to continuous functions and operators*, SIAM J. Sci. Comput., 25 (2004), pp. 1743–1770.

[6] A. Bayliss, D. Gottlieb, B. Matkowsky, and M. Minkoff, *An adaptive pseudospectral method for reaction diffusion problems*, J. Comput. Phys., 81 (1989), pp. 421–443.

[7] A. Bayliss and B. Matkowsky, *Fronts, relaxation oscillations and period doubling in solid fuel combustion*, J. Comput. Phys., 71 (1987), pp. 147–188.

[8] A. Bayliss and E. Turkel, *Mappings and accuracy for Chebyshev pseudo-spectral approximations*, J. Comput. Phys., 101 (1992), pp. 349–359.

[9] J.-P. Berrut, *Rational functions for guaranteed and experimentally well-conditioned global interpolation*, Comput. Math. Appl., 15 (1988), pp. 1–16.

[10] J.-P. Berrut, *The barycentric weights of rational interpolation with prescribed poles*, J. Comput. Appl. Math., 86 (1997), pp. 45–52.
[11] J.-P. Berrut and R. Baltensperger, *The linear rational pseudospectral method for boundary value problems*, BIT, 41 (2001), pp. 868–879.
[12] J.-P. Berrut, R. Baltensperger, and H. D. Mittelmann, *Recent developments in barycentric rational interpolation*, Internat. Ser. Numer. Math., 151 (2005), pp. 27–51.
[13] J.-P. Berrut and H. D. Mittelmann, *Rational interpolation through the optimal attachment of poles to the interpolating polynomial*, Numer. Algorithms, 23 (2000), pp. 315–328.
[14] J.-P. Berrut and H. D. Mittelmann, *The linear rational pseudospectral method with iteratively optimized poles for two-point boundary value problems*, SIAM J. Sci. Comput., 23 (2001), pp. 961–975.
[15] J.-P. Berrut and H. D. Mittelmann, *Linear rational interpolation and its application in approximation and boundary value problems*, Rocky Mountain J. Math., 32 (2002), pp. 527–544.
[16] J.-P. Berrut and H. D. Mittelmann, *Optimized point shifts and poles in the linear rational pseudospectral method for boundary value problems*, J. Comput. Phys., 204 (2005), pp. 292–301.
[17] J.-P. Berrut and L. N. Trefethen, *Barycentric Lagrange interpolation*, SIAM Rev., 46 (2004), pp. 501–517.
[18] D. Day and L. Romero, *Roots of polynomials expressed in terms of orthogonal polynomials*, SIAM J. Numer. Anal., 43 (2005), pp. 1969–1987.
[19] J. W. Dold, *Analysis of the early stage of thermal runaway*, Quart. J. Mech. Appl. Math., 38 (1985), pp. 361–387.
[20] J. R. Dormand and P. J. Prince, *A family of embedded Runge-Kutta formulae*, J. Comput. Appl. Math., 6 (1980), pp. 19–26.
[21] T. A. Driscoll and L. N. Trefethen, *Schwarz–Christoffel Mapping*, Cambridge University Press, Cambridge, UK, 2002.
[22] H. Guillard and R. Peyret, *On the use of spectral methods for the numerical solution of stiff problems*, Comput. Methods Appl. Mech. Engrg., 66 (1988), pp. 17–43.
[23] W. Huang, Y. Ren, and R. D. Russell, *Moving mesh methods based on moving mesh partial differential equations*, J. Comput. Phys., 113 (1994), pp. 279–290.
[24] D. Kosloff and H. Tal-Ezer, *A modified Chebyshev pseudospectral method with an $O(N^{-1})$ time step restriction*, J. Comput. Phys., 104 (1993), pp. 457–469.
[25] A. A. Lacey, *Diffusion models with blow-up*, J. Comput. Appl. Math., 97 (1998), pp. 39–49.
[26] L. S. Mulholland, W.-Z. Huang, and D. M. Sloan, *Pseudospectral solution of near-singular problems using numerical coordinate transformations based on adaptivity*, SIAM J. Sci. Comput., 19 (1998), pp. 1261–1289.
[27] L. S. Mulholland, Y. Qiu, and D. M. Sloan, *Solution of evolutionary partial differential equations using adaptive finite differences with pseudospectral post-processing*, J. Comput. Phys., 131 (1997), pp. 280–298.
[28] S. P. Suetin, *On the convergence of rational approximations to polynomial expansions in domains of meromorphy of a given function.*, Math. USSR-Sb., 34 (1978), pp. 367–381.
[29] L. N. Trefethen, *Spectral Methods in MATLAB*, SIAM, Philadelphia, 2000.
[30] J. A. C. Weideman, *Computing the dynamics of complex singularities of nonlinear PDEs*, SIAM J. Appl. Dyn. Syst., 2 (2003), pp. 171–186.

SIAM J. SCI. COMPUT.
Vol. 28, No. 5, pp. 1812–1836
© 2006 Society for Industrial and Applied Mathematics

EFFICIENT COMPUTATION OF ISOMETRY-INVARIANT DISTANCES BETWEEN SURFACES*

ALEXANDER M. BRONSTEIN†, MICHAEL M. BRONSTEIN†, AND RON KIMMEL†

Abstract. We present an efficient computational framework for isometry-invariant comparison of smooth surfaces. We formulate the Gromov–Hausdorff distance as a multidimensional scaling–like continuous optimization problem. In order to construct an efficient optimization scheme, we develop a numerical tool for interpolating geodesic distances on a sampled surface from precomputed geodesic distances between the samples. For isometry-invariant comparison of surfaces in the case of partially missing data, we present the partial embedding distance, which is computed using a similar scheme. The main idea is finding a minimum-distortion mapping from one surface to another, while considering only relevant geodesic distances. We discuss numerical implementation issues and present experimental results that demonstrate its accuracy and efficiency.

Key words. generalized multidimensional scaling, isometric embedding, Gromov–Hausdorff distance, intrinsic geometry, isometry-invariant surface matching, iterative optimization, multiresolution methods

AMS subject classifications. 15A15, 15A09, 15A23

DOI. 10.1137/050639296

1. Introduction.

Many objects in nature can be described as nonrigid deformable (or bendable) surfaces. Often, the metric and the topological properties of the surface are preserved during bending. For example, most human organs bend without significantly stretching or tearing their surfaces. Study of living organs done through medical imaging often puts forth the challenge of comparing two deformable surfaces, which in turn requires identifying surface properties invariant to deformations. This paper addresses the problem of bending-invariant comparison between surfaces. More formally, we distinguish between the *extrinsic* and the *intrinsic* geometry of a surface. *Extrinsic* properties describe the way a surface is immersed into the ambient space. These properties are subject to changes while the surface bends. *Intrinsic* properties, on the other hand, are associated with the internal metric structure of the surface. Most natural deformations preserve, at least approximately, the intrinsic properties. Such deformations are called *inelastic deformations*. Our goal is therefore to define an efficiently computable *inelastic deformation invariant* measure of similarity between surfaces.

Going back to the human organs example, an interesting application for such a measure of similarity is expression-invariant three-dimensional face recognition. In [6], we showed that facial expressions can be modeled as near-isometric deformations of the facial surface. Thus, an isometry-invariant similarity measure between faces would be a good candidate for face recognition insensitive to facial expressions.

1.1. Prior work.

The case of Euclidean-isometry matching of surfaces was intensively explored. A classical result for rigid surface matching is the *Cohn-Vossen rigidity*

*Received by the editors August 31, 2005; accepted for publication (in revised form) May 2, 2006; published electronically October 27, 2006. This work was supported in part by the H. & R. Sohnis Cardiology Research Fund, United States–Israel Science Foundation grant 2004274, Israel Science Foundation grant 738/04, and the European FP 6 NoE 507752 (MUSCLE).
http://www.siam.org/journals/sisc/28-5/63929.html
†Department of Computer Science, Technion – Israel Institute of Technology, Haifa 32000, Israel (alexbron@ieee.org, bronstein@ieee.org, ron@cs.technion.ac.il).

(ICP) method [27], finding a Euclidean transformation between two discrete surfaces represented as point clouds.

A near-isometry-invariant comparison of surfaces based on Euclidean embeddings was presented by Elad and Kimmel in [13] as a generalization of [23]. The main idea is to map the intrinsic (metric) structure of the surfaces to a low-dimensional space and compare the resulting images in this space. The numerical core of the Elad–Kimmel approach is based on a multidimensional scaling (MDS) algorithm [3], whose numerical solutions can be computed efficiently. It is also related to dimensionality reduction [22, 12, 25] and to texture mapping [28, 17]. Similar problems are analyzed in graph theory in the context of representation of discrete metric spaces [20]. Yet, when dealing with abstract metric spaces like graphs, or even point clouds with local distances, smoothness of the underlying geometry cannot be assumed, and thus embedding is a much harder problem.

One of the caveats of the Euclidean embedding approach is the fact that usually a Riemannian surface cannot be perfectly represented in a finite-dimensional Euclidean space, thus inevitably introducing a metric distortion. It was argued and proved empirically in [5, 4] that using a two-dimensional or a three-dimensional sphere instead of the Euclidean space results in smaller embedding error for certain types of objects, and consequently the recognition results get better. In [26], embedding into a space with hyperbolic geometry was shown.

In [16], Gromov introduced the Gromov–Hausdorff distance, which in a sense measures the metric distortion between metric spaces. Such a distance is invariant to isometries. In their recent fundamental paper [21], Mémoli and Sapiro adopted the Gromov–Hausdorff distance to the problem of deformable shape recognition. They considered a discrete setting in which surfaces are given as point clouds. In this case the computation of the Gromov–Hausdorff distance requires evaluating all the permutations between the samples. Such a combinatorial optimization problem appears to be computationally prohibitive. Mémoli and Sapiro showed a theoretical probabilistic framework and a practical procedure, which allowed them to formulate an approximation of the Gromov–Hausdorff distance between two point clouds without computing all the permutations.

1.2. Main contribution. In many applications, like computer graphics, surfaces are represented as triangulated meshes that capture the local smoothness properties of the underlying surface. One contribution of our paper is the derivation of an MDS-like formulation for the Gromov–Hausdorff distance between surfaces. This results in a continuous nonconvex optimization problem, which lends itself to a numerically exact computation of the Gromov–Hausdorff distance between surfaces.

In many practical cases, where surfaces are given only partially, the Gromov–Hausdorff distance cannot be used. For this purpose we introduced in [7] the , which can handle isometry-invariant partial surface matching. The key idea is to embed one surface into another and measure the minimal possible distortion of such a mapping. A numerical procedure devised in [7] for this purpose is similar in nature to MDS and therefore has been named or GMDS. GMDS can be thought of as a natural extension of previous works on isometric embedding into non-Euclidean spaces [26, 5, 4], as visualized in Figure 1. Here we provide the formal support for the framework presented in [7] and show the relationship between our partial embedding distance and the Gromov–Hausdorff distance.

Our paper comprises five sections. Section 2 provides the theoretical foundations of isometry-invariant surface matching. There, we define the Gromov–Hausdorff dis-

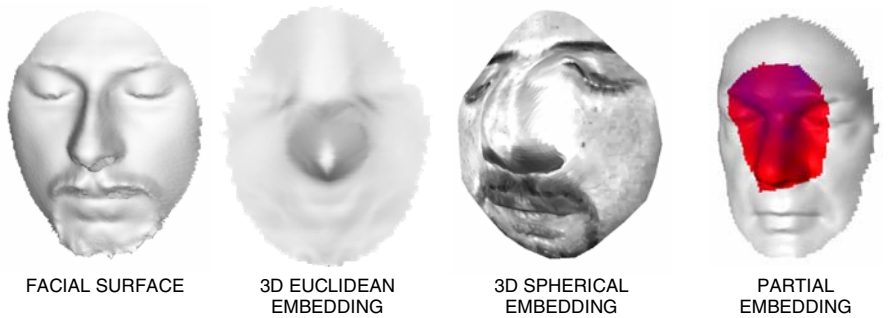


FIG. 1. *Evolution of isometry-invariant object recognition based on isometric embedding, exemplified on the problem of face recognition. Left to right: a facial surface; embedding into* $\mathbb{R}^3$ [6]*; embedding into* $\mathbb{S}^3$ (*shown is the maximum-variance projection onto* $\mathbb{R}^3$) [5]*; embedding part of the face into another facial surface* [7].

TABLE 1
Notation and symbols.

$\mathbb{R}$; $\mathbb{R}^+$	Real numbers; nonnegative real numbers
$\mathbb{R}^m$	m-dimensional Euclidean space
$\mathbb{R}^{M\times N}$	Space of $M \times N$ matrices
$\mathbb{M}$	Set of compact metric spaces
$(\mathcal{S}, d_{\mathcal{S}})$; $\mathcal{S}$	Metric space $\mathcal{S}$ with metric $d_{\mathcal{S}}$ arising from a Riemannian surface
s ; $\mathbf{s}$	A point on $\mathcal{S}$; extrinsic coordinates of s
$d_{\mathcal{S}}\vert_{\mathcal{S}'}$	Restriction of $d_{\mathcal{S}}$ to $\mathcal{S}'$
$\mathcal{S}_N$	Finite sampling of $\mathcal{S}$ consisting of N points
$\mathbf{D}_{\mathcal{S}}$	Matrix of geodesic distances on $\mathcal{S}_N$
$T_{\mathcal{S}}$; t^k	Triangulation of $\mathcal{S}_N$; kth triangle in $T_{\mathcal{S}}$
$\mathcal{S}^r$; $\mathcal{S}^r_N$	r-covering; finite r-covering of $\mathcal{S}$
$B_{\mathcal{S}}(s_0, r)$	Open ball of radius r in the metric space $\mathcal{S}$ around s_0
$\operatorname{diam}\mathcal{S}$	Diameter of metric space $\mathcal{S}$
$\operatorname{dis}\psi$	Distortion of the mapping ψ
$d_{\mathrm{H}}^{\mathcal{Z}}$	Hausdorff metric in the metric space $\mathcal{Z}$
d_{GH}	Gromov–Hausdorff metric between metric spaces
d_{P}	Permutation distance
d_{PE}	Partial embedding distance
Π_N ; π	Permutations of the set $\{1, \ldots, N\}$; a permutation in Π_N
$\mathbf{1}_{N\times N}$	$N \times N$ matrix of ones

tance and review previous attempts for its approximation. We also describe the partial embedding distance and summarize its properties. In section 3, we introduce an efficient algorithm for our isometry-invariant distance computation. We address the problem of geodesic distance interpolation, a vital component of our numerical algorithm. We also discuss the question of GMDS convergence and show a multiresolution optimization approach as a means to overcome convergence to local minima. Section 4 presents an experimental validation of the proposed framework. Finally, section 5 concludes the paper. Proofs and technical details can be found in the appendices. Notation and symbols used throughout the paper are summarized in Table 1.

2. Theoretical background. The objects we deal with are surfaces. We use $\mathcal{S}$ to denote a surface, and assume it to be modeled as a smooth connected complete Riemannian 2-manifold. We denote by $d_{\mathcal{S}} : \mathcal{S} \times \mathcal{S} \mapsto \mathbb{R}^+$ the geodesic distance function induced by the Riemannian metric. For simplicity, hereafter we say [illegible] $\mathcal{S}$, implying the underlying metric space $(\mathcal{S}, d_{\mathcal{S}})$. We assume, without loss of generality, that $\mathcal{S}$ is immersed into $\mathbb{R}^3$ and denote the extrinsic coordinates of a point $s \in \mathcal{S}$ by $\mathbf{s} = (s^1, s^2, s^3)$.

In practice, we work with sampled surfaces represented by finite metric spaces. To this end, we use the following definition: a subset $\mathcal{S}^r \subset \mathcal{S}$ is called an r-[illegible] of $\mathcal{S}$ (or an r-[illegible] in $\mathcal{S}$) if

$$\mathcal{S} = \bigcup_{s \in \mathcal{S}^r} B_{\mathcal{S}}(s, r). \tag{1}$$

Here $B_{\mathcal{S}}(s_0, r) = \{s \in \mathcal{S} : d_{\mathcal{S}}(s, s_0) < r\}$ is an open ball of radius r around s in $\mathcal{S}$. A finite r-covering of $\mathcal{S}$ consisting of N points is denoted by $\mathcal{S}^r_N$. The space $(\mathcal{S}^r_N, d_{\mathcal{S}}|_{\mathcal{S}^r_N})$ constitutes a finite metric space, where the metric $d_{\mathcal{S}}|_{\mathcal{S}^r_N}$ is obtained by restricting the original metric $d_{\mathcal{S}}$ to the subset $\mathcal{S}^r_N$. An arbitrary finite sampling of $\mathcal{S}$ consisting of N points (a[illegible]) is denoted by $\mathcal{S}_N = \{s_1, \ldots, s_N\}$. In order to make a distinction between the original and the sampled surfaces, we refer to $\mathcal{S}$ as a [illegible] surface and to $\mathcal{S}_N$ as a [illegible] surface.

A sampled surface can be represented as a cloud of points $\mathcal{S}_N$ and an $N \times N$ matrix $\mathbf{D}_{\mathcal{S}} = (d_{\mathcal{S}_N}(s_i, s_j))$ of the corresponding geodesic distances. In addition, we assume that a triangulation $T_{\mathcal{S}}$ capturing the local geometry of the surface is given (here $T_{\mathcal{S}} = \{(t_{11}, t_{12}, t_{13}), \ldots, (t_{K,1}, t_{K,2}, t_{K,3})\}$ is a list of K triangles, where each triangle is a triplet of sample indices, $t_{ki} \in \{1, \ldots, N\}$). The resulting triangulated mesh can be thought of as a polyhedral approximation of the smooth surface $\mathcal{S}$.

Surface $\mathcal{S}$ is called [illegible] if $\operatorname{diam}\mathcal{S} = \sup_{s,s' \in \mathcal{S}} d_{\mathcal{S}}(s, s')$ is finite. $\mathcal{S}$ is said to be [illegible] if all its open covers have a finite subcover. If $\mathcal{S}$ is compact, it is necessarily bounded. This will be our tacit assumption, as surfaces encountered in practical applications are compact. We use $\mathbb{M}$ to denote the set of all such surfaces.

Given two surfaces $\mathcal{S}$ and $\mathcal{Q}$ in $\mathbb{M}$, a transformation $\varphi : \mathcal{S} \mapsto \mathcal{Q}$ is said to have a [illegible] or distort the metric by

$$\operatorname{dis}\varphi \equiv \sup_{s,s' \in \mathcal{S}} \left| d_{\mathcal{S}}(s, s') - d_{\mathcal{Q}}(\varphi(s), \varphi(s')) \right|. \tag{2}$$

If in addition for every $q \in \mathcal{Q}$ there exists $s \in \mathcal{S}$ such that $d_{\mathcal{Q}}(q, \varphi(s)) \leq \epsilon$ (we call this property ϵ-[illegible]), φ is called an ϵ-[illegible] and the spaces $\mathcal{S}$ and $\mathcal{Q}$ are termed ϵ-[illegible]. The image of $\mathcal{S}$, $\varphi(\mathcal{S})$ is called an ϵ-[illegible] of $\mathcal{S}$ in $\mathcal{Q}$. A transformation $\varphi : \mathcal{S} \mapsto \mathcal{Q}$ with $\operatorname{dis}\varphi = 0$ is called an [illegible], and $\mathcal{S}$ and $\mathcal{Q}$ admitting such a transformation are called [illegible]. Unlike ϵ-isometries, an isometry is always bijective and bi-Lipschitz continuous.

In this paper, we address the question of matching between two surfaces, insensitively to their isometries. More formally, we study a [illegible] $d : \mathbb{M} \times \mathbb{M} \mapsto \mathbb{R}^+$, which assigns a nonnegative number to every pair of surfaces, measuring the degree of their similarity. The basic property we require is that for all isometries f of $\mathcal{S}$ and g of $\mathcal{Q}$, $d(f(\mathcal{S}), g(\mathcal{Q})) = d(\mathcal{S}, \mathcal{Q})$.

2.1. Gromov–Hausdorff distance. Mémoli and Sapiro in [21] introduced the Gromov–Hausdorff distance into the deformable surface matching arena. Here we use this powerful theory and present numerical methods for its consistent computation

subject to the disclaimer that we cannot guarantee completely avoiding the intrinsic local minima of the associated measure. Our starting point is the classical Hausdorff distance. Assume that the surfaces $\mathcal{S}$ and $\mathcal{Q}$ are subsets of some larger metric space $(\mathcal{Z}, d_{\mathcal{Z}})$. Then the [illegible] can be defined in the following way:

$$d_{\mathrm{H}}^{\mathcal{Z}}(\mathcal{S},\mathcal{Q}) \equiv \max\left\{\sup_{s\in\mathcal{S}} d_{\mathcal{Z}}(s,\mathcal{Q}), \sup_{q\in\mathcal{Q}} d_{\mathcal{Z}}(q,\mathcal{S})\right\}. \tag{3}$$

Here for notational simplicity we use $d_{\mathcal{Z}}(s,\mathcal{Q}) = \inf_{q\in\mathcal{Q}} d_{\mathcal{Z}}(s,q)$, implying a point-to-set distance. It can be shown that $d_{\mathrm{H}}^{\mathcal{Z}}$ is a metric on all the subsets of $\mathcal{Z}$. Versions of the Hausdorff distance are often used to measure similarity between rigid surfaces in $\mathbb{R}^3$, which give birth to a family of algorithms known as [illegible], or ICP [2, 27, 11]. Yet, while being a good match for comparison of extrinsic geometries, the Hausdorff distance is not isometry-invariant.

In order to overcome this problem, Mémoli and Sapiro [21] adopted an extension of the Hausdorff distance introduced by Gromov [16]. The [illegible] is defined by

$$d_{\mathrm{GH}}(\mathcal{S},\mathcal{Q}) \equiv \inf_{\substack{\mathcal{Z}\\ \rho:\mathcal{S}\mapsto\mathcal{Z}\\ \sigma:\mathcal{Q}\mapsto\mathcal{Z}}} d_{\mathrm{H}}^{\mathcal{Z}}(\rho(\mathcal{S}),\sigma(\mathcal{Q})), \tag{4}$$

where $\rho : \mathcal{S} \mapsto \mathcal{Z}$ and $\sigma : \mathcal{Q} \mapsto \mathcal{Z}$ are isometric embeddings into the metric space $\mathcal{Z}$. The Gromov–Hausdorff distance is invariant to isometries and thus allows a comparison between two surfaces in a truly intrinsically geometric way. In addition, d_{GH} has several nice properties, which we summarize here.

1. [illegible] d_{GH} is a finite metric on the space of compact surfaces $\mathbb{M}$, in which a point is defined as an equivalence class of a surface and its isometries [15, 10]. In other words, d_{GH} satisfies the following axioms: (i) $d_{\mathrm{GH}}(\mathcal{S},\mathcal{Q}) \geq 0$; (ii) $d_{\mathrm{GH}}(\mathcal{S},\mathcal{Q}) = 0$ if and only if $\mathcal{S}$ and $\mathcal{Q}$ are isometric; (iii) $d_{\mathrm{GH}}(\mathcal{S},\mathcal{Q}) = d_{\mathrm{GH}}(\mathcal{Q},\mathcal{S})$; and (iv) $d_{\mathrm{GH}}(\mathcal{R},\mathcal{Q}) \leq d_{\mathrm{GH}}(\mathcal{R},\mathcal{S}) + d_{\mathrm{GH}}(\mathcal{S},\mathcal{Q})$, also known as the [illegible].
2. [illegible] (i) If $d_{\mathrm{GH}}(\mathcal{S},\mathcal{Q}) \leq \epsilon$, then there exists a 2ϵ-Hausdorff approximation of $\mathcal{S}$ in $\mathcal{Q}$; (ii) if there exists an ϵ-Hausdorff approximation of $\mathcal{S}$ in $\mathcal{Q}$, then $d_{\mathrm{GH}}(\mathcal{S},\mathcal{Q}) \leq 2\epsilon$. This property can be thought of as an extension of the second metric axiom to ϵ-isometries (see [10] for details).
3. [illegible] Given $\mathcal{S}^r$ an r-covering of $\mathcal{S}$ and $\mathcal{Q}^{r'}$ and r'-covering of $\mathcal{Q}$, $|d_{\mathrm{GH}}(\mathcal{S},\mathcal{Q}) - d_{\mathrm{GH}}(\mathcal{S}^r,\mathcal{Q}^{r'})| \leq r + r'$. In other words, the Gromov–Hausdorff distance between sampled versions of two surfaces is close to the Gromov–Hausdorff distance between the original ones, and the discrepancy vanishes as the sample radius decreases.

The main disadvantage of $d_{\mathrm{GH}}(\mathcal{S},\mathcal{Q})$ according to definition (4) is its computational intractability.

Fortunately, for bounded metric spaces, the Gromov–Hausdorff distance can be translated to comparison of distances within $\mathcal{S}$ and $\mathcal{Q}$ to each other, thus avoiding the cumbersome construction of the metric space $\mathcal{Z}$. Formally, the Gromov–Hausdorff distance can be redefined as follows [18, 21]:

$$d_{\mathrm{GH}}(\mathcal{S},\mathcal{Q}) = \frac{1}{2}\inf_{\substack{\varphi:\mathcal{S}\mapsto\mathcal{Q}\\ \psi:\mathcal{Q}\mapsto\mathcal{S}}} \max\left\{\operatorname{dis}\phi, \operatorname{dis}\psi, \operatorname{dis}(\varphi,\psi)\right\}, \tag{5}$$

where $\operatorname{dis}\varphi$ and $\operatorname{dis}\psi$ defined in (2) measure the metric distortion of the mappings φ and ψ, respectively, and

$$\operatorname{dis}(\varphi,\psi) \equiv \sup_{\substack{s\in\mathcal{S}\\ q\in\mathcal{Q}}} \left| d_{\mathcal{S}}(s,\psi(q)) - d_{\mathcal{Q}}(q,\varphi(s)) \right| \tag{6}$$

measures how far φ and ψ are from being the inverse of each other. This reformulation, as we will show, allows us an efficient computation of the Gromov–Hausdorff distance between surfaces.

2.2. Mémoli–Sapiro approximation of d_{GH}. In [21], Mémoli and Sapiro considered the approximation of the Gromov–Hausdorff distance between point clouds by dropping the term $\operatorname{dis}(\varphi,\psi)$ in the definition of d_{GH} in (5) and bounding its contribution probabilistically. Given two bounded surfaces $\mathcal{S}$ and $\mathcal{Q}$, both sampled at N points ($\mathcal{S}_N$ and $\mathcal{Q}_N$, respectively), they defined the

$$d_{\mathrm{P}}(\mathcal{S}_N,\mathcal{Q}_N) \equiv \frac{1}{2}\min_{\pi\in\Pi_N}\max_{1\le i,j\le N}\left| d_{\mathcal{S}}(s_i,s_j) - d_{\mathcal{Q}}(q_{\pi_i},q_{\pi_j}) \right|,$$

where Π_N denotes all the permutations of $\{1,\dots,N\}$. d_{P} is not the exact Gromov–Hausdorff distance but a different measure of surface similarity related to d_{GH}. Obviously, $d_{\mathrm{GH}}(\mathcal{S}_N,\mathcal{Q}_N) \le d_{\mathrm{P}}(\mathcal{S}_N,\mathcal{Q}_N)$, since d_{P} can be considered a particular discrete case of d_{GH}, in which the correspondences between the two metric spaces are restricted only to the bijective ones. The upper bound is much more complicated and is derived in a probabilistic setting [21]. Based on these results, Mémoli and Sapiro described an algorithm for the approximation of d_{GH}. The key idea is to select subsamplings of the given point clouds, the distance between which can be related to the Gromov–Hausdorff distance. Alternative, more practical estimation methods of the distance between point clouds, motivated by the Gromov–Hausdorff distance, were also presented in [21].

Mémoli and Sapiro also noted that by restricting the $\mathcal{Z}$ space in definition (4) of d_{GH}, the metric actually boils down to the canonical form distance proposed in [13]. The canonical form distance is computed by first finding the best representation of $\mathcal{S}$ and $\mathcal{Q}$ in $\mathcal{Z}$ (in the sense of lowest metric distortion). Let $\rho : \mathcal{S} \mapsto \mathcal{Z}$ be the lowest distortion embedding of $\mathcal{S}$ in $\mathcal{Z}$; the image $\mathcal{S}_{\mathrm{can}} = \rho(\mathcal{S})$ is referred to as the canonical form of $\mathcal{S}$. Similarly, the canonical form of $\mathcal{Q}$ is computed. Once $\mathcal{S}_{\mathrm{can}}$ and $\mathcal{Q}_{\mathrm{can}}$ are available, the canonical form distance is computed according to

$$d_{\mathrm{CF}}(\mathcal{S},\mathcal{Q}) = d^{\mathcal{Z}}(\mathcal{S}_{\mathrm{can}},\mathcal{Q}_{\mathrm{can}}), \tag{7}$$

where $d^{\mathcal{Z}}$ is some distance between two sets in $\mathcal{Z}$ (e.g., the Hausdorff distance or its variant). The typical choice of $\mathcal{Z}$ is $\mathbb{R}^3$.

2.3. Partial embedding distance. The main drawback of the Gromov–Hausdorff distance and its approximate version is the fact that it compares two metric spaces as a whole, which is inappropriate for partial matching. One of the most significant differences of partial matching compared to its "full" counterpart is the lack of symmetry. We therefore address the problem of finding a match between a surface $\mathcal{S}$ and a $\mathcal{Q}$, which is a deformed version of the model. In practice, one is often required to find whether a $\mathcal{Q}' \subset \mathcal{Q}$ is similar to $\mathcal{S}$.

In [7] we defined the nonsymmetric [illegible] as

$$d_{\mathrm{PE}}(\mathcal{S},\mathcal{Q}') \equiv \frac{1}{2}\inf_{\psi:\mathcal{Q}\mapsto\mathcal{S}} \sup_{(q,q')\in P} \left|d_{\mathcal{Q}}(q,q') - d_{\mathcal{S}}(\psi(q),\psi(q'))\right|, \tag{8}$$

where $P = \{(q,q') : d_{\mathcal{Q}'}(q,q') = d_{\mathcal{Q}}(q,q')\} \subseteq \mathcal{Q}\times\mathcal{Q}$. In other words, we define the distortion only on pairs of points for which the metric on the patch $\mathcal{Q}'$ is consistent with that on $\mathcal{Q}$. Such a restriction is necessary because the assumption $d_{\mathcal{Q}'} = d_{\mathcal{Q}}|_{\mathcal{Q}'}$ (i.e., that $d_{\mathcal{Q}'}(q,q') = d_{\mathcal{Q}}(q,q')$ for all $q,q' \in \mathcal{Q}'$) holds only in particular cases, e.g., when $\mathcal{Q}'$ is a convex subset of the surface $\mathcal{Q}$. In general, minimal geodesics on $\mathcal{Q}$ that cross the boundary $\partial\mathcal{Q}'$ change in $\mathcal{Q}'$, and therefore the geodesic distances between the corresponding points in $\mathcal{Q}$ and $\mathcal{Q}'$ may differ. Note that when $P = \mathcal{Q}\times\mathcal{Q}$, we have $\sup_{(q,q')\in P}|d_{\mathcal{Q}}(q,q') - d_{\mathcal{S}}(\psi(q),\psi(q'))| = \operatorname{dis}\psi$ and the partial embedding distance becomes

$$d_{\mathrm{PE}}(\mathcal{S},\mathcal{Q}) = \frac{1}{2}\inf_{\psi:\mathcal{Q}\mapsto\mathcal{S}} \operatorname{dis}\psi. \tag{9}$$

A comment about the potential usefulness of (9) for partial matching was made in [21].

The properties of d_{PE} resemble those of d_{GH}. We summarize them here; for proofs, see Appendix A.

1. [illegible] d_{PE} satisfies some of the metric axioms: (i) $d_{\mathrm{PE}}(\mathcal{S},\mathcal{Q}) \geq 0$; (ii) $d_{\mathrm{PE}}(\mathcal{S},\mathcal{Q}) = 0$ if and only if $\mathcal{Q}$ is isometrically embeddable into $\mathcal{S}$; (iii) $d_{\mathrm{PE}}(\mathcal{S},\mathcal{Q}) \leq d_{\mathrm{PE}}(\mathcal{S},\mathcal{R}) + d_{\mathrm{PE}}(\mathcal{R},\mathcal{Q})$, which is a nonsymmetric version of the triangle inequality.
2. [illegible] (i) If there exists an ϵ-Hausdorff approximation of $\mathcal{S}$ in $\mathcal{Q}$, then $d_{\mathrm{PE}}(\mathcal{S},\mathcal{Q}) \leq 2\epsilon$. (ii) The converse appears to be true under some more restrictive assumptions (the proof is to be published elsewhere).
3. [illegible] Given $\mathcal{S}^r$ an r-covering of $\mathcal{S}$ and $\mathcal{Q}^{r'}$ an r'-covering of $\mathcal{Q}$, $d_{\mathrm{PE}}(\mathcal{S},\mathcal{Q}) - r' \leq d_{\mathrm{PE}}(\mathcal{S}^r,\mathcal{Q}^{r'}) \leq d_{\mathrm{PE}}(\mathcal{S},\mathcal{Q}) + r$.
4. [illegible] Given $\mathcal{Q}^{r'}$ an r'-covering of $\mathcal{Q}$, $d_{\mathrm{PE}}(\mathcal{Q}^{r'},\mathcal{Q}) = 0$.

First, we see that, though not a metric in the strict sense, d_{PE} satisfies some important properties of a metric: isometry invariance and an asymmetric version of the triangle inequality. Second, by virtue of property 4 (unlike the corresponding property $d_{\mathrm{GH}}(\mathcal{Q},\mathcal{Q}^r) \leq r$ of the Gromov–Hausdorff distance, as mentioned above) d_{PE} can be used for partial matching. Third, note that a particular case of property 3 is $d_{\mathrm{PE}}(\mathcal{S},\mathcal{Q}) - r \leq d_{\mathrm{PE}}(\mathcal{S},\mathcal{Q}^r) \leq d_{\mathrm{PE}}(\mathcal{S},\mathcal{Q})$. When r is too large, the lower bound makes little sense, and one can obtain $d_{\mathrm{PE}}(\mathcal{S},\mathcal{Q}^r) \approx 0$ while $d_{\mathrm{PE}}(\mathcal{S},\mathcal{Q})$ is large. This reveals a potential danger in partial matching: when the patch $\mathcal{Q}^r$ is too small, such a comparison is liable to be meaningless, since a sufficiently small patch can be embedded with low metric distortion into every surface.

3. Efficient computation of the partial embedding and the Gromov–Hausdorff distances. We now address a practical question of efficiently computing the Gromov–Hausdorff and the partial embedding distances between two polyhedral surfaces. For clarity, we start with the computation of the simpler $d_{\mathrm{PE}}(\mathcal{S},\mathcal{Q})$, of which the computation of $d_{\mathrm{GH}}(\mathcal{S},\mathcal{Q})$ will be a straightforward generalization. As input, let us be given the following:

1. A discretization of the continuous surface $\mathcal{Q}$, defined by the point cloud $\mathcal{Q}_M^r = \{q_i\}_{i=1}^M$.
2. A polyhedral approximation of the continuous surface $\mathcal{S}$ defined by the point cloud $\mathcal{S}_N^r = \{s_i\}_{i=1}^N$ and a triangulation $T_\mathcal{S} = \{(m_{11}, m_{12}, m_{13}), \ldots, (m_{N_T,1}, m_{N_T,2}, m_{N_T,3})\}$. In our notation, the polyhedron $\mathcal{S}_N^r$ consists of N_T triangular faces, where the tth face is described by the vertices $s_{m_{t1}}, s_{m_{t2}}, s_{m_{t3}}$.
3. Two distance matrices $\mathbf{D}_\mathcal{S} = (d_\mathcal{S}(s_i, s_j))$ and $\mathbf{D}_\mathcal{Q} = (d_\mathcal{Q}(q_i, q_j))$, representing the geodesic distances between the samples of $\mathcal{S}$ and $\mathcal{Q}$, respectively. In practice, only an approximation of these distances can be computed numerically, e.g., using the fast marching method [24].

Using the triangulated surface $\mathcal{S}_N^r$ as a polyhedral r approximation of $\mathcal{S}$, we parameterize the points on $\mathcal{S}$ in local [illegible] derived from the triangulation. Let s be a point on $\mathcal{S}_N^r$ belonging to some triangle $t \in \{1, \ldots, N_T\}$; the Euclidean coordinates $\mathbf{s}$ of s can be represented as a convex combination of the triangle vertices $\mathbf{s}_{m_{t1}}, \mathbf{s}_{m_{t2}}, \mathbf{s}_{m_{t3}}$:

$$\mathbf{s} = \sum_{i=1}^{3} \mathbf{s}_{m_{ti}} u_i. \tag{10}$$

Hence, every point on $\mathcal{S}_N^r$ can be identified with a vector of barycentric coordinates $(t, \mathbf{u}) = (t, u_1, u_2, u_3)$, where $u_i \geq 0$, $u_1 + u_2 + u_3 = 1$. We will freely switch between s and its barycentric representation.

The partial embedding distance between $\mathcal{S}_N^r$ and $\mathcal{Q}_M^r$ can be formulated as the following constrained minimization problem:

$$\min_{s_i', \epsilon \geq 0} \epsilon \quad \text{s.t.} \quad \left| d_\mathcal{S}(s_i', s_j') - d_\mathcal{Q}(q_i, q_j) \right| \leq \epsilon, \tag{11}$$

where $(i, j) \in P \subseteq \{1, \ldots, M\} \times \{1, \ldots, M\}$ (for the time being we assume P to be given; a way to find it is shown in section 3.4). Once the minimal ϵ is found, our approximation to the partial embedding distance is $\hat{d}_{\mathrm{PE}}(\mathcal{S}_N^r, \mathcal{Q}_M^r) = \epsilon/2$.

Note that we use the images $s_i' = \psi(q_i)$ (in their barycentric coordinates) directly as the optimization variables, which frees us from the need to optimize over all possible permutations of s_i and q_j. In fact, the problem above is a modest-sized constrained minimization problem with $M + 1$ variables and at most $\frac{1}{2}M(M-1)$ inequality constraints. Also note that the points s_i are given, and so are the distances $d_\mathcal{Q}(q_i, q_j)$, which are the elements of the matrix $\mathbf{D}_\mathcal{Q}$. On the other hand, the distances $d_\mathcal{S}(s_i', s_j')$ have to be approximated numerically (see section 3.3); in the meantime, we assume them to be computable exactly. The reason is that the images s_i' [illegible] with the vertices of $\mathcal{S}_N^r$, respectively, and can fall inside any triangle on the latter polyhedron.

Theoretically, our computation is accurate up to the sampling radius, i.e., $|\hat{d}_{\mathrm{PE}}(\mathcal{S}_N^r, \mathcal{Q}_M^r) - d_{\mathrm{PE}}(\mathcal{S}, \mathcal{Q})| \leq 2r$. If in addition we take into account that the distances $d_\mathcal{S}$ and $d_\mathcal{Q}$ are computed with accuracy δ, we have $|\hat{d}_{\mathrm{PE}}(\mathcal{S}_N^r, \mathcal{Q}_M^r) - d_{\mathrm{PE}}(\mathcal{S}, \mathcal{Q})| \leq 2(r + \delta)$. In practice, a finer grid is used to approximate the distances, and therefore $\delta \ll r$. Sampling the surfaces sufficiently densely (r sufficiently small), we can compute d_{PE} with any desired accuracy. In other words, our approach is numerically [illegible].[1]

[1] Note that since our optimization problem is nonconvex, no practical optimization algorithm guarantees convergence to the global minimum. Yet, in section 3.2 we show that, in practice, local convergence can be avoided using multiresolution methods.

Problem (11) can be viewed as a minimization of maximum distortion, or a minmax (ℓ_∞ norm) problem. Practical considerations may favor the choice of norms other than ℓ_∞. In order to handle them numerically, we abandon the constrained minimization formulation with the artificial variable ϵ, and minimize

$$F(s'_1,\ldots,s'_M)=\sum_{i,j} w_{ij}\eta\left(d_{\mathcal{S}}(s'_i,s'_j)-d_{\mathcal{Q}}(q_i,q_j)\right) \tag{12}$$

with respect to s'_i, where $w_{ij}=1$ for $(i,j)\in P$ and zero otherwise are weights representing the set of "consistent pairs" P, and $\eta(x)$ is some nonlinear function (e.g., the choices $\eta(x)=x^2$ and $\eta(x)=|x|$ yield the ℓ_2 and the ℓ_1 norms, respectively). Note the resemblance of F to the [illegible] used in the MDS problems.

Though the properties of $\operatorname{argmin}_{s'_i} F$ are dramatically different from those of d_{PE}, it is a good measure of surface similarity with most of the important properties of the true partial embedding distance, yet less susceptible to noise and numerical errors. We will henceforth denote the ℓ_p version of d_{PE} as d^p_{PE}.

The Gromov–Hausdorff distance is computed essentially in the same way as $\hat{d}_{\mathrm{PE}}$ by solving

$$\min_{s'_i,q'_k,\epsilon\geq 0}\epsilon \quad \text{s.t.} \quad \begin{aligned} &\left|d_{\mathcal{Q}}(q'_i,q'_j)-d_{\mathcal{S}}(s_i,s_j)\right|\leq\epsilon,\\ &\left|d_{\mathcal{S}}(s'_k,s'_l)-d_{\mathcal{Q}}(q_k,q_l)\right|\leq\epsilon,\\ &\left|d_{\mathcal{S}}(s_i,s'_k)-d_{\mathcal{Q}}(q_k,q'_i)\right|\leq\epsilon, \end{aligned} \tag{13}$$

where $j>i$, $l>k$. Here two additional terms standing for dis φ and $\operatorname{dis}(\varphi,\psi)$ are added. As in the case of $\hat{d}_{\mathrm{PE}}$, it might be advantageous to replace the ℓ_∞ norm with some more robust nonlinearity, yielding the stress

$$\begin{aligned} F=&\sum_{i,j=1}^{N}\eta\left(d_{\mathcal{Q}}(q'_i,q'_j)-d_{\mathcal{S}}(s_i,s_j)\right)+\sum_{k,l=1}^{M}\eta\left(d_{\mathcal{S}}(s'_k,s'_l)-d_{\mathcal{Q}}(q_k,q_l)\right)\\ &+\sum_{i,j}\eta\left(d_{\mathcal{S}}(s_i,s'_k)-d_{\mathcal{Q}}(q_k,q'_i)\right). \end{aligned} \tag{14}$$

3.1. Iterative minimization algorithm. We now present a practical numerical algorithm for computation of $\hat{d}_{\mathrm{PE}}$. For clarity, we will henceforth consider only the unconstrained minimization of (12); in order to handle the constrained problem (13), it is first converted to an unconstrained one using the [illegible] [1], which is then solved in an essentially similar way. The same technique can be used to compute the Gromov–Hausdorff distance.

Our goal is to bring $F(s'_1,\ldots,s'_M)$ to a (possibly local) minimum over $s'_i=(t_i,\mathbf{u}_i)$ by starting with some initial guess $s'^{(0)}_i=(t^{(0)}_i,\mathbf{u}^{(0)}_i)$ of the points and proceeding by iteratively updating their locations, producing a decreasing sequence of function values. Let $s'^{(k)}_i=(t^{(k)}_i,\mathbf{u}^{(k)}_i)$ be the optimization variables at kth iteration and let there be a set of directions $\mathbf{d}^{(k)}_i$ such that displacement of $\mathbf{u}^{(k)}_i$ along them by some step size $\alpha^{(k)}$ decreases the value of F (for convenience, we omit the iteration index k wherever possible). The simplest way to select the directions is $\mathbf{d}_i=-\nabla_{\mathbf{u}_i}F$, which yields a family of algorithms usually termed as [illegible] or [illegible]. There exist a variety of more efficient strategies for selecting $\mathbf{d}_i$, including conjugate gradients and quasi-Newton methods, just to name a few [1].

The step size α has to be chosen to guarantee a sufficient decrease of F. When constant step is used, there is generally a tradeoff between too small steps, which result in slow convergence, and too large steps, which are liable to increase the value of F. To provide guaranteed decrease of F, we adaptively select the step size at every iteration using the [illegible], which first sets $\alpha = \alpha_0$ and then successively reduces it by some factor $\beta \in (0,1)$ until

$$F(s'_1,\dots,s'_M) - F(s''_1,\dots,s''_M) \geq -\sigma\alpha \sum_i \mathbf{d}_i^{\mathrm{T}} \nabla_{\mathbf{u}_i} F(s'_1,\dots,s'_M), \tag{15}$$

where $\sigma \in (0,1)$ and s''_i represent the points s'_i displaced by α along $\mathbf{d}_i$. In our implementation, we used $\sigma = 0.3$, $\beta = 0.5$. The initial value α_0 is first selected to be large, and gradually refined at each iteration. A similar rule can be applied when the update is performed for a single point per iteration, yielding a block-coordinate descent algorithm.

Note that unlike common optimization problems, here, in order to displace s'_i, one cannot simply add $\alpha\mathbf{d}_i$ to $\mathbf{u}_i$, since the latter might leave the triangle t_i, thus invalidating the barycentric representation. Instead, the displacement is performed by following a polylinear path starting at $\mathbf{u}_i$, propagating along a straight line in the direction $\mathbf{d}_i$ until the first intersection with the triangle boundary, then proceeding along a line inside the triangle adjacent to the intersected edge, and so on until the total length of the path is α (see Figure 2).

The computation is carried out by the following algorithm, which accepts an initial point $s = (t, \mathbf{u})$, a direction $\mathbf{d}$, and a step size α and returns the displaced point $s' = (t', \mathbf{u}')$:

1. Compute $\mathbf{u}' = \mathbf{u} + \alpha\mathbf{d}$. If $\mathbf{u}'$ is inside the triangle t, return $s' = (t, \mathbf{u}')$ and stop.
2. Find $\alpha' < \alpha$, for which $\mathbf{u}' = \mathbf{u} + \alpha'\mathbf{d}$ lies on the boundary of t.
3. Otherwise, if there is no triangle adjacent to the edge on which $\mathbf{u}'$ lies, we can proceed no further, since the shape boundary is reached. Return $s' = (t, \mathbf{u}')$ and stop.
4. Otherwise, let t' be a triangle sharing the edge on which $\mathbf{u}'$ lies. Transform $\mathbf{u}'$ and $\mathbf{d}$ to the barycentric coordinates of t', replace α with $\alpha - \alpha'$ and $(t, \mathbf{u}) \leftarrow (t', \mathbf{u}')$, and go to step 1.

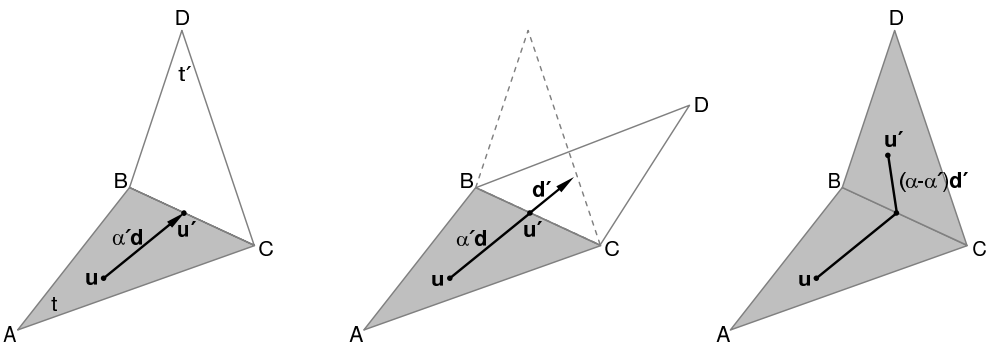


FIG. 2. *Displacement of the point* $\mathbf{u}$ *in the triangle* $t = ABC$ *along the direction* $\mathbf{d}$. *From left to right: the point is displaced along* $\mathbf{d}$ *until the triangle edge* BC *is reached; the adjacent triangle* $t' = BCD$ *is rotated about* BC *and the direction vector* $\mathbf{d}$ *is transformed to the barycentric coordinates of* t', $\mathbf{d}'$; *the final point* $\mathbf{u}'$ *is distant* α *from* $\mathbf{u}$ *along a polylinear path.*

The transformation of $\mathbf{u}'$ from barycentric coordinates in t to those in t' carried out in step 4 is trivial, since $\mathbf{u}'$ is located on the edge shared by the two triangles and thus has the same barycentric representation in t' up to vertex numbering. In order to transform $\mathbf{d}$, we rotate the triangle t' about the edge shared with t so that both triangles lie in the same plane, and we find such $\mathbf{d}'$ in the coordinate system of t', which defines the same planar direction as $\mathbf{d}$ in t.

Since at each iteration the Armijo rule produces a decreasing sequence of step sizes, it is sufficient to compute the displacement of $s_i'^{(k)}$ by $\alpha_0\mathbf{d}_i$ and cache the entire path; this allows us to save path computations for subsequent smaller steps.

3.2. Multiresolution minimization. GMDS, similarly to MDS algorithms in general [3], is liable to local convergence, since the objective function involved in the computation of $\hat{d}_{\mathrm{GH}}$ and $\hat{d}_{\mathrm{PE}}$ is nonconvex. Here we propose multiresolution optimization schemes as a practical remedy for this problem, in light of the results obtained in [9] for traditional MDS. An additional advantage of multiresolution optimization is a significant speedup of convergence.

The key idea of a multiresolution optimization scheme is to work with a hierarchy of problems, starting from a coarse version of the problem containing a small number of variables (points). The coarse level solution is interpolated to the next resolution level and is used as an initialization for the optimization at that level. The process is repeated until the finest level solution is obtained. Such a multiresolution scheme can be thought of as a smart way of initializing the optimization problem. Small local minima tend to disappear at coarse resolution levels, thus reducing the risk of local convergence, which is more probable when working at a single resolution.

The main components of a multiresolution scheme are the hierarchy of data which defines optimization problems at different resolution levels and the interpolation procedure, which allows to pass from coarse level to a finer one. Formally, let us denote by $\mathcal{Q}_{M_1} \subset \mathcal{Q}_{M_2} \subset \cdots \subset \mathcal{Q}_{M_L} = \mathcal{Q}_M$ an L-level hierarchy of data. The points on the $(l-1)$st resolution level are obtained by removing part of the points on the lth level. The corresponding distance matrices $\mathbf{D}_1, \ldots, \mathbf{D}_L = \mathbf{D}_{\mathcal{Q}}$ are created as submatrices of $\mathbf{D}_{\mathcal{Q}}$.

One possible means of constructing such a hierarchy is the farthest point sampling (FPS) strategy [14]. At the coarsest resolution level $\mathcal{Q}_{M_1}$, we select M_1 points. In practical problems, a few landmark points can be selected on the surface (e.g., in the face recognition problem, it is usually easy to roughly locate points such as the nose and the eyes); otherwise, these points can be chosen randomly. At the next resolution level, we add points in the following manner: q_{M_1+1} is selected as the most distant point from $\mathcal{Q}_{M_1}$, and so on, $q_{M_1+k} = \mathrm{argmax}_{q_i \in \mathcal{Q}_M} d_{\mathcal{Q}_M}(q_i, \{q_1, \ldots, q_{M_1+k-1}\})$. Taking the first M_l points from the sequence produced in this manner, we obtain $\mathcal{Q}_{M_l}$ (see the example in Figure 3).

Let us assume that at the lth resolution level, $\mathcal{Q}_{M_l} = \{q_1, \ldots, q_{M_l}\}$ were embedded into $\mathcal{S}_N^r$ using the iterative minimization algorithm outlined in section 3.1. As a result, the set of images $\psi(\mathcal{Q}_{M_l}) = \{s_1', \ldots, s_{M_l}'\}$ on the polyhedron $\mathcal{S}_N^r$ was obtained. At the next resolution level, we have to embed a larger set $\mathcal{Q}_{M_{l+1}}$ into $\mathcal{S}_N^r$, solving the minimization problem for $\psi(\mathcal{Q}_{M_{l+1}}) = \{s_1', \ldots, s_{M_{l+1}}'\}$. The initialization for the first M_l points is readily available from the solution at the previous level. The initial locations for the remaining points s_i' for $i = M_l + 1, \ldots, M_{l+1}$ have to be interpolated.

It is reasonable to initialize s_i' with such a point on S_N^r that the geodesic distances from it to the points $s_1', \ldots, s_{M_l}'$ are as close as possible to the geodesic distances

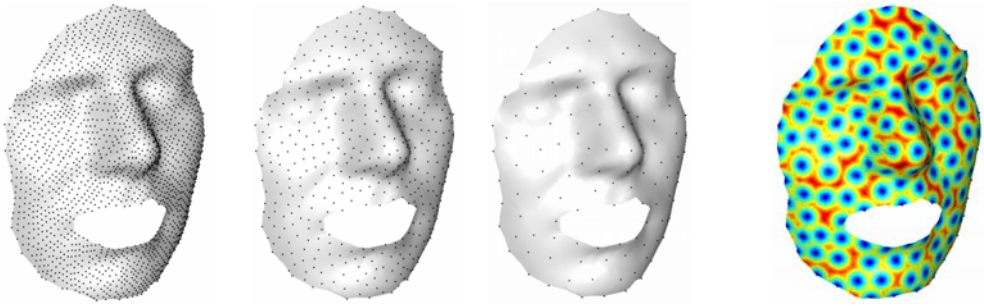


FIG. 3. *Example of data hierarchy construction for multiresolution optimization. Left: three resolution levels constructed using the farthest point sampling algorithm. Right: Geodesic distances from the coarsest grid points.*

from q_i to $q_1, \ldots, q_{M_l}$. Formally, s'_i can be expressed as

$$s'_i = \arg\min_{s} \sum_{j=1}^{M_l} \left(d_{\mathcal{S}}(s, s'_j) - d_{\mathcal{Q}}(q_i, q_j) \right)^2 . \tag{16}$$

Note that practically the minimum can be found by exhaustively searching over all $s_1, \ldots, s_N$ or even a coarser subset of $\mathcal{S}^r_N$. The complexity of such a search is negligible compared to the complexity of the iterative minimization process.

3.3. Interpolation of geodesic distances. So far, we considered the distance terms in (11) readily available, though it is obvious that only the distances between the surface samples, namely, $d_{\mathcal{S}}(s_i, s_j)$, are available, whereas distances between two arbitrary points in $\mathcal{S}^r_N$ have to be computed. Let us now devise a numerical procedure for computation of the geodesic distances between a pair of points on the polyhedron $\mathcal{S}^r_N$. As an input, we are given the following:

1. Euclidean coordinates $\{\mathbf{s}_i\}_{i=1}^N$ of the polyhedron vertices s_i.
2. The distance matrix $\mathbf{D}$, with elements $d_{ij} = d_{\mathcal{S}}(s_i, s_j)$.
3. Two arbitrary points $s = (t, \mathbf{u})$, $s' = (t', \mathbf{u}')$ in barycentric coordinates.

The output of our numeric procedure is the approximate geodesic distance $\hat{d}_{\mathcal{S}}(s, s') \approx d_{\mathcal{S}}(s, s')$.

Although the geodesic distance between two arbitrary points s, s' is generally unavailable, each point lies inside a triangle with vertices on the grid. Let s_1, s_2, s_3 and $s'_1, s\prime_2, s\prime_3$ be the vertices of the triangles t and t' enclosing s and s', respectively. Note that the mutual geodesic distances $d_{\mathcal{S}}(s_i, s'_j)$ are known. A naïve approach is to use the average of the above distances as $\hat{d}_{\mathcal{S}}(s, s')$, i.e.,

$$\hat{d}_{\mathcal{S}}(s, s') = \frac{1}{3} \sum_{i=1}^{3} d_{\mathcal{S}}(s_i, s'_i) . \tag{17}$$

By the triangle inequality, the approximation error has the order of the radius of the largest triangle, that is, $\mathcal{O}(r)$. In order to make the interpolant $\hat{d}_{\mathcal{S}}(s, s')$ smooth as a function $\mathbf{u}$ and $\mathbf{u}'$, the average is replaced by a weighted sum, where the weights are chosen according to proximity of $\mathbf{u}$ and $\mathbf{u}'$ to each of the enclosing triangle vertices. Such an interpolation procedure has very low computational complexity; yet its main disadvantage is the fact that it ignores the underlying geometry. Here we develop

an alternative geometric interpolation approach, which is a bit more elaborate, yet provides significantly more accurate approximation of the geodesic distances.

The basic idea of our interpolator is to compute first an approximation to the geodesic distances $d_{\mathcal{S}}(s, s'_i)$ between the point s and each of the three vertices s'_i of t'. Computation of each of these distances can be regarded as the problem of finding an approximation to $d_{\mathcal{S}}(s, s'_1)$ given $d_{\mathcal{S}}(s_i, s'_1)$. We handle the solution of this problem by a numerical procedure that we named the [illegible], which is detailed in Appendix B. Once the approximate $d_{\mathcal{S}}(s, s'_i)$ are available, we invoke again the three-point geodesic distance approximation, this time to compute the desired distance $d_{\mathcal{S}}(s, s')$ from $d_{\mathcal{S}}(s'_i, s)$.

3.4. Selection of weight for partial matching. So far, we have assumed that the set P of the consistent pairs of points in (11) or, alternatively, the weights w_{ij} in (12) are given. Here we show a way to find it in practice. Let $\mathcal{S}$ be the model surface and let $\mathcal{Q}$ be the probe surface. We denote by $\mathcal{Q}'$ a patch of $\mathcal{Q}$ and assume that its sampled version is $\mathcal{Q}'_{M'}$. The question is how to find the indices $(i, j) \in \{1, \ldots, M'\} \times \{1, \ldots, M'\}$ of pairs of points for which $d_{\mathcal{Q}'_{M'}}(q_i, q_j) \neq d_{\mathcal{Q}}(q_i, q_j)$, and thus must be excluded from P.

If a sampled version $\mathcal{Q}_M$ of $\mathcal{Q}$ is given (this is a common situation in face recognition, where one has the full probe surface but wishes to deliberately remove some parts of it [7]), the solution is straightforward: we remove the pairs of points (i, j) for which $|d_{\mathcal{Q}'_{M'}}(q_i, q_j) - d_{\mathcal{Q}_M}(q_i, q_j)| < \delta$, where δ is some small threshold, related to distance approximation accuracy. In the case when only the patch $\mathcal{Q}'$ is available, we note that the minimal geodesics may change due to the change in the boundary of $\mathcal{Q}$. We therefore remove pairs of points (i, j) for which $d_{\mathcal{Q}'}(q_i, \partial\mathcal{Q}') + d_{\mathcal{Q}'}(q_j, \partial\mathcal{Q}') < d_{\mathcal{Q}'}(q_i, q_j)$. Here $\partial\mathcal{Q}'$ denotes the boundary of $\mathcal{Q}'$. Since in practice a sampled version $\mathcal{Q}'_{M'}$ is given, this criterion must be discretized, and the distances approximated numerically.

4. Numerical results. We now present numerical experiments that assess the validity of our theoretical and computational framework. In the first experiment, we show the performance of the partial embedding distance in matching between spheres with different radii. In the second experiment, matching of articulated objects is performed using the Gromov–Hausdorff distance. The Elad–Kimmel canonical form distance [13] is used as the baseline reference in the two experiments. In the third experiment, we show matching of partially missing objects using the partial embedding distance. Additional numerical evaluations of the partial embedding distance can be found in [7, 8].

4.1. Partial embedding distance between spherical patches. In the first experiment, we measured the distance between a unit two-dimensional sphere sampled at 3200 points, and spheres with radii in the range $R = [0.5, 2.5]$ sampled at a smaller number of points according to the farthest point sampling strategy with a random seed. Ten random samplings were used for each radius. Two distance measures were compared: d^2_{PE} with $M = 100$ points and d_{CF} with $M = 100$, 250, and 500 points. d^2_{PE} was computed using the multiresolution minimization algorithm.

Figure 4 presents the normalized distances as a function of the sphere radius. In Figure 5, a close-up look is shown for radii in the range of $R = [0.88, 1.1]$. The gray areas represent the variance of the distance computed for the ten different samplings of the spheres. This variance makes spheres within a certain interval of radii around $R = 1$ indistinguishable by the distance function. The partial embedding distance appears to be extremely sensitive to the geometry of the surfaces. A change as small

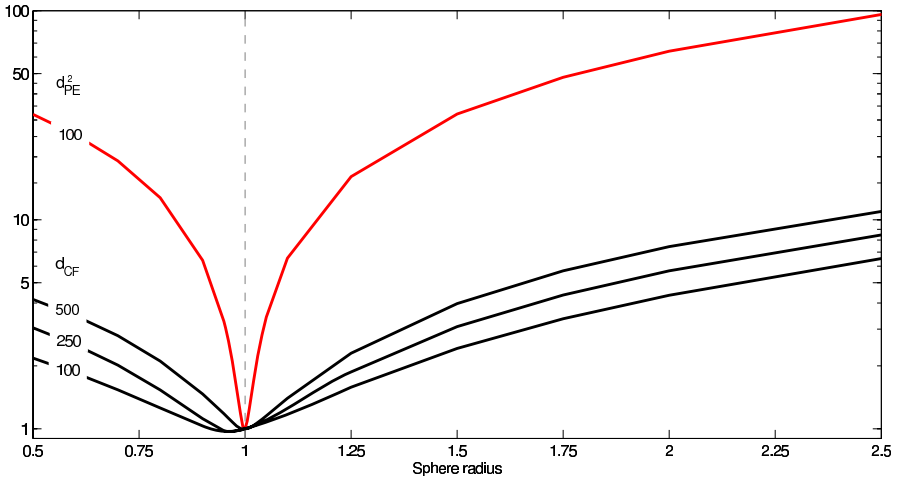


FIG. 4. *Normalized distance between a densely sampled unit sphere and spheres of different radii measured using the ℓ_2 approximation of the partial embedding distance (top line), and the canonical form distance (three bottom lines). The numbers on the plot stand for the number of points used for distance computation.*

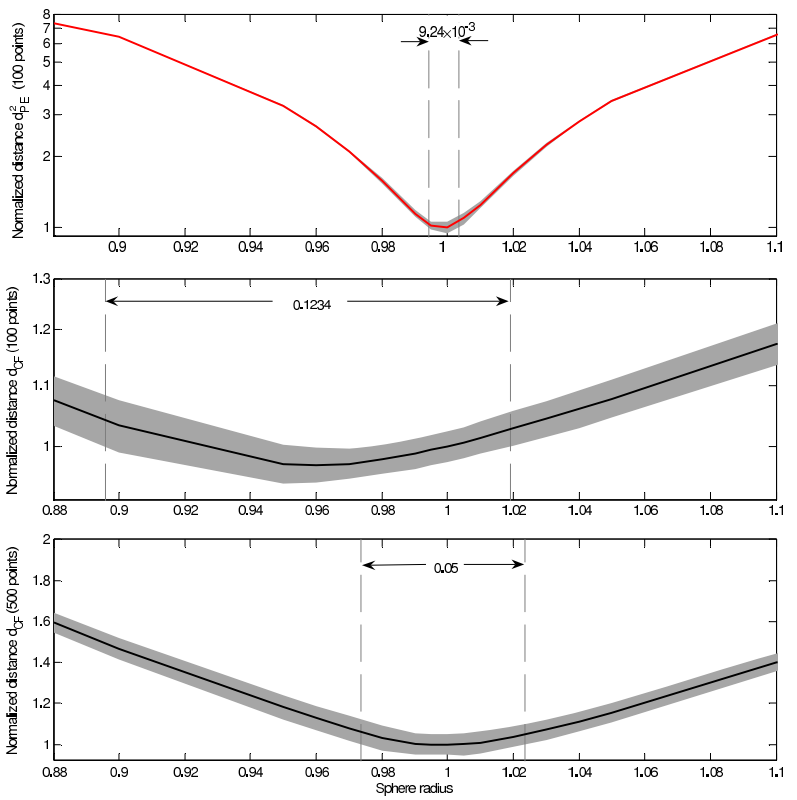


FIG. 5. *A close-up look at the normalized distance between a unit sphere and spheres of different radii measured using d^2_{PE} with 50 points (top) and d_{CF} with 100 and 500 points (middle and bottom plots). Gray areas denote the variance measured over ten random furthest point samplings. Intervals of sphere radii that fall below the sensitivity threshold of the distance measure are indicated.*

as 0.1% in the spherical patch radius (from 1.000 to 0.999) increases d^2_{PE} by a value exceeding the variance due to sampling. Similar results are achieved using d^1_{PE}, and with d_{PE} results are slightly inferior. For comparison, with the same number of points (100), d_{CF} is unable to discern between spheres differing in radius by more than 10%. Increasing the number of points to 500 makes d_{CF} sensitive to radius differences of about 2.64%, which is still one order of magnitude below the sensitivity of d^2_{PE}. From these results, d_{PE} appears to be a promising tool for very accurate comparison of surfaces.

Reasonable execution times were observed. The generalized stress function (12) and its gradient implemented in C are evaluated in about 40 msec on a Pentium IV PC for $M = 100$ points. The entire computation of d_{PE} using nonoptimized MATLAB and C code took about 10–60 sec.[2]

4.2. Gromov–Hausdorff distance between isometric surfaces. In the second experiment, we evaluated the performance of the Gromov–Hausdorff distance on six articulated objects and their approximate isometries. The ℓ_2 and ℓ_∞ approximations of d_{GH} were used. All surfaces were represented as triangulated polyhedral meshes (see Figure 6). The embedding spaces consisted of the full-density meshes, whereas the embedded meshes were sampled at $M = N = 50$ points. Optimization was performed using the multiresolution scheme. Geodesic distances were computed using the fast marching method for triangulated domains [19]. As a reference, the canonical form distance was computed between the surfaces by comparing a full-density canonical form to canonical forms sampled at $M = 100$, 250, and 600 points as well as to full-density canonical forms. Moment signatures and an iterative closest point algorithm were used to compare the canonical forms.

Figure 7 presents the distance matrices obtained using the ℓ_2 and ℓ_∞ approximations of d_{GH}. As few samples as 50 give perfect recognition of all the objects using both d^2_{GH} and d_{GH}. For comparison, the best result of d_{CF} with full-density canonical forms and ICP (Figure 7, bottom right) yielded about 11% equal error rate. Figure 8 depicts Euclidean representations of the dissimilarities between the objects measured using d^2_{GH} and d_{CF}. Note the tight clusters produced by d^2_{GH} compared to looser clusters obtained from d_{CF}. The dog and paper objects exhibit larger intracluster dissimilarities, which we attribute to larger deviations from isometricity.

In order to accurately quantify the discriminative power of the compared distances, we measured the [illegible] as

$$\sigma = \min_n \left\{ \frac{\min_{i,j,m\neq n} d(\mathcal{S}_{mi}, \mathcal{S}_{mj})}{\max_{i,j} d(\mathcal{S}_{ni}, \mathcal{S}_{nj})} \right\}, \tag{18}$$

where S_{ni} denotes the ith instance of the nth object. In simple words, σ expresses the worst-case ratio between the intercluster and the intracluster distances. The higher σ, the better the discrimination between the objects. d^2_{GH} yielded $\sigma = 3.47$, which is a manifest of excellent separation; slightly inferior results were exhibited by d_{GH}. For comparison, the best result obtained by d_{CF} was $\sigma = 0.58$.

Increasing the number of points from 100 to 600 in d_{CF} increases the computational complexity with small or no benefit in accuracy. We also noted that starting from about 250 points, the use of ICP in the computation of the canonical form distance gives negligible accuracy improvement compared to the faster moment signature comparison. This leads us to the conclusion that in this experiment, the discrimina-

[2]The codes and datasets used in this paper will be published at http://tosca.cs.technion.ac.il.

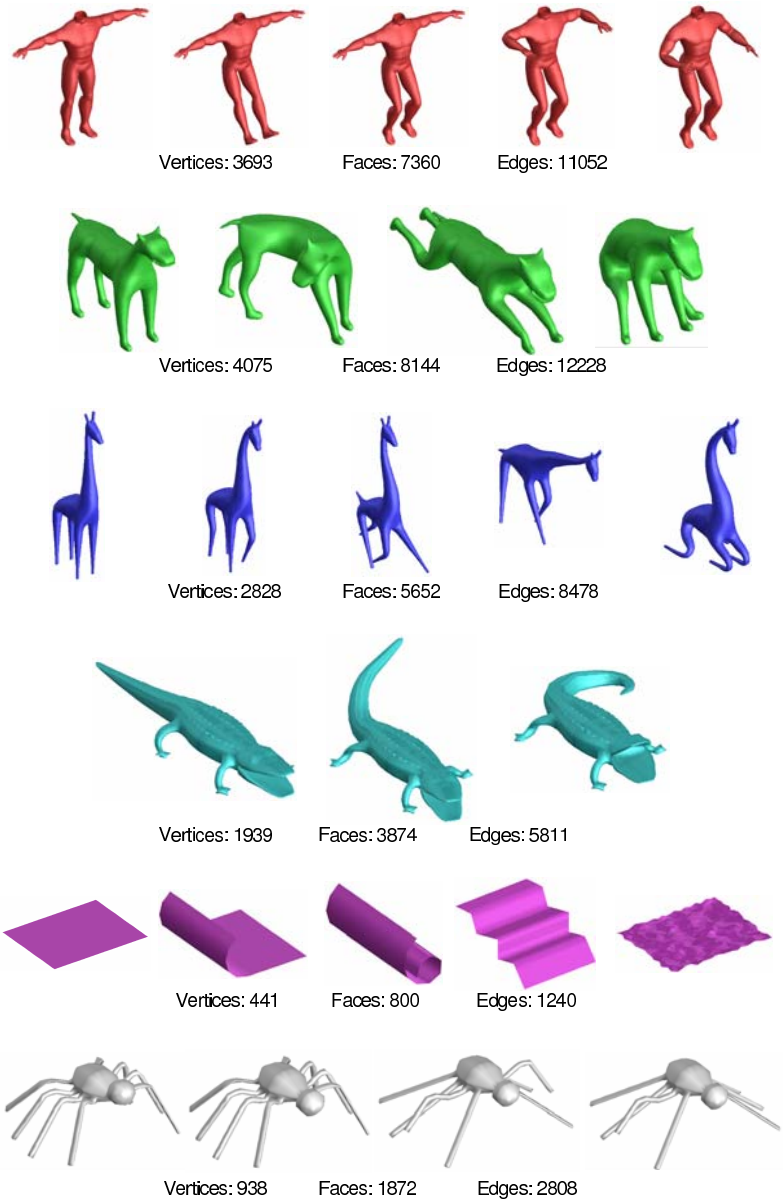


FIG. 6. *Six objects (human, dog, giraffe, crocodile, paper, and spider) with their approximate isometries.*

tive power of the canonical form distance is essentially limited by metric distortion introduced by embedding the surface into $\mathbb{R}^3$.

4.3. Partial matching of isometric surfaces. In the third experiment, six partial probes (Figure 9) were matched against the objects from the previous experiment using the ℓ_2 and ℓ_∞ approximations of d_{PE}. As before, the embedding spaces consisted of full-density meshes, whereas the partial probes were sampled at $M = N = 50$ points. Optimization was performed using the multiresolution scheme.

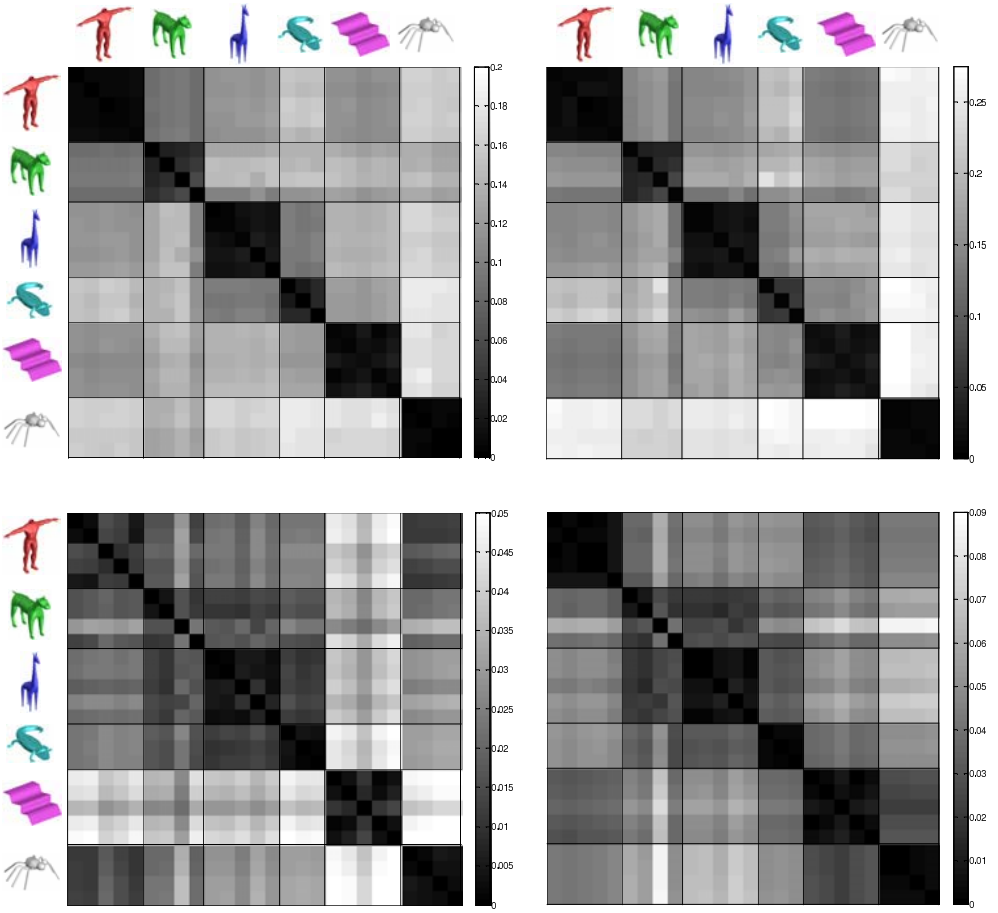


FIG. 7. *Matrices of distances (dissimilarities) between the objects obtained using* d^2_{GH} *(top left) and* d_{GH} *(top right) with* $M = N = 50$*; and* d_{CF} *with* $M = 600$ *(bottom left) and full-density canonical forms (bottom right).*

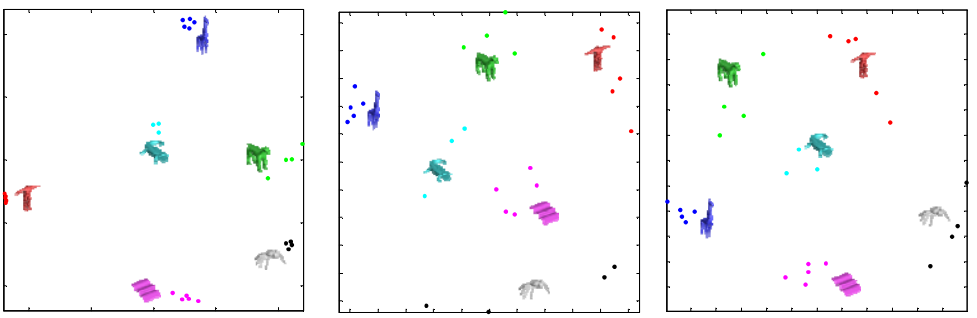


FIG. 8. *Euclidean proximity pattern representing the distances between different objects measured according to* d^2_{GH} *with* $M = N = 50$ *(left) and* d_{CF} *with* $M = 600$ *and full-density canonical forms (middle and right, respectively). Each point represents an object; Euclidean distance between two points represents dissimilarity between the corresponding objects.*

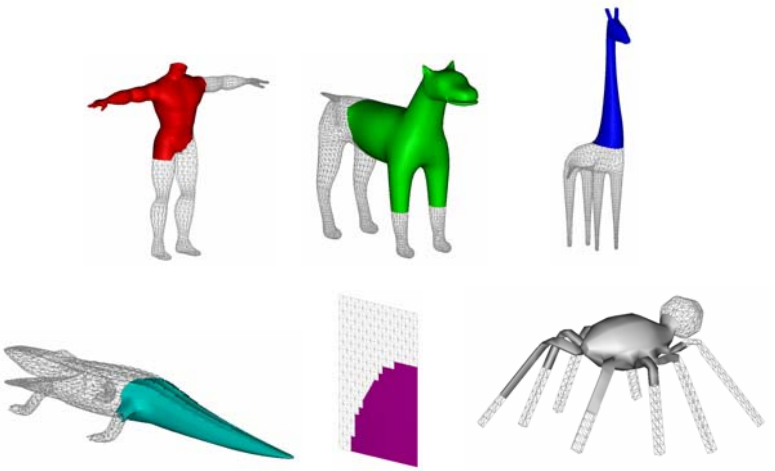


FIG. 9. *Parts of the human, dog, giraffe, crocodile, paper, and spider objects used as probes in the third experiment.*

Perfect matching was achieved by both distance functions, yielding the separability of $\sigma = 1.62$ and 1.29 for d^2_{PE} and d_{PE}, respectively.

Additional numerical assessment of the partial embedding distance was presented in [8], where it was used for face recognition from partially occluded data.

5. Conclusions. In this paper, we introduced a new computational framework for isometry-invariant surface matching. This framework allowed us to formulate the Gromov–Hausdorff distance between two surfaces as a solution of the generalized MDS problem. We also mentioned that the same numerical framework can be used to perform isometry-invariant matching of partially missing surfaces [7]. The GMDS problem is posed as a continuous optimization problem and is solved efficiently using the proposed iterative optimization algorithm. A multiresolution scheme can be a practical means of avoiding convergence to local minima. In light of recent results [9], we foresee that further performance boosting can be obtained using multigrid optimization methods.

Appendix A. Proofs of properties of the partial embedding distance.

1 (i) is trivial since $\operatorname{dis}\psi \geq q$.

The first direction of (ii) is straightforward, since if $\mathcal{Q}$ is isometrically embeddable into $\mathcal{S}$, there exists $\psi : \mathcal{Q} \mapsto \mathcal{S}$ with $\operatorname{dis}\psi = 0$. The second direction is more elaborate; our proof closely follows the proof of Theorem 7.3.30 in [10]. From $d_{\mathrm{PE}}(\mathcal{S}, \mathcal{Q}) = 0$ it follows that there exists a sequence of mappings $\{\psi_n : \mathcal{Q} \mapsto \mathcal{S}\}$ satisfying $\operatorname{dis}\psi_n \leq \frac{1}{n}$. Fix a countable dense set $\mathcal{Q}' \subset \mathcal{Q}$. Using the Cantor diagonalization procedure, one can choose a subsequence $\{\psi_{n_k}\}$, such that for every $q' \in \mathcal{Q}'$ the sequence $\{\psi_{n_k}(q')\}$ converges in $\mathcal{S}$. Let us assume without loss of generality that such a sequence is $\{\psi_n\}$ itself. We define a map $\psi : \mathcal{Q}' \mapsto \mathcal{S}$ as the pointwise limit of $\{\psi_n\}$, namely, $\psi(q') = \lim_{n\to\infty} \psi_n(q')$ for every $q' \in \mathcal{Q}'$. Since $|d_{\mathcal{S}}(\psi_n(q''), \psi_n(q')) - d_{\mathcal{Q}}(q'', q')| \leq \operatorname{dis}\psi_n \to 0$, we have

$$d_{\mathcal{S}}(\psi(q''), \psi(q')) = \lim_{n\to\infty} d_{\mathcal{S}}(\psi_n(q''), \psi_n(q')) = d_{\mathcal{Q}}(q'', q')$$

for all $q', q'' \in \mathcal{Q}'$, i.e., ψ is a distance preserving embedding of $\mathcal{Q}'$ into $\mathcal{S}$. Since

$\mathcal{Q}'$ is dense, by Proposition 1.5.9 in [10], ψ can be extended to a distance preserving embedding of the entire $\mathcal{Q}$ into $\mathcal{S}$.

To prove (iii), let $d_{\mathrm{PE}}(\mathcal{R},\mathcal{Q}) = \delta_1$ and $d_{\mathrm{PE}}(\mathcal{S},\mathcal{R}) = \delta_2$. Then there exist two sequences of mappings $\{\varphi_n : \mathcal{Q} \mapsto \mathcal{R}\}$ and $\{\psi_n : \mathcal{R} \mapsto \mathcal{S}\}$ satisfying $\operatorname{dis}\varphi_n < 2\delta_1 + \frac{1}{n}$ and $\operatorname{dis}\psi_n < 2\delta_2 + \frac{1}{n}$. Denote by $\{\zeta_n : \mathcal{Q} \mapsto \mathcal{S}\}$ the sequence of the compositions $\zeta_n = \psi_n \circ \varphi_n$. Invoking the triangle inequality for real numbers, one has

$$\begin{aligned}|d_{\mathcal{S}}(\zeta_n(q),\zeta_n(q')) - d_{\mathcal{Q}}(q,q')| &\le |d_{\mathcal{S}}(\zeta_n(q),\zeta_n(q')) - d_{\mathcal{R}}(\varphi_n(q),\varphi_n(q'))| \\ &\quad + |d_{\mathcal{R}}(\varphi_n(q),\varphi_n(q')) - d_{\mathcal{Q}}(q,q')| \\ &\le \operatorname{dis}\varphi_n + \operatorname{dis}\psi_n < 2(\delta_1+\delta_2) + \frac{2}{n}\end{aligned}$$

for all $q, q' \in \mathcal{Q}'$. Hence, $\operatorname{dis}\zeta_n \le 2(\delta_1+\delta_2) + \frac{2}{n}$ and $d_{\mathrm{PE}}(\mathcal{S},\mathcal{Q}) \le \delta_1 + \delta_2$. □

Proof of property 2 (i) This property follows straightforwardly from the fact that $d_{\mathrm{PE}}(\mathcal{S},\mathcal{Q}) \le d_{\mathrm{GH}}(\mathcal{Q},\mathcal{S})$. □

Proof of property 3 We divide the proof into the following two separate statements:

(i) $d_{\mathrm{PE}}(\mathcal{S},\mathcal{Q}) - r \le d_{\mathrm{PE}}(\mathcal{S},\mathcal{Q}^r) \le d_{\mathrm{PE}}(\mathcal{S},\mathcal{Q})$.
(ii) $d_{\mathrm{PE}}(\mathcal{S},\mathcal{Q}) \le d_{\mathrm{PE}}(\mathcal{S}^{r'},\mathcal{Q}) \le d_{\mathrm{PE}}(\mathcal{S},\mathcal{Q}) + r'$.

The combination of (i) and (ii) gives property 3.

The right side of inequality (i) is straightforward, since $\sup_{q,q'\in\mathcal{Q}} |d_{\mathcal{Q}^r}(q,q') - d_{\mathcal{S}}(\psi(q),\psi(q'))| \le \sup_{q,q'\in\mathcal{Q}} |d_{\mathcal{Q}}(q,q') - d_{\mathcal{S}}(\psi(q),\psi(q'))|$. To show the left side, we invoke property 1(ii), which yields $d_{\mathrm{PE}}(\mathcal{S},\mathcal{Q}) \le d_{\mathrm{PE}}(\mathcal{S},\mathcal{Q}^r) + d_{\mathrm{PE}}(\mathcal{Q}^r,\mathcal{Q})$. It remains to show that $d_{\mathrm{PE}}(\mathcal{Q}^r,\mathcal{Q}) \le r$. Let the mapping $\psi : \mathcal{Q} \mapsto \mathcal{Q}^r$ be defined in the following way: for $q \in \mathcal{Q}^r$, $\psi(q) = q$; and for $q \in \mathcal{Q}\backslash\mathcal{Q}^r$, $\psi(q) = \arg\min_{q'\in\mathcal{Q}^r} d_{\mathcal{Q}}(q,q')$ (the minimum exists, since $\mathcal{Q}^r$ can be replaced by its finite subcovering). From the fact that $\mathcal{Q}^r$ is an r-covering, it follows that $d_{\mathcal{Q}}(q,\psi(q)) \le r$. Let q_1, q_2 be two points in $\mathcal{Q}$; if both are in $\mathcal{Q}^r$, then $|d_{\mathcal{Q}}(q_1,q_2) - d_{\mathcal{Q}}(\psi(q_1),\psi(q_2))| = 0$. If both q_1 and q_2 are in $\mathcal{Q}^r$, invoking the triangle inequality, one has $|d_{\mathcal{Q}}(q_1,q_2) - d_{\mathcal{Q}}(\psi(q_1),\psi(q_2))| \le d_{\mathcal{Q}}(q_1,\psi(q_1)) + d_{\mathcal{Q}}(q_2,\psi(q_2)) \le 2r$. Similarly, when only q_1 is in $\mathcal{Q}^r$, $|d_{\mathcal{Q}}(q_1,q_2) - d_{\mathcal{Q}}(\psi(q_1),\psi(q_2))| \le r$. Hence, $\operatorname{dis}\psi \le 2r$.

The left side of inequality (ii) is straightforward. To show the right side, let us be given some $\psi : \mathcal{Q} \to \mathcal{S}$. We define a new mapping $\tilde{\psi} : \mathcal{Q} \to \mathcal{S}^{r'}$ in the following way:

$$\tilde{\psi}(q) = \begin{cases} \psi(q), & \psi(q) \in \mathcal{S}^{r'}, \\ \operatorname{argmin}_{s\in\mathcal{S}^{r'}} d_{\mathcal{S}}(s,\psi(q)) & \text{else.} \end{cases}$$

Since $\mathcal{S}^{r'}$ is an r'-covering of $\mathcal{S}$, $\sup_{q,q'\in\mathcal{Q}} |d_{\mathcal{Q}}(q,q') - d_{\mathcal{S}}(\tilde{\psi}(q),\tilde{\psi}(q'))| \le \sup_{q,q'\in\mathcal{Q}} |d_{\mathcal{Q}}(q,q') - d_{\mathcal{S}}(\psi(q),\psi(q'))| + 2r'$ for all ψ, and consequently $d_{\mathrm{PE}}(\mathcal{S}^{r'},\mathcal{Q}) \le d_{\mathrm{PE}}(\mathcal{S},\mathcal{Q}) + r'$. □

Proof of property 4 The right side of the inequality follows from the combination of properties 3 and 1(ii), $d_{\mathrm{PE}}(\mathcal{S}_M^{r'},\mathcal{Q}_N^r) \le d_{\mathrm{PE}}(\mathcal{S}_M^{r'},\mathcal{Q}) + d_{\mathrm{PE}}(\mathcal{Q},\mathcal{Q}_N^r) \le d_{\mathrm{PE}}(\mathcal{S},\mathcal{Q}) + r'$. The left side of the inequality follows from property 3, $d_{\mathrm{PE}}(\mathcal{S},\mathcal{Q}) \le d_{\mathrm{PE}}(\mathcal{S},\mathcal{Q}_N^r) + r \le d_{\mathrm{PE}}(\mathcal{S}_M^{r'},\mathcal{Q}_N^r) + r$. □

Appendix B. Three-point geodesic distance approximation. We now outline a numerical scheme for interpolation of geodesic distance $d_{\mathcal{S}}(s,s')$ between two arbitrary points $s = (t,\mathbf{u})$ and $s' = (t',\mathbf{u}')$ on a polyhedral surface $\mathcal{S}_N$ from the matrix $\mathbf{D}_{\mathcal{S}}$ of geodesic distances between all s_i and s_j. Let us denote the vertices of

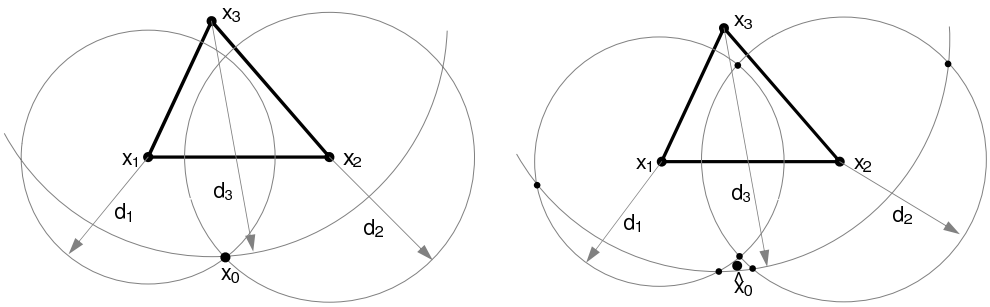


FIG. 10. *Geodesic distances associated with the triangle vertices* $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$ *are used to find a virtual "source" point, distant* d_1, d_2, *and* d_3 *from the corresponding vertices. When the surface is a plane, the three circles intersect at one point* $\mathbf{x}_0$ *and such a source exists (left); in the general case, only an approximate source* $\hat{\mathbf{x}}_0$ *can be found, yet every pair of circles usually intersects at two distinct points (right).*

the triangle t, in which the point s lies, as $s_1, s_2, s_3 \in \mathcal{S}_N$. Similarly, the vertices of t' will be denoted by s'_1, s'_2, s'_3.

As a starting point, let us consider the particular case where s' is one of the polyhedron vertices. In this case, the distances $d_j = d_{\mathcal{S}}(s_i, s')$ for $i = 1, 2, 3$ can be obtained from the matrix $\mathbf{D}_{\mathcal{S}}$. We will assume without loss of generality that the triangle s_1, s_2, s_3 lies in $\mathbb{R}^2$ with s_1 at the origin and s_2 on the horizontal axis, denoting by $\mathbf{x}_1 = (0,0)^{\mathrm{T}}$, $\mathbf{x}_2 = (x_2, 0)^{\mathrm{T}}$, and $\mathbf{x}_3 = (x_3, y_3)^{\mathrm{T}}$ the planar coordinates of the corresponding vertices. The original triangle can always be brought to this coordinate system by a simple precomputed transformation. In other words, the actual vertex coordinates $\mathbf{s}_i$ in a triangle are not required by the algorithm.

Let us assume that there exists a (virtual) source point $\hat{\mathbf{x}}_0 = (\hat{x}_0, \hat{y}_0)^{\mathrm{T}}$ in the plane of the triangle, whose distance from $\mathbf{x}_i$ is d_i. When the surface $\mathcal{S}$ is planar, there indeed exists such a point consistent with the data, and it can be found by computing the intersection of three circles of radii d_i centered at the corresponding triangle vertices $\mathbf{x}_i$ (Figure 10, left). This gives rise to the following system of quadratic equations:

$$\begin{aligned} \hat{x}_0^2 + \hat{y}_0^2 &= d_1^2, \\ (\hat{x}_0 - x_2)^2 + \hat{y}_0^2 &= d_2^2, \\ (\hat{x}_0 - x_3)^2 + (\hat{y}_0 - y_3)^2 &= d_3^2. \end{aligned} \tag{19}$$

Once $\hat{\mathbf{x}}_0$ is determined, the Euclidean distance $\|\hat{\mathbf{x}}_0 - \mathbf{x}\|$ serves as the approximation to the geodesic distance $d_{\mathcal{S}}(\mathbf{x}_0, \mathbf{x})$. In the above particular case, the two coincide.

However, in the general case when the surface has nonzero Gaussian curvature, the Euclidean distances $\|\hat{\mathbf{x}}_0 - \mathbf{x}_i\|$ will be inconsistent with the geodesic distances d_i, and thus the three circles will not generally intersect at a single point (Figure 10, right). Even if $\hat{\mathbf{x}}_0$ can be found to satisfy (19) in the "best possible way," the Euclidean distance $\|\hat{\mathbf{x}}_0 - \mathbf{x}\|$ will not generally coincide with the true geodesic distance $d_{\mathcal{S}}(\mathbf{x}, \mathbf{x}_0)$. Yet this scheme still gives a good approximation to $d_{\mathcal{S}}(\mathbf{x}, \mathbf{x}_0)$.[3]

[3]Theoretically, the scheme is first order, but in practice we observed that its accuracy is often superior by about two orders of magnitude compared to the straightforward "nongeometric" linear interpolation of geodesic distances on articulated objects.

Here, we find the virtual source $\hat{\mathbf{x}}_0$ by a weighted least-squares solution of (19), namely,

$$\hat{\mathbf{x}}_0\left(\mathbf{X},\mathbf{d},\mathbf{w}\right)=\arg\min_{\mathbf{x}_0}\ell\left(\mathbf{x}_0;\mathbf{X},\mathbf{d},\mathbf{w}\right), \tag{20}$$

where

$$\ell(\mathbf{x}_0;\mathbf{X},\mathbf{d},\mathbf{w})=\sum_{i=1}^{3}w_i\big(\left\|\mathbf{x}_0-\mathbf{x}_i\right\|^2-d_i^2\big)^2, \tag{21}$$

$\mathbf{X}=(\mathbf{x}_1,\mathbf{x}_2,\mathbf{x}_3)$, $\mathbf{d}=(d_1^2,d_2^2,d_3^2)^{\mathrm{T}}$, and $\mathbf{w}=(w_1,w_2,w_3)^{\mathrm{T}}$ is a vector of weights controlling the influence of each of the vertices. For a moment, we assume that the weights are given; later we will discuss their choice.

In our implementation, the minimization is carried out using Newton steps [1]. The gradient $\mathbf{g}$ and the Hessian matrix $\mathbf{H}$ of $\ell(\mathbf{x}_0;\mathbf{X},\mathbf{d},\mathbf{w})$ with respect to $\mathbf{x}_0$, required by the Newton method, are given by

$$\mathbf{g}=4\left(\mathbf{w}^{\mathrm{T}}\mathbf{f}\cdot\mathbf{x}-\mathbf{X}\cdot\mathbf{w}\odot\mathbf{f}\right), \tag{22}$$

$$\mathbf{H}=4\ \mathbf{w}^{\mathrm{T}}\mathbf{f}\cdot\mathbf{I}+8\sum_{i=1}^{3}w_i\left(\mathbf{x}_0-\mathbf{x}_i\right)\left(\mathbf{x}_0-\mathbf{x}_i\right)^{\mathrm{T}}, \tag{23}$$

where $(\mathbf{f})_i=\left\|\mathbf{x}_0-\mathbf{x}_i\right\|^2-d_i^2$, and $\mathbf{w}\odot\mathbf{f}$ denotes the elementwise (Hadamard) product. When the vertices $\mathbf{x}_i$ are not collinear, which normally happens in a valid triangulation, the Hessian is full rank for $w_i\neq 0$. When one of the weights, say w_3, is zero, the first two equations in (19) are satisfied and hence the first terms of $\mathbf{f}$ vanish. Consequently, the first term in the Hessian is zero, whereas the second term is a sum of two outer products of linearly independent vectors, which is full rank. When only one weight, say w_1, is nonzero, the Hessian assumes the form $\mathbf{H}=8w_1\left(\mathbf{x}_0-\mathbf{x}_1\right)\left(\mathbf{x}_0-\mathbf{x}_1\right)^{\mathrm{T}}$, which is a rank one matrix. Since in this case the solution $\hat{\mathbf{x}}_0$ admits only the first equation in (19), the minimum of $\ell(\mathbf{x}_0;\mathbf{X},\mathbf{d},\mathbf{w})$ is achieved on the circle described by the said equation, and the null vector of $\mathbf{H}$ corresponds to the tangential direction, in which the function does not change.

Due to the lack of convexity of $\ell(\mathbf{x}_0;\mathbf{X},\mathbf{d},\mathbf{w})$, we need to pay attention to the initialization. In our implementation, we first find simultaneous solutions of different pairs of equations from (19), giving at most six different points, and then initialize the minimization algorithm by the one yielding the smallest value of ℓ. The solution of a pair of quadratic equations has an analytic expression; for example, the first two equations yield

$$\hat{x}_0=\frac{d_1^2-d_2^2+x_2^2}{2x_2},$$

$$\hat{y}_0=\pm\sqrt{d_1^2-\hat{x}_0^2} \tag{24}$$

wherever two, one, or none of the solutions exist. Since by its virtue such an initialization is sufficiently close to the optimal $\hat{\mathbf{x}}_0$, the algorithm converges in about two or three iterations.

LEMMA B.1. $\mathbf{x}_i$ $\mathbf{w}$ $\hat{\mathbf{x}}_0(\mathbf{X},\mathbf{d},\mathbf{w})$ $\mathbf{d}$ $\mathbf{w}$

$$\mathbf{J}_{\mathbf{w}} = \nabla_{\mathbf{w}}\hat{\mathbf{x}}_0^{\mathrm{T}} = -\mathbf{G}_{\mathbf{w}}\mathbf{H}^{-1},$$

$$\mathbf{J}_{\mathbf{d}} = \nabla_{\mathbf{d}}\hat{\mathbf{x}}_0^{\mathrm{T}} = -\mathbf{G}_{\mathbf{d}}\mathbf{H}^{-1}, \tag{25}$$

$\mathbf{G}_{\mathbf{w}} = \nabla_{\mathbf{w}}\mathbf{g}^{\mathrm{T}}$ $\mathbf{G}_{\mathbf{d}} = \nabla_{\mathbf{d}}\mathbf{g}^{\mathrm{T}}$

$$\mathbf{G}_{\mathbf{w}} = 4\mathbf{F}\left(\mathbf{x}_0\mathbf{1}^{\mathrm{T}} - \mathbf{X}\right),$$

$$\mathbf{G}_{\mathbf{d}} = -4\mathbf{W}\left(\mathbf{x}_0\mathbf{1}^{\mathrm{T}} - \mathbf{X}\right). \tag{26}$$

$\mathbf{W}$ $\mathbf{F}$ $\mathbf{w}$ $\mathbf{f}$ $\mathbf{1} = (1,1,1)^{\mathrm{T}}$. Let us start with the derivative with respect to $\mathbf{w}$. By (20), $\hat{\mathbf{x}}_0$ is a local minimum of $\ell(\mathbf{x}_0;\mathbf{X},\mathbf{d},\mathbf{w})$, hence $\mathbf{g}(\hat{\mathbf{x}}_0;\mathbf{X},\mathbf{d},\mathbf{w}) = 0$, where $\mathbf{g}$ stands for the gradient $\nabla_{\mathbf{x}_0}\ell$. Adding an infinitesimal perturbation to $\mathbf{w}$ results in $\mathbf{g}(\hat{\mathbf{x}}_0;\mathbf{X},\mathbf{d},\mathbf{w}+d\mathbf{w}) \neq 0$, yet there exists an infinitesimal perturbation $d\hat{\mathbf{x}}_0$, such that $\mathbf{g}(\hat{\mathbf{x}}_0 + d\hat{\mathbf{x}}_0;\mathbf{X},\mathbf{d},\mathbf{w}+d\mathbf{w}) = 0$. From the first order Taylor expansion,

$$\begin{aligned}\mathbf{g}(\hat{\mathbf{x}}_0 + d\hat{\mathbf{x}}_0;\mathbf{X},\mathbf{d},\mathbf{w}+d\mathbf{w}) &\approx \mathbf{g}(\hat{\mathbf{x}}_0;\mathbf{X},\mathbf{d},\mathbf{w}) \\ &\quad + \mathbf{H}(\hat{\mathbf{x}}_0;\mathbf{X},\mathbf{d},\mathbf{w})\,d\hat{\mathbf{x}}_0 + \mathbf{G}_{\mathbf{w}}^{\mathrm{T}}(\hat{\mathbf{x}}_0;\mathbf{X},\mathbf{d},\mathbf{w})\,d\mathbf{w},\end{aligned} \tag{27}$$

where $\mathbf{H}$ stands for the Hessian matrix of ℓ with respect to $\mathbf{x}_0$ given in (23). Substituting $\mathbf{g}(\hat{\mathbf{x}}_0;\mathbf{X},\mathbf{d},\mathbf{w}) = 0$ and neglecting second order terms, we obtain

$$\mathbf{H}d\hat{\mathbf{x}}_0 + \mathbf{G}_{\mathbf{w}}^{\mathrm{T}}d\mathbf{w} = 0. \tag{28}$$

Since for noncollinear $\mathbf{x}_i$ and at most one zero weight the Hessian is invertible,

$$d\hat{\mathbf{x}}_0 = -\mathbf{H}^{-1}\mathbf{G}_{\mathbf{w}}^{\mathrm{T}}d\mathbf{w}, \tag{29}$$

from which $\mathbf{J}_{\mathbf{w}}$ is straightforwardly obtained. In the same way, $\mathbf{J}_{\mathbf{d}}$ is derived. □

Having the approximant of the virtual source $\hat{\mathbf{x}}_0(\mathbf{X},\mathbf{d},\mathbf{w})$ and its first order derivatives, we can construct the three-point geodesic distance approximant,

$$\hat{d}_{\mathcal{S}}^2(\mathbf{x};\mathbf{X},\mathbf{d},\mathbf{w}) = \left\|\mathbf{x} - \hat{\mathbf{x}}_0(\mathbf{X},\mathbf{d},\mathbf{w})\right\|^2, \tag{30}$$

and it derivatives,

$$\begin{aligned}\nabla_{\mathbf{x}}\hat{d}_{\mathcal{S}}^2(\mathbf{x};\mathbf{X},\mathbf{d},\mathbf{w}) &= 2\left(\mathbf{x} - \hat{\mathbf{x}}_0\right), \\ \nabla_{\mathbf{w}}\hat{d}_{\mathcal{S}}^2(\mathbf{x};\mathbf{X},\mathbf{d},\mathbf{w}) &= 2\mathbf{J}_{\mathbf{w}}\left(\hat{\mathbf{x}}_0 - \mathbf{x}\right), \\ \nabla_{\mathbf{d}}\hat{d}_{\mathcal{S}}^2(\mathbf{x};\mathbf{X},\mathbf{d},\mathbf{w}) &= 2\mathbf{J}_{\mathbf{d}}\left(\hat{\mathbf{x}}_0 - \mathbf{x}\right),\end{aligned} \tag{31}$$

where for convenience we use the squared distance. Assuming that $s = (t,\mathbf{u})$ is given in normalized barycentric coordinates $u_1 + u_2 + u_3 = 1$, one has $\mathbf{x} = \mathbf{X}\mathbf{u} = u_1\mathbf{x}_1 + u_2\mathbf{x}_2 + u_3\mathbf{x}_3$, where $\mathbf{X}$ is a precomputed matrix defining the geometry of the triangle t.

Hence, we have a tool to compute the approximate geodesic distance $\hat{d}^2_{\mathcal{S}}(s,s')$ given $\mathbf{d} = (d^2_{\mathcal{S}}(s_1,s'), d^2_{\mathcal{S}}(s_2,s'), d^2_{\mathcal{S}}(s_3,s'))^{\mathrm{T}}$ and a vector of weights $\mathbf{w}$. The derivatives of $\hat{d}^2_{\mathcal{S}}$ with respect to $\mathbf{u}$ are obtained using the chain rule, $\nabla_{\mathbf{u}} \hat{d}^2_{\mathcal{S}} = \mathbf{X}^{\mathrm{T}} \nabla_{\mathbf{x}} \hat{d}^2_{\mathcal{S}}(\mathbf{X}\mathbf{u})$.

Let us now address the issue of weights. We fix s' to be a point on $\mathcal{S}_N$ and assume that the vector $\mathbf{d} = (d^2_1, d^2_2, d^2_3)^{\mathrm{T}}$ of the geodesic distances $d_{\mathcal{S}}(s', s_i)$ for $i = 1,2,3$ is known. It is straightforward that for nonzero weights $\mathbf{w}$ and $s = (t, \mathbf{u})$ belonging to the interior of the triangle s_1, s_2, s_3 (i.e., $u_i > 0$), the function $\hat{d}^2_{\mathcal{S}}(\mathbf{u})$ is $\mathcal{C}^1$. However, the transition of s to an adjacent triangle, say s_4, s_3, s_2, will cause the vector of distances to change to $\mathbf{d} = (d^2_4, d^2_2, d^2_3)^{\mathrm{T}}$. As a result, $\hat{d}^2_{\mathcal{S}}(\mathbf{u})$ is liable to be discontinuous along the triangle edges. This complicates the numerical solution of (13). Therefore, our goal is to achieve at least $\mathcal{C}^1$ continuity for $\hat{d}^2_{\mathcal{S}}$ on all of $\mathcal{S}_N$.

In order to handle this problem, we set the weights $\mathbf{w}$ proportional to the barycentric coordinates of s, $\mathbf{u}$. Inside the triangle one has $u_i > 0$ and, consequently, the weights are set to some positive values inside the triangle. For s located on the edge $\overline{s_2 s_3}$, one has $w_1 = u_1 = 0$, which cancels the influence of the vertex s_1 in the least squares problem. Hence, the same vertices participate with the same weights when $\hat{d}^2_{\mathcal{S}}$ is computed from both triangles s_1, s_2, s_3 and s_2, s_3, s_4, yielding a continuous transition of $\hat{d}^2_{\mathcal{S}}$ across the edge. In order to enforce $\mathcal{C}^1$ continuity for $\hat{d}^2_{\mathcal{S}}$, a change of $\mathbf{u}$ by a unit vector $\Delta\mathbf{u}$ and of $\mathbf{u}'$ by a unit vector $\Delta\mathbf{u}'$ must displace the point $\mathbf{x}$ inside the triangles t and t' by the same Euclidean distance. Since barycentric coordinates are defined up to scale, this can be enforced by selecting an appropriate scaling for $\mathbf{u}$ in each triangle.

Since the least-squares problem (20) is underdetermined when one of the weights is set to zero and generally yields two distinct solutions, one has to select the solution consistent with the case when the weight is slightly larger than zero. In practice, it is easier to avoid the second inconsistent solution by setting the weight to some small positive constant ϵ rather than strictly to zero. At the triangle vertices two weights vanish, which makes the least-squares problem underdetermined. In this case, however, there is no need for the least-squares solution, since the geodesic distances between the triangle edges to s' are assumed to be known.

Our final step is to put everything together to obtain the approximant $d^2_{\mathcal{S}}(s,s')$ of the geodesic distance between two arbitrary points $s = (t, \mathbf{u})$ and $s' = (t', \mathbf{u}')$ on $\mathcal{S}_N$, given the mutual geodesic distances $d_{\mathcal{S}}(s_i, s'_j)$ between the vertices s_1, s_2, s_3 of t and s'_1, s'_2, s'_3 of t'. We first compute the approximate geodesic distances from s to s'_i, $\hat{d}^2_{\mathcal{S}}(s, s'_i) = \hat{d}^2_{\mathcal{S}}(\mathbf{u}; \mathbf{X}, \mathbf{d}_i, \mathbf{w} = \mathbf{u})$ for $i = 1,2,3$, where $\mathbf{X}$ is the matrix with the canonical coordinates of s_1, s_2, s_3 and $\mathbf{d}_i = (d^2_{\mathcal{S}}(s_1, s'_i), d^2_{\mathcal{S}}(s_2, s'_i), d^2_{\mathcal{S}}(s_3, s'_i))^{\mathrm{T}}$. Using the vector of distances $\hat{\mathbf{d}} = (\hat{d}^2_{\mathcal{S}}(s, s'_1), \hat{d}^2_{\mathcal{S}}(s, s'_2), \hat{d}^2_{\mathcal{S}}(s, s'_3))^{\mathrm{T}}$ and operating in the triangle t', we compute the approximate geodesic distance between s and s', $\hat{d}^2_{\mathcal{S}}(s, s') = \hat{d}^2_{\mathcal{S}}(\mathbf{u}'; \mathbf{X}', \hat{\mathbf{d}}, \mathbf{w} = \mathbf{u}')$, where $\mathbf{X}'$ are the canonical coordinates of s'_1, s'_2, s'_3.

Using the chain rule, it is straightforward to show that the derivatives of $\hat{d}^2_{\mathcal{S}}(s, s')$ with respect to $\mathbf{u}$ and $\mathbf{u}'$ are given by

$$\nabla_{\mathbf{u}'} \hat{d}^2_{\mathcal{S}}(s,s') = \nabla_{\mathbf{u}} \hat{d}^2_{\mathcal{S}}\big(\mathbf{u}'; \mathbf{X}', \hat{\mathbf{d}}, \mathbf{u}'\big) + \nabla_{\mathbf{w}} \hat{d}^2_{\mathcal{S}}\big(\mathbf{u}'; \mathbf{X}', \hat{\mathbf{d}}, \mathbf{u}'\big),$$

$$\nabla_{\mathbf{u}} \hat{d}^2_{\mathcal{S}}(s,s') = \big(\nabla_{\mathbf{u}} \hat{d}^2_1, \nabla_{\mathbf{u}} \hat{d}^2_2, \nabla_{\mathbf{u}} \hat{d}^2_3\big)^{\mathrm{T}} \nabla_{\mathbf{d}} \hat{d}^2_{\mathcal{S}}\big(\mathbf{u}'; \mathbf{X}', \hat{\mathbf{d}}, \mathbf{u}'\big), \tag{32}$$

where

$$\nabla_{\mathbf{u}} \hat{d}^2_i = \nabla_{\mathbf{u}} \hat{d}^2_{\mathcal{S}}(\mathbf{u}; \mathbf{X}, \mathbf{d}_i, \mathbf{u}) + \nabla_{\mathbf{w}} \hat{d}^2_{\mathcal{S}}(\mathbf{u}; \mathbf{X}, \mathbf{d}_i, \mathbf{u}).$$

Acknowledgment. We are grateful to D. Burago, A. Brook, and V. Lin for their valuable comments and fruitful discussions. We thank our reviewers for their illuminating remarks, which helped us to improve the clarity of this paper.

REFERENCES

[1] D. Bertsekas, *Nonlinear Programming*, 2nd ed., Atlanta Scientific, Belmont, MA, 1999.
[2] P. J. Besl and N. D. McKay, *A method for registration of 3D shapes*, IEEE Trans. PAMI, 14 (1992), pp. 239–256.
[3] I. Borg and P. Groenen, *Modern Multidimensional Scaling: Theory and Applications*, Springer-Verlag, Berlin, Heidelberg, New York, 1997.
[4] A. M. Bronstein, M. M. Bronstein, and R. Kimmel, *Expression-invariant face recognition via spherical embedding*, in Proceedings of the International Conference on Image Processing, Vol. 3, 2005, pp. 756–759.
[5] A. M. Bronstein, M. M. Bronstein, and R. Kimmel, *On isometric embedding of facial surfaces into* $\mathbb{S}^3$, in Proceedings of the International Conference on Scale Space and PDE Methods in Computer Vision, Lecture Notes on Comput. Sci. 3459, Springer-Verlag, Berlin, 2005, pp. 622–631.
[6] A. M. Bronstein, M. M. Bronstein, and R. Kimmel, *Three-dimensional face recognition*, Internat. J. Comput. Vision, 64 (2005), pp. 5–30.
[7] A. M. Bronstein, M. M. Bronstein, and R. Kimmel, *Generalized multidimensional scaling: A framework for isometry-invariant partial surface matching*, Proc. Natl. Acad. Sci. USA, 103 (2006), pp. 1168–1172.
[8] A. M. Bronstein, M. M. Bronstein, and R. Kimmel, *Robust expression-invariant face recognition from partially missing data*, in Proceedings of the European Conference on Computer Vision (ECCV), 2006, pp. 396–408.
[9] M. M. Bronstein, A. M. Bronstein, R. Kimmel, and I. Yavneh, *Multigrid multidimensional scaling*, Numer. Linear Algebra Appl., 13 (2006), pp. 149–171.
[10] D. Burago, Y. Burago, and S. Ivanov, *A Course in Metric Geometry*, Grad. Stud. Math. 33, AMS, Providence, RI, 2001.
[11] Y. Chen and G. Medioni, *Object modeling by registration of multiple range images*, Image and Vision Comput., 10 (1992), pp. 145–155.
[12] D. Donoho and C. Grimes, *Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data*, Proc. Natl. Acad. Sci. USA, 100 (2003), pp. 5591–5596.
[13] A. Elad and R. Kimmel, *On bending invariant signatures for surfaces*, IEEE Trans. PAMI, 25 (2003), pp. 1285–1295.
[14] Y. Eldar, M. Lindenbaum, M. Porat, and Y. Y. Zeevi, *The farthest point strategy for progressive image sampling*, IEEE Trans. Image Processing, 6 (1997), pp. 1305–1315.
[15] M. Gromov, *Isometric immersions and embeddings*, Soviet Math. Dokl., 11 (1970), pp. 794–797.
[16] M. Gromov, *Structures métriques pour les variétés riemanniennes*, Textes Math. 1, Cedic, Paris, 1981.
[17] R. Grossmann, N. Kiryati, and R. Kimmel, *Computational surface flattening: A voxel-based approach*, IEEE Trans. Pattern Anal. Machine Intelligence, 24 (2002), pp. 433–441.
[18] N. J. Kalton and M. I. Ostrovskii, *Distances between Banach spaces*, Forum Math., 11 (1999), pp. 17–48.
[19] R. Kimmel and J. A. Sethian, *Computing geodesic on manifolds*, Proc. Natl. Acad. Sci. USA, 95 (1998), pp. 8431–8435.
[20] N. Linial, E. London, and Y. Rabinovich, *The geometry of graphs and some its algorithmic applications*, Combinatorica, 15 (1995), pp. 333–344.
[21] F. Mémoli and G. Sapiro, *A theoretical and computational framework for isometry invariant recognition of point cloud data*, Found. Comput. Math., 5 (2005), pp. 313–346.
[22] S. T. Roweis and L. K. Saul, *Nonlinear dimensionality reduction by locally linear embedding*, Science, 290 (5500) (2000), pp. 2323–2326.
[23] E. L. Schwartz, A. Shaw, and E. Wolfson, *A numerical solution to the generalized mapmaker's problem: Flattening nonconvex polyhedral surfaces*, IEEE Trans. PAMI, 11 (1989), pp. 1005–1008.
[24] J. A. Sethian, *A fast marching level set method for monotonically advancing fronts*, Proc. Nat. Acad. Sci. USA, 93 (1996), pp. 1591–1595.
[25] J. B. Tennenbaum, V. de Silva, and J. C Langford, *A global geometric framework for nonlinear dimensionality reduction*, Science, 290 (2000), pp. 2319–2323.

[26] J. Walter and H. Ritter, *On interactive visualization of high-dimensional data using the hyperbolic plane*, in Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2002, pp. 123–131.

[27] Z. Y. Zhang, *Iterative point matching for registration of free-form curves and surfaces*, Internat. J. Comput. Vision, 13 (1994), pp. 119–152.

[28] G. Zigelman, R. Kimmel, and N. Kiryati, *Texture mapping using surface flattening via multi-dimensional scaling*, IEEE Trans. Visualization and Comput. Graphics, 9 (2002), pp. 198–207.

SIAM J. SCI. COMPUT.
Vol. 28, No. 5, pp. 1837–1850
© 2006 Society for Industrial and Applied Mathematics

EXTENSIONS OF PRIEST'S DOUBLE-PRECISION SUMMATION*

YVES NIEVERGELT†

Abstract. From single-precision summands sorted in nonincreasing exponents, extended-precision faithful additions with any radix yield an extended-precision sum, which, rounded back to single-precision, suffers from a roundoff smaller than one unit in the penultimate significant digit. Such provable accuracy holds only if the radix-based logarithm of the number of summands does not exceed the difference between the number of digits in extended and single precisions.

Key words. accuracy, extended precision, faithful floating-point addition, rounding

AMS subject classification. 65G50

DOI. 10.1137/040617005

1. Introduction.

This work resolves an issue raised by Demmel and Hida [6, p. 1247] about extensions of Priest's double-precision summation [26, pp. 62–63]. Priest's double-precision summation sorts single-precision floating-point summands in nonincreasing , then adds them with -precision floating-point arithmetic using radix. Demmel and Hida's extended-precision summation sorts single-precision binary floating-point summands in nonincreasing , then adds them with -precision floating-point arithmetic properly rounding to the even floating-point number.

Adding first the summands with the largest magnitudes (or exponents) allows for exact partial sums of summands that might have nearly equal magnitudes but opposite signs. The cap on the number of summands then prevents roundoffs from compounding to a size affecting more than the penultimate digit of the rounded sum.

The issue, affirmatively settled here, is whether Priest's and Demmel and Hida's results both generalize to -precision summations using radix.

2. Notation.

The set $\mathtt{F}(B, n, E_{\min}, E_{\max})$, abbreviated by $\mathtt{F}_n$, of with n significant digits with radix B consists of all numbers of the form

$$d = (-1)^s * B^{\mathtt{E}(d)} * d_0.d_1 \dots d_{n-1}, \tag{1}$$

which is if $d_0 > 0$. Here $s \in \{0, 1\}$ is the , $\mathtt{E}(d)$ is an integer in a range $E_{\min} \le \mathtt{E}(d) \le E_{\max}$, and $d_0, \dots, d_{n-1} \in \{0, \dots, B-1\}$ are the . If $0 < |d| < B^{E_{\min}}$, but despite a leading zero d has the form

$$d = (-1)^s * B^{E_{\min}} * 0.d_1 \dots d_{n-2}d_{n-1}, \tag{2}$$

then d is called a , or , or number. For every floating-point number $d \ne 0$, a (ulp) is defined by

$$\mathtt{ulp}_n(d) := B^{\mathtt{E}(d)} * 0.0 \dots 000 \dots 01 = B^{\mathtt{E}(d)+1-n}. \tag{3}$$

*Received by the editors October 14, 2004; accepted for publication (in revised form) May 18, 2006; published electronically November 14, 2006. This work was supported in part by a Faculty Research Grant from Eastern Washington University.
http://www.siam.org/journals/sisc/28-5/61700.html
†Department of Mathematics, Eastern Washington University, 216 Kingston Hall, Cheney, WA 99004-2418 (ynievergelt@ewu.edu).

Let $\Omega_n := \max(\mathtt{F}_n) = B^{E_{\max}} \cdot (B - B^{1-n})$ denote the largest magnitude in $\mathtt{F}_n$, and define the set of floating-point numbers by $\overline{\mathtt{F}}_n := \mathtt{F}_n \cup \{-\infty, +\infty\}$.

A function denoted by $\mathtt{float}_n$ or by $\mathtt{round}_n : \mathbb{R} \to \overline{\mathtt{F}}_n$ approximates each real number $x \in \mathbb{R}$ by an extended floating-point number $\mathtt{round}_n(x) \in \overline{\mathtt{F}}_n$.

Each arithmetic operation $\circ \in \{+, -, *, /\}$ is a function $\circ : \mathbb{R}^2 \to \mathbb{R}$ (with exceptions for divisions by zero), which a floating-point computer approximates by a function $\circledcirc_n$ or $\circledcirc : \mathtt{F}_n^2 \to \overline{\mathtt{F}}_n$ (with corresponding exceptions for divisions by zero).

More generally, a floating-point computer can approximate a function $\mathrm{S} : \mathbb{R}^\ell \to \mathbb{R}$ or $\mathrm{S} : \mathtt{F}_n^\ell \to \mathbb{R}$ by a function $Ⓢ : \mathbb{R}^\ell \to \overline{\mathtt{F}}_n$ or $Ⓢ : \mathtt{F}_n^\ell \to \overline{\mathtt{F}}_n$. The difference $\mathrm{S} - Ⓢ$ is a or . Thus, if $\mathrm{S} : \mathbb{R} \to \mathbb{R}$ is the "identity" function, defined by $\mathrm{S}(r) := r$ for every $r \in \mathbb{R}$, then $Ⓢ : \mathbb{R} \to \overline{\mathtt{F}}_n$ can be $Ⓢ := \mathtt{round}_n$. Similarly, if $\mathrm{R} : \mathbb{R}_+ \to \mathbb{R}_+$ is the square root function, with $\mathrm{R}(r) := \sqrt{r}$ for every $r \in \mathbb{R}_+ = \{r \in \mathbb{R} : r \geq 0\}$, then Ⓡ is a floating-point square root.

Two extended floating-point numbers $p, q \in \overline{\mathtt{F}}_n$ are if and only if $\overline{\mathtt{F}}_n$ does not contain any element x such that $p < x < q$ or $q < x < p$. For each function $\mathrm{c} : \mathbb{R}^\ell \to \mathbb{R}$ and each point $\vec{\mathbf{r}} \in \mathtt{F}^\ell$ such that $\mathrm{c}(\vec{\mathbf{r}}) \in \mathbb{R}$, denote by $p, q \in \overline{\mathtt{F}}_n$ the two extended floating-point numbers with the $\mathrm{c}(\vec{\boldsymbol{r}})$ and

$$|p| \leq |\mathrm{c}(\vec{\mathbf{r}})| < |q|.$$

A floating-point approximation $ⓒ : \mathtt{F}_n^\ell \to \overline{\mathtt{F}}_n$ is if and only if

$$ⓒ(\vec{\mathbf{r}}) := \begin{cases} p & \text{if } p = \mathrm{c}(\vec{\mathbf{r}}), \text{ else} \\ \text{either } p \text{ or } q & \text{if } p \neq \mathrm{c}(\vec{\mathbf{r}}) \text{ and } |\mathrm{c}(\vec{\mathbf{r}})| < \Omega_n + \mathtt{ulp}(\Omega_n), \text{ else} \\ q & \text{if } |\mathrm{c}(\vec{\mathbf{r}})| \geq \Omega_n + \mathtt{ulp}(\Omega_n). \end{cases}$$

For $p \neq \mathrm{c}(\vec{\mathbf{r}})$, a faithful approximation $-\infty$ if $ⓒ(\vec{\mathbf{r}}) := \min\{p, q\}$, $+\infty$ if $ⓒ(\vec{\mathbf{r}}) := \max\{p, q\}$, ( ) if $ⓒ(\vec{\mathbf{r}}) := p$, or if $ⓒ(\vec{\mathbf{r}}) := q$. A faithful approximation if and only if

$$ⓒ(\vec{\mathbf{r}}) := \begin{cases} p & \text{if } |p| \leq |\mathrm{c}(\vec{\mathbf{r}})| < (|p| + |q|)/2 < +\infty, \text{ else} \\ \text{either } p \text{ or } q & \text{if } (p+q)/2 = \mathrm{c}(\vec{\mathbf{r}}), \text{ else} \\ q & \text{if } (|p| + |q|)/2 < |\mathrm{c}(\vec{\mathbf{r}})| < |q|. \end{cases}$$

For instance, ⓒ if and only if it is properly rounding and rounds $(p+q)/2 = \mathrm{c}(\vec{\mathbf{r}})$ to whichever value p or q has an even last digit. Thus,

$$|\mathrm{c}(\vec{\mathbf{r}}) - ⓒ(\vec{\mathbf{r}})| \leq \begin{cases} \frac{1}{2}\mathtt{ulp}_n\left[ⓒ(\vec{\mathbf{r}})\right] & \text{for properly rounding arithmetic, or} \\ \mathtt{ulp}_n\left[ⓒ(\vec{\mathbf{r}})\right] & \text{for faithful arithmetic for } ⓒ(\vec{\mathbf{r}}) \neq \pm\infty. \end{cases}$$

If $|\mathrm{c}(\vec{\mathbf{r}})| < B^{E_{\min}}$, then ⓒ may either represent $\mathrm{c}(\vec{\mathbf{r}})$ by a subnormal number $ⓒ(\vec{\mathbf{r}})$ as in (2), or may round $\mathrm{c}(\vec{\mathbf{r}})$ to either 0 or $\pm B^{E_{\min}}$, which can be accompanied by an exception flag called an . In particular, the floating-point number, $\iota_n := \min\{d \in \mathtt{F}_n : d > 0\}$, is $B^{E_{\min}}$ in the first case, or $B^{E_{\min}+1-m}$ in the second case. Therefore, if $\mathrm{c}(\vec{\mathbf{r}}) = 0$, then the foregoing definitions of faithfulness apply, with $p = 0$ and $q = \iota_n$. If $|\mathrm{c}(\vec{\mathbf{r}})| \geq \Omega_n + \mathtt{ulp}_n(\Omega_n)$, then ⓒ represents $\mathrm{c}(\vec{\mathbf{r}})$ by $\pm\infty$, which can be accompanied by an exception flag called an . The result of $0 \oslash 0$ can be yet another exception flag, $\mathtt{NaN} \notin \overline{\mathtt{F}}_n$, called . If $s = 0$ but $r \neq 0$, then $r \oslash 0$ can be either NaN, or $+\infty$ for $r > 0$, or $-\infty$ for $r < 0$.

A ($\mathtt{uls}$) is defined by

$$\mathtt{uls}(d) := B^{\mathtt{E}(d)} * 0.0\ldots010\ldots00 = B^{\mathtt{E}(d)-\ell} \tag{4}$$

if $d_\ell > 0$ but $d_i = 0$ for each $i > \ell$. The number of [illegible] of d is

$$\mathtt{Z}(d) := \mathtt{E}[\mathtt{uls}(d)] - \mathtt{E}[\mathtt{ulp}(d)],$$

while the [illegible] of d is then

$$\mathtt{L}(d) := \ell.$$

A "k-digit multiple of $\mathtt{ulp}_i(p)$" will mean a number of the form $j \cdot \mathtt{ulp}_i(p)$ with an integer $j \in \mathbb{Z}$ with [illegible] k nonzero digits, so that $0 \le |j| < B^k$.

Finally, if $m < n$, then the exponent ranges are assumed so that $\mathtt{F}_m \subset \mathtt{F}_n$.

[illegible] 1. Some faithful floating-point operations $\circledcirc$ do [illegible] factor as the composition of a real arithmetic operation $\circ$ followed by a rounding [illegible] $\mathtt{round}$. Indeed, $r \circledcirc s \ne \mathtt{round}(r \circ s)$ if $r \circledcirc s$ depends on factors other than r, s, and $r \circ s$.

[illegible] 1. Random rounding [23] is faithful yet pseudorandom.

The phenomenon in Remark 1 occurs for [illegible] floating-point addition with at least one [illegible] digit, which is faithful [24, eqn. (8), p. 403], [29, sect. 1.8, pp. 22–20].

[illegible] 2. The IBM System/360/370/390 and the space shuttle's IBM AP-101 [illegible] single-precision floating-point additions with six hexadecimal digits using one hexadecimal [illegible] digit [18, pp. 45–46], [19, p. 160], [21, pp. 18-8–18-9, pp. A-5–A-6], [30, Chap. 4]. With $B = 16$ and $n = 6$, let $\mathrm{A}_{\mathrm{hex}} := 10_{\mathrm{ten}}, \ldots, \mathrm{F}_{\mathrm{hex}} := 15_{\mathrm{ten}}$.

If $u = 123456$ and $v = 0.\mathrm{F8}$, then aligning u and v with only one guard digit chops the last digit of v and gives $123456 + 0.\mathrm{F}$, which is chopped to $u \oplus v = 123456$.

If $r = 123457$ and $s = -0.08$, then aligning r and s with only one guard digit chops the last digit of s and gives $123457 - 0.0$, which is chopped to $r \oplus s = 123457$.

Thus $u + v = 123456.\mathrm{F8} = r + s$ but $u \oplus v = 123456 \ne 123457 = r \oplus s$. Therefore, there is no function $\mathtt{round} : \mathbb{R} \to \overline{\mathtt{F}}_n$ such that $z \oplus w = \mathtt{round}(z + w)$ for all $z, w \in \mathtt{F}_n$.

[illegible] 3. For contrast with Example 4, the IBM 7090 & 7094 store binary floating-point numbers in the form $d = \pm 2^{\mathtt{E}(d)} * .d_1 \ldots d_n$ with $n = 27$, and [illegible] additions using 27 [illegible] digits. For all $s, u \in \mathtt{F}_{27}$ such that $\mathtt{E}(s) \ge \mathtt{E}(u)$, they compute $v', w' \in \mathtt{F}_{27}$ so that $s + u = v' + w' +$ lower-order bits, with $\mathrm{sign}(w') = \mathrm{sign}(v')$ and $\mathtt{E}(w') = \mathtt{E}(v') - 27$, and then set $s \oplus u := v'$ [16, pp. 32–33], [17, pp. 26–27]. The double word (v', w')—not in $\overline{\mathtt{F}}_n$ but accessible to the programmer [16, p. 43], [17, p. 33]—is yet another type of floating-point sum, described by Brown [1], [26, p. 28].

The phenomenon in Remark 1 also occurs with Dekker's and Priest's "properly truncating" addition [4, def. (3.5), p. 227], [25, p. 134], [26, p. 17], which is faithful but may have been confused with "correctly chopping" addition [22], [25, p. 134].

Definition 1. [illegible] properly truncating [illegible] $r, s \in \mathtt{F}_n$ [illegible] $|r| \ge |s|$ [illegible] $p, q \in \mathtt{F}_n$ [illegible] $p < r + s < q$ [illegible]

$$r \oplus s = \begin{cases} p & \text{if } s > 0, \\ q & \text{if } s < 0. \end{cases}$$

[illegible] 4. The IBM 704 stores binary floating-point numbers in the form $d = \pm 2^{\mathtt{E}(d)} * .d_1 \ldots d_n$ with $n = 27$ [15, pp. 8–9, 21–22]. To add $s, u \in \mathtt{F}_n$ ranked so that $\mathtt{E}(s) \ge \mathtt{E}(u)$, it aligns the digits of u into two floating-point numbers $v, w \in \mathtt{F}_n$ with $\mathrm{sign}(v) = \mathrm{sign}(u) = \mathrm{sign}(w)$, $\mathtt{E}(v) = \mathtt{E}(s)$, and $\mathtt{E}(w) = \mathtt{E}(s) - 27$, so that $u = v + w +$ chopped lower digits of u. The adder computes $v' := s + v$, and adds (or subtracts) (signed) bits from w until v' is normalized, so that $s + v + w = v' + w' +$ dropped bit in case of a carry, but $\mathrm{sign}(w') = \mathrm{sign}(u)$ regardless of $\mathrm{sign}(v')$, in contrast to Example 3.

If $s = +2^{-1} * (1 - 2^{-27})$ and $u = 2^{-29} > 0$, then $v = +2^0 * .0$ and $w = +2^{-28} * .01$. Hence $v' = s + v = s$ and $w' = w$, whence $s \oplus u = s$ is rounded with $u > 0$.

If $b = +2^{-1} = +2^0 * .1$ and $d = -2^{-29}$, then $v = +2^0 * .0$ and $w = -2^{-27} * .01$. Hence $v' = b + v = b$ and $w' = w$, whence $b \oplus d = b$ is rounded with $d < 0$.

Thus $s + u = b + d$ but $s \oplus u = s \neq b = b \oplus d$, and both are

2. If $\textcircled{c}_1$ and $\textcircled{c}_2$ are two faithful approximations of a function c, and if $\textcircled{c}_3(\vec{\mathbf{r}}) \in \{\textcircled{c}_1(\vec{\mathbf{r}}), \textcircled{c}_2(\vec{\mathbf{r}})\}$ for every $\vec{\mathbf{r}}$, then $\textcircled{c}_3$ is also a faithful approximation of c.

Remark 2 will simplify the proof of Corollary 3 below. In contrast, the assumption that $\textcircled{c}(\vec{\mathbf{r}}) = \mathtt{round}\,[\mathrm{c}(\vec{\mathbf{r}})]$ would not substantially simplify any of the proofs. Another reason for including the full generality of faithful arithmetic is to allow for all currently available faithful arithmetic, for instance, toggling between various rounding modes in interval arithmetic [28], [31], [32]. Yet another reason is to demonstrate by comparison and contrast the advantages and disadvantages of various faithful arithmetic, be they historical, available, or merely contemplated for future design.

In what follows, a function $\mathtt{round}_m : \mathtt{F}_n \to \overline{\mathtt{F}}_m$ will serve only to round an extended-precision sum $s_N \in \mathtt{F}_n$ to a single-precision value $\mathtt{round}_m(s_N) \in \mathtt{F}_m$ with $m < n$.

3. Analysis. Priest's double-precision summation adds N single-precision summands $x_1, \ldots, x_N \in \mathtt{F}_m$ sorted by nonincreasing using at least -precision faithful additions $\oplus_n$ in $\mathtt{F}_n$, with $n \geq 2m$. In contrast, following Demmel and Hida, Algorithm 1 adds N single-precision summands $r_1, \ldots, r_N \in \mathtt{F}_m$ sorted by nonincreasing $\mathtt{E}(r_1) \geq \cdots \geq \mathtt{E}(r_N)$ using -precision faithful additions $\oplus_n$ in $\mathtt{F}_n$, with any $n > m$. Following Priest, however, Algorithm 1 does not yet round the final sum to m digits.

Algorithm 1 (exponent-sorted extended-precision summation).
Data: single-precision numbers $r_1, \ldots, r_N \in \mathtt{F}_m$ such that
$\mathtt{E}(r_1) \geq \cdots \geq \mathtt{E}(r_N)$,
$2 \leq m < n$,
$N \leq 2 + B^{n-m}(1 - B^{-1})/(1 - B^{-m})$.
Result: an extended-precision sum $s_N \in \mathtt{F}_n$ approximating $s := r_1 + \cdots + r_N$.
Start
Let $s_1 := r_1$;
for each $k \in \{2, \ldots, N\}$, add with extended-precision
$s_k := s_{k-1} \oplus_n r_k \in \mathtt{F}_n$;
next k;
return s_N.
Stop.

The analysis examines the differences between the sum $s := \sum_{k=1}^{N} r_k \in \mathbb{R}$, the extended-precision sum $s_N \in \mathtt{F}_n$, and the single-precision value $\mathtt{round}_m(s_N) \in \mathtt{F}_m$.

In their analysis of Algorithm 1, Demmel and Hida compare the last bit of the current summand r_k and the last bit of the current partial sum s_{k-1}. In contrast, Priest compares the last bit of the current summand r_k and the last bit of the partial sum $s_k := r_k + s_{k-1}$. Priest's method is also the basis of the present analysis, which reveals that, starting from the first occurrence of a nonzero roundoff, the exponents of all the subsequent partial sums remain within one unit of one another (Lemmas 6, 7, and 8). The crux of the analysis weakens Priest's condition $\mathtt{ulp}_m(r_i) > \mathtt{ulp}_n(r_i \oplus_n s_{i-1})$ under which the first partial sums are exact [26, line 5, p. 63] and, instead, characterizes the conditions for $s_k = r_k \oplus_n s_{k-1}$ to equal $r_k + s_{k-1}$, as done in Theorem 2 and Corollary 3.

THEOREM 2. [illegible] $m < n$ [illegible] $\mathtt{F}_m \subset \mathtt{F}_n$ [illegible] $p \in \mathtt{F}_m$ [illegible] $q \in \mathtt{F}_n$ [illegible] q [illegible] $\mathtt{uls}(p)$ [illegible] $p \oplus_n q = p + q$ [illegible]

- $\mathtt{uls}(p) \geq \mathtt{ulp}_n(p \oplus_n q)$ [illegible]
- $\mathtt{uls}(p) = B^{-1}\mathtt{ulp}_n(p \oplus_n q)$ [illegible] $\mathtt{uls}(p) = \mathtt{ulp}_n(q)$ [illegible] $p_{\mathtt{L}(p)} + q_{n-1} = B$

[illegible]. Either $\mathtt{ulp}_n(q) < \mathtt{uls}(p)$, or $\mathtt{ulp}_n(q) = \mathtt{uls}(p)$, or $\mathtt{uls}(p) < \mathtt{ulp}_n(q)$.

(2.1) If $\mathtt{ulp}_n(q) < \mathtt{uls}(p)$, then p is an m-digit multiple of $\mathtt{uls}(p)$ while q is an $(n-1)$-digit multiple of $\mathtt{uls}(p)$. Hence $p+q$ is an n-digit multiple of $\mathtt{uls}(p)$, whence $p \oplus_n q = p + q \in \mathtt{F}_n$ by faithfulness.

(2.2) If $\mathtt{uls}(p) = \mathtt{ulp}_n(q)$, then two cases can arise.

(2.2.1) If $\operatorname{sign}(p) \neq \operatorname{sign}(q)$, then $|p|$ and $|q|$ are n-digit multiples of $\mathtt{uls}(p)$, and hence so is their difference, whence $p \oplus_n q = \pm(|p| - |q|) \in \mathtt{F}_n$, so that again $p \oplus_n q = \pm(|p| - |q|) = p + q \in \mathtt{F}_n$ by faithfulness.

(2.2.2) Alternatively, if $\operatorname{sign}(p) = \operatorname{sign}(q)$, then two subcases can occur.

(2.2.2.1) If $|p| + |q|$ is also an n-digit multiple of $\mathtt{uls}(p) = \mathtt{ulp}_n(q)$, then $|p+q| = |p| + |q| \in \mathtt{F}_n$, whence $p \oplus_n q = p + q \in \mathtt{F}_n$ by faithfulness.

(2.2.2.2) In contrast, if $|p| + |q|$ is [illegible] an n-digit multiple of $\mathtt{uls}(p)$, then $|p| + |q|$ is an $(n+1)$-digit multiple of $\mathtt{uls}(p) = \mathtt{ulp}_n(q)$, whose last digit is $p_{\mathtt{L}(p)} + q_{n-1}$, so that $p+q \neq p \oplus_n q$ if and only if the last digit of $p + q$ differs from 0, in other terms, $p_{\mathtt{L}(p)} + q_{n-1} \neq B$, but then rounding to the most significant n digits gives $\mathtt{ulp}_n(p \oplus_n q) > \mathtt{uls}(p)$.

(2.3) If $\mathtt{uls}(p) < \mathtt{ulp}_n(q)$, then $|p| < |q|/B \leq |q|/2$, and $p+q$ is a multiple of $\mathtt{uls}(p)$ whose last nonzero significant digit is either $0 + p_{\mathtt{L}(p)} = p_{\mathtt{L}(p)}$ or $B - p_{\mathtt{L}(p)}$, so that $\mathtt{uls}(p+q) = \mathtt{uls}(p)$; two cases can arise.

(2.3.1) If $\operatorname{sign}(p) = \operatorname{sign}(q)$, then $|p+q| = |p|+|q| \geq |q| \in \mathtt{F}_n$, whence $|p \oplus_n q| \geq |q|$ by faithfulness, and hence $\mathtt{uls}(p) < \mathtt{ulp}_n(q) \leq \mathtt{ulp}_n(p \oplus_n q)$, but then $p \oplus_n q \neq p + q$ because $\mathtt{uls}(p+q) = \mathtt{uls}(p)$ by item (2.3).

(2.3.2) Alternatively, if $\operatorname{sign}(p) \neq \operatorname{sign}(q)$, then $|q| \geq |p+q| = |q| - |p| > |q|/2 > |p|$, and again two subcases can arise.

(2.3.2.1) If $p + q$ is an n-digit multiple of $\mathtt{uls}(p)$, then $p \oplus_n q = p + q$ by faithfulness, but $\mathtt{uls}(p) = \mathtt{uls}(p+q) = \mathtt{ulp}_n(q \oplus_n p)$ by item (2.3).

(2.3.2.2) In contrast, if $p + q$ has more than n significant digits, then after rounding to n digits, $p + q \neq p \oplus_n q$, but then $\mathtt{uls}(p) < \mathtt{ulp}_n(q \oplus_n p)$. □

Corollary 3 extends Theorem 2 where q is an integer multiple of $\mathtt{ulp}_m(p)$.

COROLLARY 3. [illegible] $m < n$ [illegible] $\mathtt{F}_m \subset \mathtt{F}_n$ [illegible] $p \in \mathtt{F}_m$ [illegible] $q \in \mathtt{F}_n$ [illegible] q [illegible] $\mathtt{ulp}_m(p)$ [illegible] $t := p \oplus_n q \neq p + q$ [illegible]

$$|p| \leq B^{\mathtt{E}(t)+(m-n)-1}(B - B^{1-m}) < B^{\mathtt{E}(t)+m-n}.$$

[illegible]. If q is an integral multiple of $\mathtt{uls}(p)$, then Corollary 3 is the contraposition of Theorem 2, with

$$\begin{aligned} |p| &\leq B^{\mathtt{E}(p)}(B - B^{1-m}) = B^{\mathtt{E}(p)+1-m}(B^m - 1) = \mathtt{ulp}_m(p)(B^m - 1) \\ &\leq B^{-1}\mathtt{ulp}_n(t)(B^m - 1) = B^{-1}B^{\mathtt{E}(t)+1-n}(B^m - 1) < B^{\mathtt{E}(t)+m-n} \end{aligned}$$

following from $\mathtt{ulp}_m(p) \leq \mathtt{uls}(p) < \mathtt{ulp}_n(t)$ and (1), (3), (4).

If q is an integral multiple of $\mathtt{ulp}_m(p)$ but [illegible] an integral multiple of $\mathtt{uls}(p)$, then $\mathtt{ulp}_m(p) \leq \mathtt{uls}(q) < \mathtt{uls}(p)$, whence p and q are n-digit multiples of $\mathtt{uls}(q)$. If p and q have opposite signs, then $p \oplus_n q = p + q \in \mathtt{F}_n$ by integer arithmetic and faithfulness.

If p and q have the same sign, then let $k := \mathtt{E}[\mathtt{uls}(p)] - \mathtt{E}[\mathtt{uls}(q)] \le \mathtt{Z}(p)$, so that

$$\begin{array}{rlllll} p &= (-1)^s \cdot B^{\mathtt{E}(p)} \cdot p_0.p_1 \ldots p_{\mathtt{L}(p)} & 0 & \ldots & 0, \\ q &= (-1)^s \cdot B^{\mathtt{E}(q)} \cdot q_0.q_1 \ldots q_{\mathtt{L}(q)-k} & q_{\mathtt{L}(q)-(k-1)} & \ldots & q_{\mathtt{L}(q)}. \end{array}$$

Transferring the k least significant digits of q to p (for the purpose of this proof, not in the computations) yields floating-point numbers

$$\begin{array}{rlllll} p' &:= (-1)^s \cdot B^{\mathtt{E}(p)} \cdot p_0.p_1 \ldots p_{\mathtt{L}(p)} & q_{\mathtt{L}(q)-(k-1)} & \ldots & q_{\mathtt{L}(q)}, \\ q' &:= (-1)^s \cdot B^{\mathtt{E}(q)} \cdot q_0.q_1 \ldots q_{\mathtt{L}(p)} & 0 & \ldots & 0. \end{array}$$

In particular, $p+q = p'+q'$, with $q' \in \mathtt{F}_n$ and still $p' \in \mathtt{F}_m$, because p was an m-digit multiple of $\mathtt{ulp}_m(p)$ and hence of $\mathtt{uls}(q)$. Here $t' := p' \oplus_n q'$ need not equal $t = p \oplus_n q$, but t is a faithful floating-point sum of p' and q' by Remarks 1 and 2. Also, q' is an integral multiple of $\mathtt{uls}(p')$. Consequently, if $p+q \notin \mathtt{F}_n$, then $p'+q' \notin \mathtt{F}_n$, whence $t \ne p'+q'$, and hence $\mathtt{E}(p) = \mathtt{E}(p') < \mathtt{E}(t) + m - n$ by the first part of the proof. ∎

Lemmas 4 and 5 establish inequalities used repeatedly in Lemmas 6, 7, and 8.

LEMMA 4. [illegible] $\mathtt{E}(r_i) \le \mathtt{E}(s_i) + (m-n) - 1$ [illegible]

$$\mathtt{E}(r_{i+j}) \le \mathtt{E}(r_i) \le \mathtt{E}(s_i) + (m-n) - 1, \tag{5}$$

$$|r_{i+j}| \le B^{\mathtt{E}(s_i)+(m-n)-1}(B - B^{1-m}) < B^{\mathtt{E}(s_i)+(m-n)} \tag{6}$$

[illegible] $j \ge 0$ [illegible] $\mathtt{E}(r_i) \le \mathtt{E}(s_i) + (m-n) - 1$ [illegible] $\mathtt{E}(s_{i+j}) \ne \mathtt{E}(s_i)$ [illegible] $2 \le i < N$ [illegible]. If $\mathtt{E}(r_i) \le \mathtt{E}(s_i) + (m-n) - 1$, then $i \ge 2$, because $r_1 = s_1$ for $i = 1$.

If $j \ge 0$ and $\mathtt{E}(s_{i+j}) \ne \mathtt{E}(s_i)$, then $j > 0$, whence $i < N$.

Inequalities (5) and (6) follow from the ordering $\mathtt{E}(r_{i+j}) \le \mathtt{E}(r_i)$. ∎

LEMMA 5. [illegible] $B \ge 2$ [illegible] $n > m \ge 1$ [illegible]

$$2 + B^{n-m} \cdot \frac{1 - B^{-1}}{1 - B^{-m}} \le 1 + B^{1+n-m} \cdot (B-1). \tag{7}$$

[illegible] $B \ge 2$ [illegible] $n > m \ge 2$ [illegible] $0 \le k \le N-2$ [illegible]

$$N \le 1 + B^{1+n-m} \cdot (B-1),$$

[illegible] $1 + k \cdot B^{(m-n)-1} \in \mathtt{F}_n$ [illegible]

$$1 + k \cdot B^{(m-n)-1} \le B - B^{1-n}. \tag{8}$$

[illegible] $B \ge 2$ $n > m \ge 2$ [illegible] $0 \le k \le N-2$ [illegible]

$$N \le 2 + B^{n-m} \cdot \frac{1 - B^{-1}}{1 - B^{-m}}, \tag{9}$$

[illegible] $1 - k \cdot B^{-n} \cdot (B^m - 1) \in \mathtt{F}_n$ [illegible]

$$1 - k \cdot B^{-n} \cdot (B^m - 1) \ge B^{-1}. \tag{10}$$

[illegible] (9) [illegible] $0 \le k \le N-3$ [illegible] $1 + k \cdot B^{m-n} \in \mathtt{F}_n$ [illegible]

$$1 + k \cdot B^{m-n} \le B - B^{1-n}. \tag{11}$$

If $B \geq 2$ and $n > m \geq 1$, then $B - B^{1-m} \geq 2 - B^0 = 1$ and also $B^{n-m} \cdot (1 - B^{-1}) = B^{(n-m)-1} \cdot (B-1) \geq B - 1 \geq 1 \geq 1 - B^{-m}$, whence

$$\begin{aligned} B^{1+n-m} \cdot (B-1) \cdot (1 - B^{-m}) &= B^{n-m} \cdot (B-1) \cdot (B - B^{1-m}) \\ &\geq B^{n-m} \cdot \left[(1 - B^{-1}) \cdot B\right] \cdot 1 \\ &\geq B^{n-m} \cdot \left[(1 - B^{-1}) \cdot 2\right] \\ &\geq B^{n-m} \cdot (1 - B^{-1}) + (1 - B^{-m}). \end{aligned}$$

Dividing by $1 - B^{-m} > 0$ and then adding 1 yields inequality (7).

If $m \geq 2$, then $B^{2-m} \leq 1$ and $B^{(m-1)-n}$ is a multiple of $\mathtt{ulp}_n(1)$. If also $k \leq N - 2 \leq B^{1+n-m} \cdot (B-1) - 1$, then $k \leq B^{1+n-m} \cdot (B-1) - B^{2-m}$, whence

$$\begin{aligned} 1 + k \cdot B^{(m-n)-1} &\leq 1 + \left[B^{1+n-m} \cdot (B-1) - B^{2-m}\right] \cdot B^{(m-n)-1} \\ &= 1 + (B-1) - B^{2-m} \cdot B^{(m-n)-1} \\ &= B - B^{1-n}, \end{aligned}$$

proving (8). Moreover, if $k \leq N - 2 \leq B^{n-m}(1 - B^{-1})/(1 - B^{-m})$, then (10) holds:

$$1 - k \cdot B^{-n} \cdot (B^m - 1) \geq 1 - \left(B^{n-m} \cdot \frac{1 - B^{-1}}{1 - B^{-m}}\right) \cdot B^{-n} \cdot (B^m - 1) = B^{-1}.$$

Furthermore, if $k \leq N - 3 \leq B^{n-m} \cdot (1 - B^{-1})/(1 - B^{-m}) - 1$, then (11) holds:

$$\begin{aligned} 1 + k \cdot B^{m-n} &\leq 1 + \left(B^{n-m} \cdot \frac{1 - B^{-1}}{1 - B^{-m}} - 1\right) \cdot B^{m-n} \\ &= \frac{(1 - B^{-m}) + (1 - B^{-1}) - (B^{m-n} - B^{-n})}{1 - B^{-m}} \\ &= \frac{2 + (B^{-n} - B^{-m}) - (B^{-1} + B^{m-n})}{1 - B^{-m}} \\ &\leq \frac{B + B^{(1-m)-n} - (B^{1-m} + B^{1-n})}{1 - B^{-m}} \\ &= B - B^{1-n}, \end{aligned}$$

because $B^{-n} - B^{-m} < 0 < B^{(1-m)-n}$ and $B^{1-m} < B^{-1}$ with $B^{1-n} < B^{m-n}$. □

Lemma 6 shows that if the summand r_i is sufficiently smaller than the ith partial sum s_i in magnitude, then the exponents of all of the subsequent partial sums can drop by at most one unit below—in which case they never again exceed—that of s_i.

LEMMA 6. If $\mathtt{E}(r_i) \leq \mathtt{E}(s_i) + (m-n) - 1$, $B \geq 2$, and $2 \leq m < n$, and

$$N \leq 2 + B^{n-m}(1 - B^{-1})/(1 - B^{-m}), \tag{12}$$

then $\mathtt{E}(s_i) - 1 \leq \mathtt{E}(s_{i+j})$ for $0 \leq j \leq N - i$. Moreover, if $\ell > i$ and $\mathtt{E}(s_\ell) = \mathtt{E}(s_i) - 1$, then $\mathtt{E}(s_i) - 1 \leq \mathtt{E}(s_{\ell+j}) \leq \mathtt{E}(s_i)$ for $0 \leq j \leq N - \ell$.

PROOF. Assume (13) for some i and j with $0 \leq j \leq N - i \leq N - 2$:

$$|s_{i+j}| \geq B^{\mathtt{E}(s_i)-1}\left\{B - j\left[B^{1+(m-n)} - B^{1-n}\right]\right\}. \tag{13}$$

For instance, (13) holds for $j = 0$ and every $i \leq N$. If $i + j = N$, then $j = N - i$ reaches its upper bound, which does not require any further induction, because $i + (j+1) > N$. Hence assume that $i + j < N$, whence $j + 1 \leq N - i \leq N - 2$.

Combining (13) with (6) from Lemma 4 gives

$$|s_{i+j}| - |r_{i+j+1}| \geq B^{\mathtt{E}(s_i)-1} \left\{ B - (j+1) \left[B^{1+(m-n)} - B^{1-n} \right] \right\}.$$

The reverse triangle inequality, inequality (10) from Lemma 5, and faithfulness yield

$$|s_{i+j+1}| \geq B^{\mathtt{E}(s_i)-1} \left\{ B - (j+1) \left[B^{1+(m-n)} - B^{1-n} \right] \right\}.$$

Thus (13) holds for $j+1$ instead of j. Consequently, $\mathtt{E}(s_{i+j}) \geq \mathtt{E}(s_i) - 1$ for every j such that $0 \leq j \leq N - i$.

Moreover, if there exists $\ell > i$ such that equality holds, $\mathtt{E}(s_\ell) = \mathtt{E}(s_i) - 1$, then $|s_\ell| \leq B^{\mathtt{E}(s_i)-1} \cdot (B - B^{1-n})$. However, $\ell > i \geq 2$ by Lemma 4, whence $\ell \geq 3$, so that there remain at most $N - 3$ summands $r_{\ell+j}$ for $1 \leq j \leq N - \ell \leq N - 3$, with $|r_{\ell+1+j}| \leq B^{\mathtt{E}(s_i)+(m-n)-1} \cdot (B - B^{1-m}) < B^{\mathtt{E}(s_i)} \cdot B^{m-n}$, by hypothesis on $\mathtt{E}(r_i)$.

Hence, assume (14) for some index j with $0 \leq j \leq N - \ell \leq N - 3$:

$$|s_{\ell+j}| \leq B^{\mathtt{E}(s_i)} \cdot \left(1 + j \cdot B^{m-n}\right). \tag{14}$$

For instance, (14) holds for $j = 0$. If also $j + 1 \leq N - \ell$, then

$$|s_{\ell+j}| + |r_{\ell+j+1}| \leq B^{\mathtt{E}(s_i)} \cdot \left(1 + j \cdot B^{m-n}\right) + B^{\mathtt{E}(s_i)} \cdot B^{m-n},$$

by hypothesis on $\mathtt{E}(r_i)$. Consequently, the triangle inequality gives

$$|s_{\ell+j+1}| \leq B^{\mathtt{E}(s_i)} \cdot \left[1 + (j+1) \cdot B^{m-n}\right],$$

by faithfulness and (11) in Lemma 5. Thus (14) also holds for $j + 1$. Therefore, $\mathtt{E}(s_i) - 1 \leq \mathtt{E}(s_{\ell+j}) \leq \mathtt{E}(s_i)$ for every j with $0 \leq j \leq N - \ell$. □

Lemma 7 shows that if the summand r_i is sufficiently smaller than the i-th partial sum s_i in magnitude, then the exponents of all of the following partial sums can jump by at most one unit above—in which case they never drop back below—that of s_i.

Lemma 7. If $\mathtt{E}(r_i) \leq \mathtt{E}(s_i) + (m-n) - 1$, $B \geq 2$, $2 \leq m < n$, and

$$N \leq 1 + B^{1+n-m}(B-1),$$

then $\mathtt{E}(s_{i+j}) \leq \mathtt{E}(s_i) + 1$ for every j with $0 \leq j \leq N - i$.

If $N \leq 2 + B^{n-m}(1 - B^{-1})/(1 - B^{-m})$, then for every $\ell > i$ with $\mathtt{E}(s_\ell) = \mathtt{E}(s_i) + 1$ one has $\mathtt{E}(s_i) \leq \mathtt{E}(s_{\ell+j}) \leq \mathtt{E}(s_i) + 1$ for every j with $0 \leq j \leq N - \ell$.

Proof. Assume (15) for some i and j with $0 \leq j \leq N - i \leq N - 2$:

$$|s_{i+j}| \leq B^{\mathtt{E}(s_i)+1} \left[1 + jB^{(m-n)-1}\right]. \tag{15}$$

For instance, (15) holds for $j = 0$ and every $i \leq N$. If $i+j = N$, then $j = N-i$ reaches its upper bound, which does not require any further induction, because $i+(j+1) > N$. Hence assume that $i + j < N$, whence $j + 1 \leq N - i \leq N - 2$.

Combining (15) with (6) from Lemma 4 gives

$$|s_{i+j}| + |r_{i+j+1}| \leq B^{\mathtt{E}(s_i)+1} \left[1 + (j+1)B^{(m-n)-1}\right].$$

The triangle inequality, inequality (8) from Lemma 5, and faithfulness then give

$$|s_{i+j+1}| \leq B^{\mathtt{E}(s_i)+1} \left[1 + (j+1)B^{(m-n)-1}\right].$$

Thus (15) also holds for $j+1$, whence $\mathtt{E}(s_{i+j}) \le \mathtt{E}(s_i)+1$ for every $j \ge 0$.

Moreover, if $N \le 2 + B^{n-m}(1-B^{-1})/(1-B^{-m})$ and if there exists $\ell > i$ such that equality holds, $\mathtt{E}(s_\ell) = \mathtt{E}(s_i)+1$, then $\mathtt{E}(r_\ell) \le \mathtt{E}(r_i) \le \mathtt{E}(s_i) + (m-n) - 1 < \mathtt{E}(s_\ell) + (m-n) - 1$, whence Lemma 6 holds for ℓ instead of i, which states that $\mathtt{E}(s_i) = \mathtt{E}(s_\ell) - 1 \le \mathtt{E}(s_{\ell+j})$ for every j with $0 \le j \le N-\ell$. □

Hence, Lemma 8 shows that after the first occurrence of a nonzero roundoff, where $s_k \ne s_{k-1} + r_k$, the exponents of all of the subsequent partial sums differ from one another by at most one unit: $|\mathtt{E}(s_{k+i}) - \mathtt{E}(s_{k+j})| \le 1$ for all $i, j \ge 0$.

LEMMA 8. [illegible] $s_i = s_{i-1} + r_i$ [illegible] $i < k$ [illegible] $s_k \ne s_{k-1} + r_k$ [illegible] $N \le 2 + B^{n-m}(1-B^{-1})/(1-B^{-m})$ [illegible] $B \ge 2$ [illegible] $2 \le m < n$ [illegible]

- $\mathtt{E}(s_k) - 1 \le \mathtt{E}(s_{k+j}) \le \mathtt{E}(s_k)$ [illegible] j [illegible] $0 \le j \le N-k$ [illegible]
- $\mathtt{E}(s_k) \le \mathtt{E}(s_{k+j}) \le \mathtt{E}(s_k)+1$ [illegible] j [illegible] $0 \le j \le N-k$

[illegible]. Thanks to the ordering of the summands, s_{k-1} is an integral multiple of $\mathtt{ulp}_m(r_k)$. From $s_k = s_{k-1} \oplus_n r_k \ne s_{k-1} + r_k$, Corollary 3 yields

$$|r_k| \le B^{\mathtt{E}(s_k)+(m-n)-1}(B - B^{1-m}).$$

Hence Lemmas 5, 6, and 7 show that $\mathtt{E}(s_k) - 1 \le \mathtt{E}(s_{k+j}) \le \mathtt{E}(s_k)+1$ for every $j \ge 0$.

Consequently, if there does [illegible] exist any $\ell > k$ such that $\mathtt{E}(s_\ell) = \mathtt{E}(s_k)+1$, then $\mathtt{E}(s_k) - 1 \le \mathtt{E}(s_{k+j}) \le \mathtt{E}(s_k)$ for every $j \ge 0$.

In contrast, if there exists $\ell > k$ such that $\mathtt{E}(s_\ell) = \mathtt{E}(s_k)+1$, so that $\mathtt{E}(s_\ell) > \mathtt{E}(s_k)$, then the contraposition of Lemma 6 with k instead of i shows that $\mathtt{E}(s_\ell) \ne \mathtt{E}(s_k) - 1$, whence $\mathtt{E}(s_\ell) > \mathtt{E}(s_k) - 1$ and hence $\mathtt{E}(s_\ell) \ge \mathtt{E}(s_k)$ for every $\ell \ge k$, or, equivalently, $\mathtt{E}(s_k) \le \mathtt{E}(s_{k+j})$ for every $j \ge 0$, whence Lemma 7 shows that $\mathtt{E}(s_k) \le \mathtt{E}(s_{k+j}) \le \mathtt{E}(s_k)+1$ for every $j \ge 0$. □

Theorem 9 shows the accuracy of Algorithm 1 for faithful addition with any radix.

THEOREM 9. [illegible] B, m, n, N [illegible] $B \ge 2$ [illegible] $n > m \ge 2$ [illegible] $N \le 2 + B^{n-m} \cdot (1-B^{-1})/(1-B^{-m})$ [illegible]

$$|s_N - s| \le [1 + (N-2)B] \cdot \mathtt{ulp}_n(s_N); \tag{16}$$

$$|\mathtt{round}_m(s_N) - s| < B \cdot \mathtt{ulp}_m\left[\mathtt{round}_m(s_N)\right]. \tag{17}$$

[illegible]. By hypothesis, s_{k-1} is exact. By faithfulness, the magnitude of the first roundoff does not exceed $B^{\mathtt{E}(s_k)+1-n}$. Hence, by Lemma 8, two cases can occur.

(9.1) If $\mathtt{E}(s_k) - 1 \le \mathtt{E}(s_\ell) \le \mathtt{E}(s_k)$ for each $\ell \ge k$, then the $N-1$ subsequent roundoffs are bounded by $B^{\mathtt{E}(s_k)+1-n}$, and two subcases can arise.

(9.1.1) If there does [illegible] exist any $\ell \ge k$ such that $\mathtt{E}(s_\ell) = \mathtt{E}(s_k) - 1$, then $\mathtt{E}(s_N) = \mathtt{E}(s_k)$, whence the total error is bounded by

$$\begin{aligned} |s - s_N| &\le (N-1)B^{\mathtt{E}(s_k)+1-n} \\ &= (N-1)\mathtt{ulp}_n(s_N) \\ &\le [1 + (N-2)B]\,\mathtt{ulp}_n(s_N), \end{aligned}$$

which proves (16) in this subcase.

(9.1.2) If there exists some $\ell \ge k$ such that $\mathtt{E}(s_\ell) = \mathtt{E}(s_k) - 1$, then at least one roundoff (namely that of s_ℓ) is bounded by $B^{\mathtt{E}(s_k)-n}$, whence

$$\begin{aligned} |s - s_N| &\le [1 + (N-2)B]\,B^{\mathtt{E}(s_k)-n} \\ &\le [1 + (N-2)B]\,\mathtt{ulp}_n(s_N), \end{aligned}$$

because $\mathtt{E}(s_k) - 1 \le \mathtt{E}(s_N) \le \mathtt{E}(s_k)$, which proves (16).

(9.2) In contrast, if $\mathtt{E}(s_k) \le \mathtt{E}(s_\ell) \le \mathtt{E}(s_k)+1$ for every $\ell \ge k$, then each roundoff after that of s_k is bounded by $B^{\mathtt{E}(s_k)+2-n}$, whence

$$\begin{aligned} |s - s_N| &\le [1+(N-2)B]\, B^{\mathtt{E}(s_k)+1-n} \\ &\le [1+(N-2)B]\, \mathtt{ulp}_n(s_N), \end{aligned}$$

because $\mathtt{E}(s_k) \le \mathtt{E}(s_N) \le \mathtt{E}(s_k)+1$, which proves (16).

Rounding $s_N \in \mathtt{F}_n$ to $\mathtt{round}_m(s_N) \in \mathtt{F}_m$ either truncates or rounds up magnitudes, which does not decrease the exponent, so that $\mathtt{E}\,[\mathtt{round}_m(s_N)] \ge \mathtt{E}(s_N)$. Therefore,

$$\begin{aligned} \mathtt{ulp}_m\,[\mathtt{round}_m(s_N)] &= B^{\mathtt{E}[\mathtt{round}_m(s_N)]+1-m} \\ &\ge B^{\mathtt{E}(s_N)+1-m} \\ &= B^{n-m} \cdot B^{\mathtt{E}(s_N)+1-n} \\ &= B^{n-m} \cdot \mathtt{ulp}_n(s_N). \end{aligned} \tag{18}$$

The roundoff from $\mathtt{round}_m : \mathtt{F}_n \to \overline{\mathtt{F}}_m$ is at most $\mathtt{ulp}_m\,[\mathtt{round}_m(s_N)] \cdot (1-B^{m-n})$, barring overflow. Hence (12), (16), and (18) lead to (17):

$$\begin{aligned} |s - \mathtt{round}_m(s_N)| &\le \left\{(1-B^{m-n}) + [1+B(N-2)]\, B^{m-n}\right\} \cdot \mathtt{ulp}_m\,[\mathtt{round}_m(s_N)] \\ &\le \left(1 + B \cdot \frac{1-B^{-1}}{1-B^{-m}} \cdot B^{m-n}\right) \cdot \mathtt{ulp}_m\,[\mathtt{round}_m(s_N)] \\ &< B \cdot \mathtt{ulp}_m\,[\mathtt{round}_m(s_N)]\,, \end{aligned}$$

because $(B-1)\cdot B^{m-n} < (B-1)\cdot(1-B^{-m})$ for all $B \ge 2$ and $n > m \ge 2$. □

THEOREM 10. [illegible] (16) [illegible] (17) [illegible] 9 [illegible] $m \ge 2$ [illegible] $\mathtt{F}(B, n, E_{\min}, E_{\max})$ [illegible] $E_{\min} < -2m < -n < 0 \le E_{\max}$ [illegible]

$$N := 2 + B^{n-m}(1-B^{-1}) \tag{19}$$

[illegible] N [illegible] $(r_1, \ldots, r_N) = \vec{r}(m, B, n, E_{\min}, E_{\max}) \in \mathtt{F}_n^N$ [illegible]

$$\lim_{E_{\min}\to-\infty} |s - s_N| = \left[1 + B(N-2) - B^{1+n-2m}\right] \mathtt{ulp}_n(s_N), \tag{20}$$

[illegible] m [illegible] n [illegible] $n-m$ [illegible]

$$\lim_{\substack{m,n,n-m\to\infty \\ E_{\min}+n\to-\infty}} \frac{|s - \mathtt{round}_m(s_N)|}{\mathtt{ulp}_m\,[\mathtt{round}_m(s_N)]} = B. \tag{21}$$

[illegible]. For any m and n with $2 \le m < n < 2m$, define $r_1, r_2 \in \mathtt{F}_m$ by

$$\begin{aligned} r_1 &:= 1 + B^{1-m} \in \mathtt{F}_m, \\ r_2 &:= -B^{-m}(1+B^{1-m}) \in \mathtt{F}_m. \end{aligned}$$

Then $r_1 + r_2 = 1 + B^{1-m} - (B^{-m} + B^{1-2m})$ takes the algebraically equivalent form

$$r_1 + r_2 = 1 + \left[B^{n-m}(1-B^{-1}) - 1\right] B^{1-n} + (B^{1-n} - B^{1-2m}),$$

where $1 + B^{1-m} - (B^{-m} + B^{1-n}) \in \mathtt{F}_n$ and $B^{1-n} - B^{1-2m}$ gets truncated. Hence

$$s_2 = r_1 \oplus_n r_2 = 1 + \left[B^{n-m}(1-B^{-1}) - 1\right] B^{1-n}$$

if $\oplus_n$ rounds faithfully toward zero (truncates). Then let

$$L := B^{n-m}(1 - B^{-1}) - 1,$$
$$N := 2 + B^{n-m}(1 - B^{-1}),$$

and for each $i \in \{1, \ldots, N-2\}$, define the next summands

$$r_{2+i} := -B^{E_{\min}}.$$

Hence truncating subtractions gives

$$\begin{aligned} s_{2+i} &= 1 + (L - i)B^{1-n}, \\ s_{2+L} &= 1, \\ s_N &= 1 - B^{-n}, \\ \texttt{round}_m(s_N) &= 1 - B^{-m}. \end{aligned} \tag{22}$$

However, the exact sum of all the summands is

$$s = 1 + LB^{1-n} + (B^{1-n} - B^{1-2m}) - (N-2)B^{E_{\min}}.$$

Consequently, before the final rounding, the total error amounts to

$$\begin{aligned} s - s_N &= LB^{1-n} + (B^{1-n} - B^{1-2m}) + B^{-n} - (N-2)B^{E_{\min}} \\ &= \left[B(N-2) + 1 - B^{1+n-2m}\right] \cdot \texttt{ulp}_n(s_N) - (N-2)B^{E_{\min}}. \end{aligned} \tag{23}$$

Formula (23) establishes (20), and also yields

$$B(N-2) \cdot \texttt{ulp}_n(s_N) \le \lim_{E_{\min} \to -\infty} (s - s_N) \le [1 + B(N-2)] \cdot \texttt{ulp}_n(s_N),$$

which corroborates the sharpness of (16). After the final rounding,

$$\begin{aligned} s - \texttt{round}_m(s_N) &= LB^{1-n} + (B^{1-n} - B^{1-2m}) + B^{-m} - (N-2)B^{E_{\min}} \\ &= B(1 - B^{-m})B^{-m} - (N-2)B^{E_{\min}} \\ &= B(1 - B^{-m}) \cdot \texttt{ulp}_m\left[\texttt{round}_m(s_N)\right] - (N-2)B^{E_{\min}}. \end{aligned}$$

Yet $\texttt{ulp}_m\left[\texttt{round}_m(s_N)\right] = B^{-m}$ by (22), whence (19) gives

$$\frac{|s - \texttt{round}_m(s_N)|}{\texttt{ulp}_m\left[\texttt{round}_m(s_N)\right]} = B(1 - B^{-m}) - (1 - B^{-1})B^{n+E_{\min}}. \tag{24}$$

Equation (24) shows that (17) is sharp in the sense of the limit (21).

The conditions for this limit ($m \to \infty$ and $n \to \infty$, with $n - 2m \to -\infty$ and also $E_{\min} + n \to -\infty$) hold, for instance, with $n = m + \lfloor m/2 \rfloor$ and $E_{\min} \le -2n$. □

5. This example shows that if the number of summands exceeds the upper bound (12), then all the significant digits in the rounded sum can be inaccurate. To this end, let $B := 2$, $m := 2$, $n := 3$, $E_{\min} \ll -n$, and

$$N := 4 = 3 + 2^{3-2}(1 - 2^{-1}) = \left\lceil 2 + 2^{3-2}(1 - 2^{-1})/(1 - 2^{-2}) \right\rceil.$$

With binary notation, define $r_1 := 1.1$, $r_2 := -0.001$, and $r_3 := r_4 := -2^{E_{\min}}$. If $\oplus_3$ rounds toward zero, then $s_2 = 1.01$ because $r_1 + r_2 = 1.011$, whence $s_3 = 1.00$ and

TABLE 1
Maximum value of $N = \lfloor 2 + B^{n-m}(1 - B^{-1})/(1 - B^{-m}) \rfloor$ *as a function of* B, m, *and* n.

SYSTEM [SOURCE]	B	"SINGLE" NAME[a]	m	"EXTENDED" NAME	n	$N = \lfloor 2 + B^{n-m}(1 - B^{-1})/(1 - B^{-m}) \rfloor$
IEEE Std 754 [14]	2	single	24	sgl. x.[b]	32	130
IEEE Std 754 [14]	2	single	24	double	53	268,435,474
IEEE Std 754 [14]	2	single	24	dbl. x.[c]	64	549,755,846,658
IEEE Std 754 [14]	2	sgl. x.	32	double	53	1,048,578
IEEE Std 754 [14]	2	sgl. x.	32	dbl. x.	64	2,147,483,650
IEEE Std 754 [14]	2	double	53	dbl. x.	64	1,026
DEC VAX 11 [7]	2	`F`	24	`G`	53	268,435,474
DEC VAX 11 [7]	2	`F`	24	`D`	56	2,147,483,778
DEC VAX 11 [7]	2	`F`	24	`H`	113	309,485,028,268,090,241,946,025,986
DEC VAX 11 [7]	2	`G`	53	`D`	56	6
DEC VAX 11 [7]	2	`G`	53	`H`	113	576,460,752,303,423,554
DEC VAX 11 [7]	2	`D`	56	`H`	113	72,057,594,037,927,939
GE-625/635 [9]	2	single	27	double	63	34,359,738,626
HP 2116 [10]	2	single	23	double	39	32,770
HP 3000 [11]	2	single	23	double	55	2,147,483,906
Honeywell 6000 [13]	2	single	27	double	63	34,359,738,626
Raytheon 706 [27]	2	single	22	middle	30	130
GEMINI [3] [d]	2	min.[e]	7	max.[f]	111	$> 10^{31}$
Burroughs 6900 [2]	8	single	13	double	26	481,036,337,154
GEMINI [3] [g]	10	min.	1	max.	7	1,000,002
HP 28 & 48 [12]	10	single	12	`DOT`[h]	15	902
IBM Syst./360 [18]	16	short	6	long	14	4,026,532,082
IBM Syst./370 [20]	16	short	6	extended	28	290,142,214,001,334,601,824,399,362
IBM Syst./370 [20]	16	long	14	extended	28	67,553,994,410,557,442

[a] The precision name used in the cited source.
[b] Abbreviation in this table for "single extended" in [14].
[c] Abbreviation in this table for "double extended" in [14].
[d] Variable number of digits, chopped, or faithfully rounded, up from trailing 0001.
[e] Minimum number of digits.
[f] Maximum number of digits.
[g] Variable number of digits, chopped, or properly rounded, "traditionally" up from trailing 5.
[h] s_N is not available, but $\texttt{round}_m(s_N) := \vec{\mathbf{r}} \odot \vec{\mathbf{1}}$ with a built-in extend-precision dot-product $\odot$.

$s_4 = 0.111$ similarly. If $\texttt{round}_2$ also rounds toward zero, then $\texttt{round}_2(s_4) = 0.11$ has no accurate significant digit, because $s = 1.011 - 2^{1-E_{\min}}$, whence $s - \texttt{round}_2(s_4) = 0.101 - 2^{1-E_{\min}}$, with $0.101 = (2 + 2^{-1})\texttt{ulp}_2\,[\texttt{round}_2(s_4)]$.

Table 1 lists maximum values of the number N of summands (computed with [illegible] [33]) for systems spanning decades of multiple-precision floating-point arithmetic. Because of faithful—instead of proper—rounding, the values are about an order of magnitude below those listed by Demmel and Hida [5, Table 1, p. 1222].

4. Conclusions. Any extended precision, by any number of additional digits, with any radix and any faithful rounding mode, can increase the accuracy of floating-point sums by several additional correct significant digits.

Acknowledgment. I thank the referees for suggestions that vastly improved the manuscript.

REFERENCES

[1] W. S. BROWN, *A simple but realistic model of floating-point computation*, ACM Trans. Math. Software, 7 (1981), pp. 445–480.

[2] Burroughs Corporation, *B*6900 *System Reference Manual*, Detroit, MI, 1981, http://www.bitsavers.org/pdf/burroughs/B6900/5010986_B6900_sysRef_Jul81.pdf.

[3] Computer Operations, Inc., *GEMINI Computer Systems Principles of Operation*, preliminary ed., 1970, http://www.bitsavers.org/pdf/coi/GEMINI_prelimPrincOps.pdf.

[4] T. J. Dekker, *A floating-point technique for extending the available precision*, Numer. Math., 18 (1971), pp. 224–242.

[5] J. Demmel and Y. Hida, *Accurate Floating Point Summation*, Computer Science Division Technical Report UCB//CSD-02-1180, University of California, Berkeley, CA, 2002, http://www.cs.berkeley.edu/~demmel/AccurateSummation.pdf.

[6] J. Demmel and Y. Hida, *Accurate and efficient floating point summation*, SIAM J. Sci. Comput., 25 (2003), pp. 1214–1248.

[7] Digital Equipment Corporation, *VAX* 11 *Architecture Reference Manual*, Digital Press, Digital Equipment Corporation, Bedford, MA, 1982, http://www.bitsavers.org/pdf/dec/vax/archSpec/EK-VAXAR-RM-001_Arch_May82.pdf.

[8] B. Einarsson, ed., *Accuracy and Reliability in Scientific Computing*, SIAM, Philadelphia, 2005.

[9] General Electric Company, *GE*-625/635 *Programming Reference Manual*, revised ed., General Electric Company, Information Systems Equipment Division, C-83, Phoenix, AZ, 1969, http://www.bitsavers.org/pdf/ge/CPB1004F_GE635pgmRef_Jul69.pdf.

[10] Hewlett-Packard, *Operating and Service Manual* 2152*A Floating Point Processor*, Hewlett-Packard Company, Cupertino, CA, 1970, http://www.bitsavers.org/pdf/hp/2116/02152-90004_2116FPP_sep70.pdf.

[11] Hewlett-Packard, *HP* 3000 *Computer Systems Machine Instruction Set Reference Manual*, Hewlett-Packard Company, Cupertino, CA, 1984, http://www.bitsavers.org/pdf/hp/3000/instrSet/30000-90022_InsSetRef_Jun84.pdf.

[12] Hewlett-Packard Co., *HP* 48*G Series User's Guide*, 7th ed., Part 00048-90126, Corvallis Division, Corvallis, OR, 1994.

[13] Honeywell Information Systems, Inc., *Series* 6000 *Summary Description*, MS 061, Waltham, MA, 1971, http://www.bitsavers.org/pdf/honeywell/series6000/DA48_series6000_summary_1971.pdf.

[14] Institute of Electrical and Electronic Engineers, *IEEE Standard for Binary Floating-Point Arithmetic*, ANSI/IEEE Std 754-1985, New York, 1985.

[15] International Business Machines Corporation, *IBM* 704 *Electronic Data-Processing Machine Manual of Operation*, New York, 1955, http://www.bitsavers.org/pdf/ibm/704/24-6661-2_704_Manual_1955.pdf.

[16] International Business Machines Corporation, *IBM* 7094 *Principles of Operation*, 5th ed., IBM Systems Products Division, Product Publications, Dept. B98, Poughkeepsie, NY, 1962, http://www.bitsavers.org/pdf/ibm/7094/A22-6703-4_7094_PoO_Oct66.pdf.

[17] International Business Machines Corporation, *Reference Manual IBM* 7090 *Data Processing System*, 1962, http://www.bitsavers.org/pdf/ibm/7090/22-6528-4_7090Manual.pdf.

[18] International Business Machines Corporation, *IBM System/*360 *Principles of Operation*, 8th ed., IBM Systems Products Division, Product Publications, Dept. B98, Poughkeepsie, NY, 1968, http://www.bitsavers.org/pdf/ibm/360/poo/A22-6821-7_360PrincOpsDec67.pdf.

[19] International Business Machines Corporation, *IBM System/*370 *Principles of Operation*, 1st ed., IBM Systems Products Division, Product Publications, Dept. B98, Poughkeepsie, NY, 1970, http://www.bitsavers.org/pdf/ibm/370/GA22-7000-0_370PoO_Jun70.pdf.

[20] International Business Machines Corporation, *IBM System/*370 *Principles of Operation*, 4th ed., IBM Systems Products Division, Product Publications, Dept. B98, Poughkeepsie, NY, 1975, http://www.bitsavers.org/pdf/ibm/370/GA22-7000-4_370PoO_Sep75.pdf.

[21] International Business Machines Corporation, *Enterprise Systems Architecture* 390 *Principles of Operation*, 9th ed., Form SA22-7201-8, IBM Systems Products Division, Department 55JA, Poughkeepsie, NY, 2003, http://publibz.boulder.ibm.com/epubs/pdf/dz9ar008.pdf.

[22] S. Linnainmaa, *Analysis of some methods of improving the accuracy of floating-point computations*, BIT, 14 (1974), pp. 167–202.

[23] C. Maxfield and A. Brown, *How Computers Do Math*, Wiley, New York, 2005, http://www.diycalculator.com/.

[24] M. Pichat, *Correction d'une somme en arithmetique à virgule flottante*, Numer. Math., 19 (1972), pp. 400–406.

[25] D. M. Priest, *Algorithms for arbitrary precision floating point arithmetic*, in Proceedings of

the 10th Symposium on Computer Arithmetic, P. Kornerup and D. Matula, eds., IEEE Computer Society Press, Los Alamitos, CA, 1991, pp. 132–143.

[26] D. M. Priest, *On Properties of Floating Point Arithmetics: Numerical Stability and the Cost of Accurate Computations*, Ph.D. thesis, University of California at Berkeley, Berkeley, CA, 1992, ftp://ftp.icsi.berkeley.edu/pub/theory/priest-thesis.ps.Z, http://citeseer.nj.nec.com/priest92propertie.html.

[27] Raytheon Computer, *Raytheon* 706 *Computer User's Manual*, Raytheon Computer, Santa Anna, CA, 1970, http://www.bitsavers.org/pdf/raytheon/Raytheon706usersMan_Feb70.pdf.

[28] S. M. Rump, *Computer-assisted proofs and self-validating methods*, in Accuracy and Reliability in Scientific Computing, B. Einarsson, ed., SIAM, Philadelphia, 2005, Chapter 10, pp. 195–240.

[29] P. H. Sterbenz, *Floating-Point Computation*, Prentice-Hall, Englewood Cliffs, NJ, 1974.

[30] J. E. Tomayko, *Computers in Spaceflight: The NASA Experience*, NASA Contractor Report CP-182505, National Aeronautics and Space Administration, Scientific and Technical Information Division, 1988, http://www.hq.nasa.gov/office/pao/History/computers/contents.html.

[31] M. A. Vouk, *Software-reliability engineering of numerical systems*, in Accuracy and Reliability in Scientific Computing, B. Einarsson, ed., SIAM, Philadelphia, 2005, Chapter 13, pp. 265–299.

[32] G. W. Walster, *The use and implementation of interval data types*, in Accuracy and Reliability in Scientific Computing, B. Einarsson, ed., SIAM, Philadelphia, 2005, Chapter 9, pp. 173–194.

[33] S. Wolfram, *The Mathematica Book*, 3rd ed., Wolfram Media, Champaign, IL, 1996.

SIAM J. SCI. COMPUT.
Vol. 28, No. 5, pp. 1851–1885
© 2006 Society for Industrial and Applied Mathematics

WAVELET FILTERING FOR EXACT CONTROLLABILITY OF THE WAVE EQUATION*

M. NEGREANU†, A.-M. MATACHE‡, AND C. SCHWAB‡

Abstract. In this paper we propose and analyze a wavelet-based numerical method for exact controllability of the one-dimensional wave equation, concentrating on the particular case of *Dirichlet* controls. We use the Hilbert uniqueness method (HUM) and prove this result in the context of the finite element space semidiscretization with spline wavelet basis functions. The algorithm for the numerical construction of the controls is based on a conjugate gradient method proposed by Glowinski [*J. Comput. Phys.*, 103 (1992), pp. 189–221] and a two-grid filtering technique in the wavelet basis. We show that wavelet filtering yields numerically exact controllability in the optimal time of the continuous problem and the convergence of the numerical controls using a multilevel discretization filtering technique. Classical arguments then allow proving the uniform boundedness of the numerical controls and passing to the limit as the mesh size tends to zero. We prove that the condition number of the discretized HUM operator Λ with wavelet filtering is uniformly bounded with respect to the mesh width and prove convergence of the numerical controls obtained with the two-grid filtering technique. Numerical experiments show that wavelet filtering yields numerical controls with the same control time T as in the continuous case.

Key words. boundary controllability, uniform observability, multigrid algorithm, finite element discretization, wavelet filtering, wave equation

AMS subject classifications. 65N30, 65N55, 35L05, 35L20, 93B05, 93B07

DOI. 10.1137/050622894

1. Introduction.

In the present paper, we consider the numerically exact controllability of wave equations, the simplest example of which is the constant coefficient one-dimensional wave equation:

$$\begin{cases} u_{tt} - u_{xx} = 0, & 0 < x < 1, \quad 0 < t < T, \\ u(0,t) = 0, \quad u(1,t) = v(t), & 0 < t < T, \\ u(x,0) = u^0(x), \qquad u_t(x,0) = u^1(x), & 0 < x < 1. \end{cases} \tag{1.1}$$

In (1.1), $u = u(x,t)$ describes the displacement of a vibrating string occupying the interval $(0,1)$, (u^0, u^1) is the given initial state, and v is an unknown control function that acts on the system through the end point $x = 1$. System (1.1) is well posed in appropriate functional spaces.

When no control acts on the boundary $x = 1$, i.e., $v = 0$, the total energy

$$E(t) = \frac{1}{2} \int_0^1 \left(|u_x(x,t)|^2 + |u_t(x,t)|^2 \right) dx \tag{1.2}$$

*Received by the editors January 17, 2005; accepted for publication (in revised form) June 1, 2006; published electronically November 14, 2006. This research was supported by the project *Homogenization and Multiple Scales* (HMS) 2000 of EC, contract HPRN-CT-2000-00109, and was performed while the first author visited the Seminar for Applied Mathematics of ETH Zürich in 2002–2003.

http://www.siam.org/journals/sisc/28-5/62289.html

†Department of Algebra, Faculty of Mathematics, University Complutense of Madrid, 28040, Madrid, Spain (mihaela_negreanu@mat.ucm.es).

‡Seminar for Applied Mathematics, ETH-Zentrum CH-8092 Zürich, Switzerland (amatache@math.ethz.ch, schwab@math.ethz.ch).

of the solution is conserved, i.e.,

$$\frac{dE(t)}{dt} = E'(t) = 0 \quad \forall t \in [0,T].$$

The following exact controllability result is well known (see, e.g., [7]): [illegible] $T \geq 2$ [illegible] $(u^0, u^1) \in L^2(0,1) \times H^{-1}(0,1)$ [illegible] $v \in L^2(0,T)$ [illegible] $u = u(x,t)$ [illegible] (1.1) [illegible]

$$u(T) = u_t(T) = 0, \quad 0 < x < 1. \tag{1.3}$$

The goal in this control problem is therefore to drive solutions to equilibrium at time $t = T$. Once the equilibrium configuration is reached at time $t = T$, by taking null controls for $t \geq T$, i.e., $v = 0$ for $t \geq T$, the solution remains at rest for all $t \geq T$.

Let us now introduce the homogeneous system

$$\begin{cases} \phi_{tt} - \phi_{xx} = 0, & 0 < x < 1, \quad 0 < t < T, \\ \phi(0,t) = \phi(1,t) = 0, & 0 < t < T, \\ \phi(x,0) = \phi^0(x), \qquad \phi_t(x,0) = \phi^1(x), & 0 < x < 1. \end{cases} \tag{1.4}$$

It admits a unique solution $\phi = \phi(x,t) \in \mathcal{C}^0\left([0,T], H_0^1(0,1)\right) \cap \mathcal{C}^1\left([0,T], L^2(0,1)\right)$ when $(\phi^0, \phi^1) \in H_0^1(0,1) \times L^2(0,1)$. This system is the adjoint of (1.1) up to an inversion in time.

The exact controllability of (1.1) is reduced to the study of the observability properties of the adjoint system: [illegible] $T \geq 2$ [illegible] $C(T) > 0$ [illegible]

$$E(0) \leq C(T) \int_0^T \left| \frac{\partial \phi(1,t)}{\partial x} \right|^2 dt \tag{1.5}$$

[illegible] $\phi = \phi(x,t)$ [illegible]. We refer to Lions [7] for a detailed analysis of the equivalence between controllability and observability through the [illegible] (HUM).

For any given $(u^0, u^1) \in L^2(0,1) \times H^{-1}(0,1)$, $T > 0$, there may exist many controls v with (1.3). Among all controls with the property (1.3) there exists a unique control of minimal L^2-norm, the so-called HUM control. Note that (1.3) makes sense, since in these conditions (1.1) admits a unique solution $u \in \mathcal{C}([0,T]; L^2(0,1)) \cap \mathcal{C}^1([0,T]; H^{-1}(0,1))$. Once we know that such a control exists, it is natural to try to construct this control numerically. A possible procedure is to approximate the wave operator by some sequence of discrete or semidiscrete operators and to obtain the control of system (1.1) as the limit of the sequence of controls of the approximating equations.

It is a well-known fact that the interaction of waves with the numerical mesh produces dispersion phenomena and spurious high-frequency oscillations [15, 16]. The observation and control properties of the discrete model eventually disappear. This has some impact when computing the control for the continuous model.

The goal of this paper is to obtain a convergent numerical approximation of the exact control.

For the space semidiscrete system, one method for constructing a convergent sequence of computed controls consists in filtering the high frequencies by truncating

the Fourier series of the solution. This method is natural from a theoretical point of view, but rather hard to implement in numerical simulations [6, 17].

Another method was proposed by Glowinski et al. in [4, 5]. They employ a Tychonoff regularization procedure which allows one to uniformly control the semidiscrete system at the price of adding an extra internal control such that the boundary component of the controls converges to the boundary control of the continuous wave equation. This method is efficient to compute.

Glowinski in [4] introduced a two-grid algorithm that also allows one to compute efficiently the control of the continuous model. It consists in using a coarse and a fine grid, and to project the initial data from the fine grid to the coarse grid. With that strategy, the highest frequencies cannot exist alone. The convergence of this method has been proved in [10, 11] for $T = 4$ for the finite difference and finite element semidiscrete approximation in one space dimension with a discrete multiplier approach. In [8], Loreti and Mehrenberger have proposed a new proof of this fact thanks to an Ingham-type approach. This leads to the critical time $T \geq 2$.

In the full one-dimensional space-time discretization with equal space and time step, the high-frequency pathologies no longer arise and the HUM controls for the discrete model (with finite difference and finite element approximation) converge to the control of the continuous wave equation. The detailed proof of these results is given in [9, 11]. When $\Delta t/\Delta x = \alpha < 1$, or in the semidiscrete case, it is easy to check that both the observability constant and the computed controls diverge [6, 12, 13].

In this paper we propose an algorithm to approximate the HUM control based on the conjugate gradient method proposed by Glowinski [4] and multilevel filtering in a hierarchical spline wavelet basis. We obtain convergence of the numerical controls computed by the multilevel filtering. In Theorem 3.4 we prove the convergence of the numerical controls computed by the multigrid filtering.

This paper is organized as follows: In section 2 we recall the HUM algorithm and discuss its application to the exact boundary controllability of the wave equation (1.1). In section 3 we discuss the application of this algorithm in the semidiscrete case using finite element spaces of spline wavelets as approximating spaces. Moreover, we analyze the conjugate gradient method with multigrid filtering and obtain the convergence of this procedure for the finite element controls. We also prove that the condition number of the discretized HUM operator corresponding to wavelet filtering is uniformly bounded with respect to the mesh width. In section 4 we present some numerical results.

2. The HUM method and the operator Λ. We describe briefly a method due to Lions called HUM (the Hilbert uniqueness method) for control of a wave equation from a part of the boundary $x = 1$; for more details see [1, 4, 5].

2.1. Operator Λ. Let us denote by $E = H_0^1(0,1) \times L^2(0,1)$ the space of initial states for the homogeneous wave equation. Further let $E' = H^{-1}(0,1) \times L^2(0,1)$ be the dual space of E, and define the operator

$$\Lambda : E \to E'$$

as follows:

- Given $e = \{\phi^0, \phi^1\} \in E$, let ϕ be the solution of the uncontrolled wave equation (1.4) from $t = 0$ to $t = T$ with initial state e.
- Let ψ be the solution of the backward controlled system from $t = T$ to $t = 0$

with boundary data $\frac{\partial\phi(1,t)}{\partial x}$ at the end point $x = 1$:

$$\text{(2.1)}\quad \begin{cases} \psi_{tt} - \psi_{xx} = 0, & 0 < x < 1, \quad 0 < t < T, \\ \psi(0,t) = 0 \qquad \psi(1,t) = \dfrac{\partial\phi(1,t)}{\partial x}, & 0 < t < T, \\ \psi(x,T) = \psi_t(x,T) = 0, & 0 < x < 1. \end{cases}$$

- Define $\Lambda\, e$ as

$$\text{(2.2)}\qquad \Lambda\, e = \{\psi_t(0), -\psi(0)\}\,.$$

Using the HUM approach, the original control problem (1.3) is transformed into an identification of initial conditions problem, and the linear problem (2.2) is in duality with a minimization problem,

$$\min_{\mathcal{U}} \int_0^T v^2 dt$$

(see [7, 4] for more details), where the set of admissible controls is given by

$$\mathcal{U} = \{v \in L^2(0,T) \,|\, y_{tt} - y_{xx} = 0 \,\text{in}\, (0,1)\times(0,T),\ y(0,t) = 0,\ y(1,t) = v \,\text{in}\, (0,T),$$
$$y(x,0) = u^0(v),\ y_t(x,0) = u^1(x),\ y(x,T) = y_t(x,T) = 0\}.$$

The HUM control is the unique solution of this minimization problem.

Following this systematic way for constructing such control v provided by the HUM approach, the original exact controllability problem (1.3) is transformed into the problem of identifying initial values for an associated adjoint wave equation.

2.2. HUM algorithm. We discuss now an application to the exact boundary controllability of the wave equation (1.1) for the initial conditions (u^0, u^1). To this end, $f = \{u^1, -u^0\} \in E'$ and let $e = \{\phi^0, \phi^1\} \in E$ be such that

$$\text{(2.3)}\qquad \Lambda\, e = f.$$

The linear problem (2.3) admits a unique solution, provided that T is sufficiently large [7]. The optimal control v is then obtained as follows: take $e = \{\phi^0, \phi^1\}$ as the initial state in (1.4) and define v as $v = \partial\phi(1,t)/\partial x$. This yields the HUM control v which satisfies the exact controllability property. Indeed, solving the wave equation (2.1) backward in time using v as the boundary data in $x = 1$ and denoting by ψ its solution, $u = \psi$ satisfies the following by the definition of the operator Λ: $u(0,x) = u^0(x)$, $u_t(0,x) = u^1(x)$, and $u(x,T) = u_t(x,T) = 0$ for all $x \in (0,1)$.

We can rewrite (2.3) in the following symmetric variational form [1, 5]: given $f \in E'$,

$$\text{(2.4)}\qquad \text{find } e \in E \text{ such that } \langle \Lambda\, e, \bar{e}\rangle = \langle f, \bar{e}\rangle \quad \forall\, \bar{e} \in E,$$

where $\langle\cdot,\cdot\rangle$ denotes the duality pairing between E and E'.

Integrating by parts in time and space, the following representation of the form $\langle \Lambda\, e, \bar{e}\rangle$ is obtained (see [7, Chapter 1, section 2]):

$$\text{(2.5)}\qquad \langle \Lambda\, e, \bar{e}\rangle = \int_0^1 \left[\psi_t(x,0)\bar{\phi}(x,0) - \psi(x,0)\bar{\phi}_t(x,0)\right] dx = \int_0^T \left[\frac{\partial\phi}{\partial x}(1,t)\frac{\partial\bar{\phi}}{\partial x}(1,t)\right] dt$$

for all e, $\bar{e} \in E$, where $\int_0^1$ is understood as an extension of duality pairing, and $\{\phi, \psi\}$, $\{\bar{\phi}, \bar{\psi}\}$ are the solutions of (1.4)–(2.1) with initial states e and $\bar{e}$, respectively.

2.3. Properties of Λ. We collect here some properties of the operator Λ. It is proved in [7, Chapter 1, section 2] that Λ is a linear and continuous operator from E into E'; moreover, if T is sufficiently large ($T \geq T_{\min} = 2$), Λ is an isomorphism from E onto E'. If $T \geq T_{\min} = 2$, it follows from (2.5) that Λ is coercive, and symmetric, i.e., there exist constants $C > 0$ and $\gamma > 0$ such that

$$|\langle \Lambda e, \bar{e} \rangle| \leq C \|e\|_E \|\bar{e}\|_E \quad \forall\, e, \bar{e} \in E, \tag{2.6}$$

$$\langle \Lambda e, e \rangle \geq \gamma \|e\|_E^2 \quad \forall\, e \in E, \tag{2.7}$$

where for $e = \{\phi^0, \phi^1\} \in E$

$$\|e\|_E := \left(\int_0^1 (|\nabla \phi^0|^2 + |\phi^1|^2)\, dx \right)^{1/2}. \tag{2.8}$$

The combination of coercivity with the symmetry is of fundamental importance, since it implies that (2.3) can be solved by a conjugate gradient algorithm operating in the space E.

The operator Λ is composed of the solution of the two wave equations (1.4) and (2.1). From the analysis given in [5], after discretization of these equations, one in general does not end up with a positive definite matrix: it appears that the approximation of $\partial\phi(1,t)/\partial x$, which is used to construct the discrete operator Λ_h, introduces a very strong damping of the large wave number components of discrete initial data e_h (parameter h describes the size of the space mesh). This is related to the fact that the velocity of propagation of solutions tends to zero as the size of the mesh tends to zero and the wavelength of solutions is of the same order as the size of the mesh (see [12, 13]).

2.4. Conjugate gradient implementation of HUM. The outline of this section is as follows: after a short recapitulation of the conjugate gradient algorithms for the numerical solution of linear variational problems we specify its application to the boundary control of the wave equation.

2.4.1. Conjugate gradient algorithm: Generalities. As we have seen in section 2.1, the original control problem is transformed into a resolution of the equation $\Lambda e = f$, which is a particular case of the abstract linear variational problems,

$$u \in V, \qquad a(u, v) = L(v) \qquad \forall v \in V, \tag{2.9}$$

where in (2.9) the following hold:

1. V is a real Hilbert space for the scalar product $(\cdot, \cdot)$ and the corresponding norm $\|\cdot\|$.
2. $a : V \times V \to V$ is bilinear, symmetric, continuous, and V-elliptic.
3. $L : V \to \mathbb{R}$ is linear and continuous.

With the above hypotheses, problem (2.9) has a unique solution, which can be computed by the following conjugate gradient algorithm (see [5] for more details).

- Step 0. Initialization.

  With $u^0 \in V$ given (2.10)

  solve

  $$g^0 \in V, \ \ (g^0, v) = a(u^0, v) - L(v) \quad \forall v \in V. \tag{2.11}$$

If $g^0 = 0$ or is [illegible], take $u = u^0$; if not, set

$$w^0 = g^0. \tag{2.12}$$

Then for $n \geq 0$, assuming that u^n, g^n, w^n are known, compute u^{n+1}, g^{n+1}, w^{n+1} as follows.

- Step 1. [illegible] Compute

$$\rho_n = \frac{\|g^n\|^2}{a(w^n, w^n)}, \tag{2.13}$$

and then

$$u^{n+1} = u^n - \rho_n w^n. \tag{2.14}$$

- Step 2. [illegible]
Solve

$$g^{n+1} \in V,$$

$$(g^{n+1}, v) = (g^n, v) - \rho_n a(w^n, v) \quad \forall v \in V; \tag{2.15}$$

if $g^{n+1} = 0$ or is [illegible], take $u = u^{n+1}$; if not, compute

$$\gamma_n = \frac{\|g^{n+1}\|^2}{\|g^n\|^2}, \tag{2.16}$$

and define the new descent direction by

$$w^{n+1} = g^{n+1} + \gamma_n w^n. \tag{2.17}$$

Take $n = n + 1$ and go to (2.13).

If u is the solution of problem (2.9), we also have (see [5])

$$\|u^n - u\| \leq C\|u^0 - u\| \left(\frac{\sqrt{\nu_a} - 1}{\sqrt{\nu_a} + 1}\right)^n, \tag{2.18}$$

where the [illegible] ν_a of $a(\cdot, \cdot)$ is defined by

$$\nu_a = \sup_{v \in V - \{0\}} \frac{a(v, v)}{\|v\|^2} : \inf_{v \in V - \{0\}} \frac{a(v, v)}{\|v\|^2}.$$

Back to the practical implementation of algorithm, by g^0 and g^{n+1} [illegible] we mean that g^0 and g^{n+1} satisfy

$$\frac{\|g^0\|}{\|u^0\|} \leq \varepsilon, \quad \frac{\|g^{n+1}\|}{\|g^0\|} \leq \varepsilon,$$

where ε is a small positive number depending upon the floating point arithmetic used by the computer (see [5] for more details).

2.4.2. Conjugate gradient solution of $\Lambda e = f$. The HUM reduces the calculation of the control of minimal L^2-norm for the continuous problem (1.1) to the resolution of $\Lambda e = f$, with $f = \{-u^1, u^0\}$ given and $e \in E = H_0^1(0,1) \times L^2(0,1)$.

Application of the general conjugate gradient algorithm (2.10)–(2.17) to the solution of $\Lambda e = f$ is possible if $T \geq 2$; indeed, we take

$$V = E,$$

$$a(\cdot,\cdot) = \langle \Lambda \cdot, \cdot \rangle, \qquad L : \check{e} \to \langle \{u^1, -u^0\}, \check{e} \rangle.$$

We obtain then the following conjugate gradient algorithm.

- Step 0.

(2.19) With $e_0^0 \in H_0^1(0,1)$ and $e_0^1 \in L^2(0,1)$ given,

solve

$$\text{(2.20)} \qquad \begin{cases} (\phi_0)_{tt} - (\phi_0)_{xx} = 0, & 0 < x < 1, \quad 0 < t < T, \\ \phi_0 = 0, & 0 < t < T, \\ \phi_0(0) = e_0^0, \qquad (\phi_0)_t(0) = e_0^1 \end{cases}$$

and

$$\text{(2.21)} \qquad \begin{cases} (\psi_0)_{tt} - (\psi_0)_{xx} = 0, & 0 < x < 1, \quad 0 < t < T, \\ \psi_0(0) = 0, \quad \psi_0(1) = (\phi_0)_x(1,t), & 0 < t < T, \\ \psi_0(T) = (\psi_0)_t(T) = 0, & 0 < x < 1. \end{cases}$$

Compute $g_0 = \{g_0^0, g_0^1\} \in E$,

$$\text{(2.22)} \qquad -(g_0^0)_{xx} = (\psi_0)_t(0) - u^1 \text{ in } (0,1), \quad g_0^0(0) = g_0^0(1) = 0,$$

$$\text{(2.23)} \qquad g_0^1 = u^0 - \psi_0(0),$$

respectively. Then set

$$\text{(2.24)} \qquad w_0 = g_0.$$

Now, for $n \geq 0$, assuming that e_n, g_n, w_n, ϕ_n, ψ_n are known, compute e_{n+1}, g_{n+1}, w_{n+1}, ϕ_{n+1}, ψ_{n+1} as follows:

- Step 1. Solve the systems

$$\text{(2.25)} \qquad \begin{cases} (\phi_n)_{tt} - (\phi_n)_{xx} = 0, & 0 < x < 1, \quad 0 < t < T, \\ \phi_n = 0, & 0 < t < T, \\ \phi_n(0) = w_n^0, \qquad (\phi_n)_t(0) = w_n^1 \end{cases}$$

and

$$\text{(2.26)} \qquad \begin{cases} (\psi_n)_{tt} - (\psi_n)_{xx} = 0, & 0 < x < 1, \quad 0 < t < T, \\ \psi_n(0) = 0, \quad \psi_n(1) = (\phi_n)_x(1,t), & 0 < t < T, \\ \psi_n(T) = (\psi_n)_t(T) = 0, & 0 < x < 1, \end{cases}$$

$$-(g_n^0)_{xx} = (\psi_n)_t(0) \text{ in } (0,1), \quad g_n^0(0) = g_n^0(1) = 0 \tag{2.27}$$

and set

$$g_n^1 = -\psi_n(0). \tag{2.28}$$

Now compute

$$\rho_n = \frac{\int_0^1 (|(g_n^0)_x|^2 + |g_n^1|^2)dx}{\int_0^1 ((g_n^0)_x (w_n^0)_x + g_n^1 w_n^1)dx}, \tag{2.29}$$

$$e_{n+1} = e_n - \rho_n w_n, \tag{2.30}$$

$$\phi_{n+1} = \phi_n - \rho_n \phi_n, \tag{2.31}$$

$$\psi_{n+1} = \psi_n - \rho_n \psi_n, \tag{2.32}$$

$$g_{n+1} = g_n - \rho_n g_n. \tag{2.33}$$

- Step 2. [illegible]
  If $g_{n+1} = 0$ or is [illegible], take $e = e_{n+1}$, $\phi = \phi_{n+1}$, $\psi = \psi_{n+1}$; if not, compute

$$\gamma_n = \frac{\int_0^1 (|(g_{n+1}^0)_x|^2 + |g_{n+1}^1|^2)dx}{\int_0^1 (|(g_n^0)_x|^2 + |g_n^1|^2)dx}, \tag{2.34}$$

  and define the new descent direction by

$$w_{n+1} = g_{n+1} + \gamma_n w_n. \tag{2.35}$$

  Set $n := n + 1$ and go to (2.29).

3. Wavelet HUM. We compute numerical solutions to (2.4) by Galerkin approximation: we restrict trial and test space E in (2.4) to finite-dimensional subspaces $E_h \subset E$ of finite element type. This procedure was proposed by Glowinski and Lions in [5]. It will lead to convergent approximations if (a) the sequence $\{E_h\}_h$ of subspaces is dense in E, and (b) if the operator Λ is realized exactly on E_h.

In practice, however, Λ [illegible] be realized exactly for two reasons: (a) the forward/backward wave equations (1.4), (2.1) must be integrated numerically by a suitable time stepping scheme, and (b) if (1.4), (2.1) are solved with initial data in the finite element space E_h, the exact solution does [illegible] belong to E_h: (1.4), (2.1) must be solved with a higher spatial resolution than the given initial data and must be projected back into E_h. As observed in [4], this projection back into the "coarse" space of initial data amounts to a filtering.

Various filtering techniques were proposed in [4]. The key requirement of uniform in h stability in the topology of E for the filtering is difficult to satisfy. For various choices proposed in [4], discrete controls with an exact controllability time $T_{\min} = 4 > T = 2$ result for (1.1) (see [10, 11] for more details).

Here, we propose space discretization of (1.4), (2.1) in a hierarchical spline wavelet basis. While this yields the usual finite spaces for Galerkin approximation of (2.1), in this basis an E-stable filtering is immediately obtained by truncating the wavelet expansion, respectively, by the quasi-interpolation operator corresponding to this truncation. The efficiency of this computational methodology is illustrated by the numerical experiments below in which we obtain the optimal observability time $T \geq T_{\min} = 2$ also for the numerical controls.

3.1. Space discretization. Given $\Omega = (0,1)$, $N \in \mathbb{N}$, we set $h = 1/(N+1)$ and consider the mesh $\mathcal{T}_h$

$$0 = x_0 < x_1 < \cdots < x_{N+1} = 1,$$

with $x_j = jh$, $j = 0, \ldots, N+1$.

Let V_h be the finite element space of continuous, piecewise linear functions on $\mathcal{T}_h$ vanishing at end points. Let $\phi_h(t) \in V_h$ denote the solution of the semidiscrete problem

$$\begin{cases} \dfrac{d^2}{dt^2}\left(\phi_h(t),\, v\right)_{L^2(\Omega)} + \left(\dfrac{\partial \phi_h(t)}{\partial x},\, \dfrac{dv}{dx}\right)_{L^2(\Omega)} = 0 \qquad \forall v \in V_h, \\ \phi_h(0) = \phi_h^0 \in V_h, \\ \phi_h'(0) = \phi_h^1 \in V_h, \end{cases} \tag{3.1}$$

with ϕ_h^0, ϕ_h^1 being the L^2-projections into V_h of ϕ^0, ϕ^1, respectively:

$$(\phi^0 - \phi_h^0, v)_{L^2(\Omega)} = (\phi^1 - \phi_h^1, v)_{L^2(\Omega)} = 0 \quad \forall\, v \in V_h.$$

If we expand both $\phi_h(t)$ and the test function v with respect to a basis $\mathcal{B} = \{\Phi_j\}_{j=1}^N$ of $V_h = \operatorname{span} \mathcal{B}$, (3.1) can be rewritten in matrix form as

$$\begin{cases} \mathbf{M} \dfrac{d^2}{dt^2} \underline{\phi}_h(t) + \mathbf{A} \underline{\phi}_h(t) = 0, \qquad 0 < t < T, \\ \underline{\phi}_h(0) = \underline{\phi}_h^0, \\ \underline{\phi}_h'(0) = \underline{\phi}_h^1. \end{cases} \tag{3.2}$$

Here we denote by $\underline{v}$ the coefficient vector of the function $v \in V_h$ with respect to the basis $\mathcal{B}$, $\mathbf{A}$ represents the stiffness matrix, and $\mathbf{M}$ is the mass matrix:

$$A_{i,j} := (\Phi_j', \Phi_i')_{L^2(\Omega)}, \quad M_{i,j} := (\Phi_i, \Phi_j)_{L^2(\Omega)} \quad \forall\, i,j = 1, \ldots, N.$$

The simplest choice for the basis $\{\Phi_i\}_{i=1}^N$ is to take the so-called hat functions $\Phi_i = \max(1 - |x - x_i|/h, 0)$. The standard hat functions do not build a hierarchical basis.

The hierarchical hat functions are obtained as follows: for $\Omega = (0,1)$, let ϕ be the hat function taking values $0, 1, 0$ at $0, 1/2, 1$, respectively, and vanishing outside Ω. We set $\mathcal{I}_l := \{0, 1, \ldots, 2^l - 1\}$, $\psi_j^l(x) := 2^{-l/2}\phi(2^l x - j)$, $j \in \mathcal{I}_l$, $x \in \Omega$, and denote $\mathcal{B}_L := \left\{\psi_j^l\right\}_{l=0,\ldots,L\ j \in \mathcal{I}_l}$. Then $\mathcal{B}_L$ is a basis of V_h for $h = 2^{-L}$, which is [illegible]:

$$\mathcal{B}_0 \subset \mathcal{B}_1 \subset \cdots \subset \mathcal{B}_L \subset \cdots.$$

The total energy of the space-semidiscrete problem (3.2) is given by

$$E_h(t) = \frac{1}{2}\left[\underline{\phi}_h'^{\top} \mathbf{M} \underline{\phi}_h' + \underline{\phi}_h^{\top} \mathbf{A} \underline{\phi}_h\right]. \tag{3.3}$$

PROPOSITION 3.1. [illegible] (3.3) [illegible] (3.2) [illegible]. Similarly to the continuous case, we multiply (3.2) by $\underline{\phi}_h' = \frac{d}{dt}\phi_h(t)$ and obtain that

$$\underline{\phi}_h'^{\top} \mathbf{M} \underline{\phi}_h'' = -\underline{\phi}_h'^{\top} \mathbf{A} \underline{\phi}_h.$$

Equivalently,

$$\frac{d}{dt}\left(\underline{\phi}_h'^{\top}\,\mathbf{M}\underline{\phi}_h' + \underline{\phi}_h^{\top}\,\mathbf{A}\underline{\phi}_h\right) = 0,$$

and therefore $\frac{dE_h(t)}{dt} = 0$ for all $t \in (0,T)$. ∎

3.2. Wavelet basis. For filtering and preconditioning purposes in a numerical version of HUM, it will be essential to employ a wavelet basis of V_h, which we now describe.

In the interval $\Omega = (0,1)$ we create a sequence $\{\mathcal{T}^l\}$ of meshes by $\mathcal{T}^0 = \{\Omega\}$ and by halving the elements $I_j^{l-1} \in \mathcal{T}^{l-1}$ to get $\mathcal{T}^l$. To make level dependence explicit, we denote by $\mathcal{V}_l$ the space of continuous, piecewise linears on $\mathcal{T}^l$ that vanish at the end points. The mesh width of $\mathcal{T}^l$ is $h_l = 2^{-l}$, and we write $N_l = \dim\mathcal{V}_l$, $M_l := N_l - N_{l-1}$, $N_{-1} := 0$. Then $N_l = 2^{l+1} - 1$, $l = 0, 1, 2, \ldots$.

We assume we are given a spline wavelet basis ψ_j^l, $j = 1, \ldots, M_l$, $l = 0, 1, 2, \ldots, L$, of

$$V_h := \mathcal{V}_L$$

with the properties

$$\mathcal{V}_L = \operatorname{span}\left\{\psi_j^l \mid 0 \le l \le L;\ 1 \le j \le M_l\right\}, \quad \operatorname{diam}\operatorname{supp}\,(\psi_j^l) \le C\,2^{-l}, \tag{3.4}$$

such that ψ_j^l for $l \ge l_0$ are obtained by scaling and translation of the function $\psi_j^{l_0}$.

Any function $v \in V_h = \mathcal{V}_L$ has the representation

$$v = \sum_{l=0}^{L}\sum_{j=1}^{M^l} v_j^l \psi_j^l, \tag{3.5}$$

with wavelet coefficients $v_j^l \in \mathbb{R}$. For $v \in H_0^1(\Omega)$ one obtains the series

$$v = \sum_{l=0}^{\infty}\sum_{j=1}^{M_l} v_j^l \psi_j^l, \tag{3.6}$$

which converges in $L^2(\Omega)$ and in $H_0^1(\Omega)$. Moreover, we assume that in the scale of spaces

$$H_\theta = \left(L^2(\Omega),\, H_0^1(\Omega)\right)_{\theta,2}$$

the following [illegible] holds:

$$\forall\, v \in H_\theta,\ c_1\|v\|_{H_\theta}^2 \le \sum_{l=0}^{\infty}\sum_{j=1}^{M_l} |v_j^l|^2 2^{2l\theta} \le c_2\|v\|_{H_\theta}^2, \quad 0 \le \theta \le 1. \tag{3.7}$$

We define the projector $P_L : L^2(\Omega) \to \mathcal{V}_L$ by truncating (3.6):

$$P_L\, v := \sum_{l=0}^{L}\sum_{j=1}^{M_l} v_j^l \psi_j^l,\ P_{-1} := 0. \tag{3.8}$$

The increment or detail spaces $\mathcal{W}^l$ are defined by

$$\mathcal{W}_l := \operatorname{span}\left\{\psi_j^l : \ 1 \le j \le M_l\right\}, \ l = 1, 2, \ldots, \quad \mathcal{W}_0 := \mathcal{V}_0. \tag{3.9}$$

Then it holds that

$$\mathcal{V}_l = \mathcal{V}_{l-1} \oplus \mathcal{W}_l \text{ for } l \ge 1 \quad \text{and} \quad \mathcal{V}_l = \mathcal{W}_0 \oplus \cdots \oplus \mathcal{W}_l, \ l \ge 0, \tag{3.10}$$

and $Q_l := P_l - P_{l-1}$ is a projection from $L^2(\Omega)$ to $\mathcal{W}_l$ (see [2, 3]).

1 (spline prewavelet basis). A wavelet basis is a prewavelet basis if $\mathcal{W}_l$ is orthogonal in $L^2(0,1)$ to $\mathcal{V}_{l-1}$, i.e.,

$$\forall\, l \ge 1, \ \forall\, j \in \{1, \ldots, M_l\}, \quad (\psi_j^l, \phi)_{L^2(0,1)} = 0 \quad \forall\, \phi \in \mathcal{V}_{l-1}.$$

In this case, P_L in (3.8) is the L^2-projection. Prewavelets are functions with small support in $\mathcal{V}_l$ which are L^2-orthogonal on $\mathcal{V}_{l-1}$. The interior wavelet functions are obtained as a linear combination of fine scaling functions with coefficients $(1/10, -6/10, 1, -6/10, 1/10)$.

For the boundary wavelets the coefficients are $(9/10, -6/10, 1/10)$ for the left boundary and $(1/10, -6/10, 9/10)$ for the right boundary.

2 (spline biorthogonal wavelet basis). Another family of wavelets are so-called biorthogonal wavelets, which need not be orthogonal on $\mathcal{V}_{l-1}$.

Let $\Omega = (0,1), \mathcal{I}^l := \{0, 1, \ldots, 2^l - 1\}$, and let ϕ be the hat function taking values 0, 1, 0 at 0, 1/2, 1, respectively, and vanishing outside $\Omega = (0,1)$. We define ψ as the continuous, piecewise linear function taking values $(0, -1/2, 1, -1/2, 0)$ at $(0, 1/4, 1/2, 3/4, 1)$ and vanishing outside $(0,1)$. Similarly, $\psi^\star$ takes values $(0, 1, -1/2, 0)$ at $(0, 1/4, 1/2, 3/4)$ and $\psi_\star$ assumes values $(0, -1/2, 1, 0)$ at $(1/4, 1/2, 3/4, 1)$. Further, we define $\psi_0^0 := \phi$ (scaling function), $\psi_0^l(x) := \psi^\star(2^{l-1}x)$, $\psi_{2^l-1}^l := \psi_\star(2^{l-1}x - 2^{l-1} + 1)$, $x \in I$, for $l \ge 1$ (boundary wavelets).

The interior wavelets are given by $\psi_j^l(x) := \psi(2^{l-2}x - 2^{-2}j + 2^{-2})$, $x \in \Omega$, for $1 \le j \le 2^l - 2$ and $l \ge 2$. These boundary wavelets are suitable to accommodate zero Dirichlet boundary conditions. In the case of Neumann boundary conditions the wavelet at the left boundary has values $-2, 2, -1, 0, \ldots, 0$ (and similarly at the right boundary); see Figure 3.1.

3.3. Two-grid space discretization. We will discuss two different choices for the system of shape functions $\{\Phi_j\}_{j=1}^N$. The standard hat functions are given by

$$\varphi_i(x) = \max(1 - |x - x_i|/h, 0), \quad i = 1, \ldots, N. \tag{3.11}$$

The second choice is the hierarchical spline wavelets basis from Example 2 (see also Figure 3.1) or the prewavelet basis from Example 1.

The direct discretization of the HUM algorithm leads to an discrete problem leading to the oscillatory results described in [4, 5]. With f_h a convenient approximation of $f = \{u^1, -u^0\}$, they approximate problem $\Lambda e = f$ by

$$\Lambda_h e_h = f_h.$$

From the numerical results discussed in detail in [4], we see that when h decreases to zero, the of the discrete problem is getting worse. From the oscillatory results it is quite clear that the trouble lies with the of the discrete solution or, to be more precise, with the way the discrete operator Λ_h acts

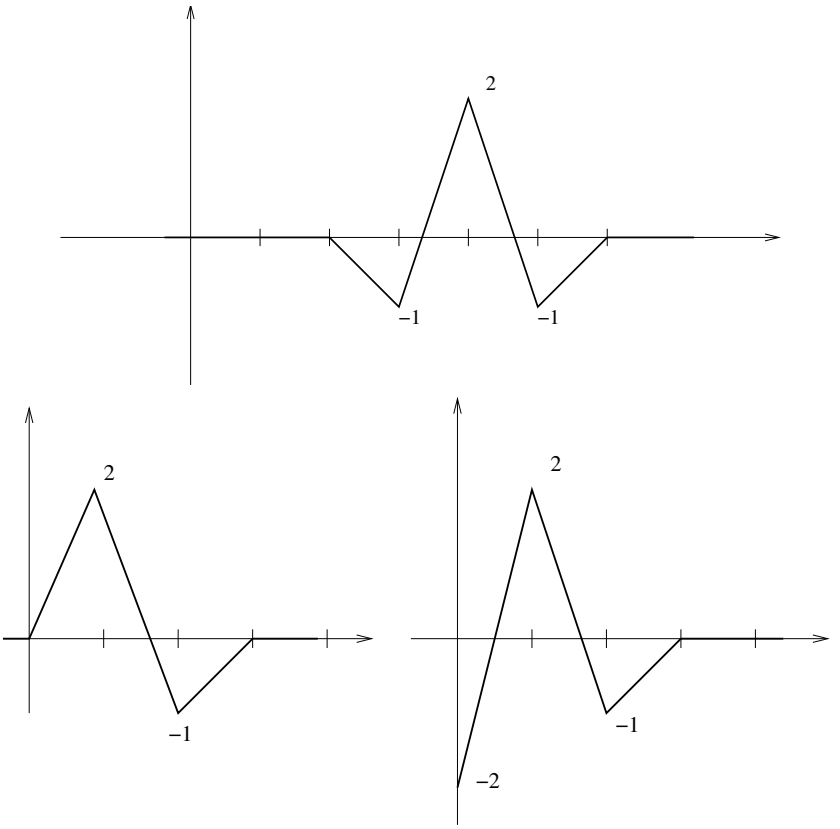


FIG. 3.1. *Generating wavelets: interior wavelets (top) and boundary wavelets for Dirichlet boundary conditions (bottom left) and Neumann boundary conditions (bottom right).*

on the [illegible] component of e_h, i.e., small perturbations of the right-hand side of the discrete problem, $\Lambda_h e_h = f_h$, can produce very large variations of the corresponding solution.

Our implementation of the HUM algorithm is based on the hierarchical spline wavelet basis and a two-grid filtering technique. The idea is to eliminate the high-frequency components of the initial conditions $\underline{\phi}_h^0$ and $\underline{\phi}_h^1$ by defining them on a finer grid with mesh size $h/2$. To this end, we take

$$N = N_L = 2^L - 1, \quad h = h_L = 2^{-L}.$$

The interpolation operator $I_L^{L+1} : \mathbb{R}^{N_L} \to \mathbb{R}^{N_{L+1}}$ maps coarse grid functions onto fine grid functions

$$I_L^{L+1} \underline{v}_w^L := \underline{v}_w^{L+1}, \tag{3.12}$$

where

$$\underline{v}_w^L = \{v_{(j,l)}^L\}_{\substack{l=0,\dots,L \\ j\in\mathcal{I}_l}}, \qquad \underline{v}_w^{L+1} = \{v_{(j',l')}^{L+1}\}_{\substack{l'=0,\dots,L+1 \\ j'\in\mathcal{I}_{l'}}},$$

with

$$v_{(j,l)}^{L+1} = \begin{cases} v_{(j,l)}^L, & l = 0,\dots,L, \quad j \in \mathcal{I}_l, \\ 0, & l = L+1, \qquad j \in \mathcal{I}_{L+1}, \end{cases}$$

and $\mathcal{I}_l := \{0, 1, \ldots, 2^l - 1\}$. The restriction operator $I_{L+1}^L : \mathbb{R}^{N_{L+1}} \to \mathbb{R}^{N_L}$ maps fine grid functions into coarse grid functions as follows:

$$\underline{v}_w^{L+1} = \{v_{(j',l')}^{L+1}\}_{\substack{l'=0,\ldots,L+1 \\ j' \in \mathcal{I}_{l'}}} \mapsto \underline{v}_w^L = \{v_{(j,l)}^{L+1}\}_{\substack{l=0,\ldots,L \\ j \in \mathcal{I}_l}}. \tag{3.13}$$

3.2. In the hat function basis, the analogous interpolation operator $J_L^{L+1} : \mathbb{R}^{N_L} \to \mathbb{R}^{N_{L+1}}$ is defined as

$$\underline{v}_{hat}^L = \{v_j^L\}_{j=1,\ldots,N_L} \mapsto \underline{v}_{hat}^{L+1} = \{v_{j'}^{L+1}\}_{j'=1,\ldots,N_{L+1}},$$

with

$$v_{2j}^{L+1} = v_j^L, \quad v_{2j+1}^{L+1} = \frac{1}{2}\left(v_j^L + v_{j+1}^L\right)$$

(see [1] for more details).

Let $v \in V_L$ be a coarse grid function. We denote by $\underline{v}_w^L$ and $\underline{v}_{hat}^L$ the coefficient vectors of v in the wavelet basis and in the hat function basis, respectively. With $\underline{v}_w^{L+1} = I_L^{L+1}\underline{v}_w^L$ and $\underline{v}_{hat}^{L+1} = J_L^{L+1}\underline{v}_{hat}^L$ it holds that

$$\sum_{l=0}^{L+1}\sum_{k=0}^{2^l-1} v_{k,l}^{L+1}\psi_k^l = \sum_{j=0}^{N_{L+1}} v_j^{L+1}\varphi_j^{L+1}, \tag{3.14}$$

i.e., the fine grid interpolation function is the same if we use the extension operator I_L^{L+1} in the hierarchical wavelet basis, which acts by adding zeros on the finest level, or if we employ the natural extension operator J_L^{L+1} in the two-grid system of hat function bases.

3.4. Wavelet filtering approximation of $\Lambda e = f$. In this section we describe a method based on two-grid algorithm wavelet filtering to approximate (2.3), i.e., $\Lambda e = f$. Let us denote by Λ_L the operator that acts as follows: given $e_L := \{\phi_L^0, \phi_L^1\} \in (V_L)^2$ as the initial condition for the homogeneous wave equation (3.2), we solve (3.2) exactly in time, and by the finite element method in x based on the finite element space V_L, find $\phi_L(t) \in V_L$ such that

$$\begin{aligned} &\frac{d^2}{dt^2}\left(\phi_L(t),\, v\right)_{L^2(\Omega)} + \left(\frac{\partial\phi_L(t)}{\partial x},\, \frac{dv}{dx}\right)_{L^2(\Omega)} = 0 \qquad \forall v \in V_L, \\ &(\phi_L(0), \phi_L'(0)) = e_L. \end{aligned}$$

Likewise, one solves backward in time the controlled system continuous in time and space semidiscrete with finite element space V_L, with terminal data $(0, 0)$ and Dirichlet control at $x = 1$ given by $\partial_x^h\phi_L(1, t)$, where the finite difference operator ∂_x^h is defined by

$$\partial_x^h\phi_L(1, t) = -\phi_{N_L}^L/h_L, \tag{3.15}$$

with $\underline{\phi}^L = \{\phi_i^L\}_{i=1}^{N_L}$ being the coefficient vector of ϕ_L in the hat function basis. Let ψ_L denote the solution of the discrete controlled system. The operator Λ_L is then defined as follows:

$$\Lambda_L e_L := \{\partial_t\psi_L(0), -\psi_L(0)\}. \tag{3.16}$$

Heuristically, $\Lambda_\infty = \Lambda$.

We define next the two-grid filtering operator $\Lambda_{L,\tilde{L}}$. For any $\tilde{L} > L$ denote by $I_L^{\tilde{L}} = I_{\tilde{L}-1}^{\tilde{L}} \circ \cdots \circ I_L^{L+1}$ the extension operator and by $I_{\tilde{L}}^{L} = I_{L+1}^{L} \circ \cdots \circ I_{\tilde{L}}^{\tilde{L}-1}$ the restriction operator acting between V_L and $V_{\tilde{L}}$.

ALGORITHM 3.3.

- Step 1. Given initial state $e_L = (\phi_L^0, \phi_L^1) \in (V_L)^2$, solve the uncontrolled wave equation (3.2) in $V_{\tilde{L}}$ with $\tilde{L} > L$. Denote by $\phi_{\tilde{L}}$ the corresponding solution.
- Step 2. Next solve backward in time the controlled system with terminal data $(0,0)$ and Dirichlet control at $x = 1$ given by $\partial_x^h \phi_{\tilde{L}}(1,t)$ with ∂_x^h defined as in (3.15), and denote by $\psi_{\tilde{L}}$ the corresponding solution.

  Note that we use here the wavelet basis for the finite element space discretization. Unlike in the case when hat shape functions are employed where the stiffness and mass matrices have a three diagonal structure, these matrices in the wavelet basis have rather $\mathcal{O}(N_L \log N_L)$ nonzero entries. They are, however, still sparse matrices, and the matrix-vector multiplication is realized in $\mathcal{O}(N_L \log N_L)$ floating point operations.

  Denote $\psi_{\tilde{L}}^0 = \psi(\cdot,0)$, $\psi_{\tilde{L}}^1 = \frac{\partial \psi}{\partial t}(\cdot,0)$ and let $\underline{\psi}_w^{0,\tilde{L}}$, $\underline{\psi}_w^{1,\tilde{L}}$ denote the coefficient vectors in the hierarchical wavelet basis.
- Step 3 (filtering). Restrict $(\underline{\psi}_w^{0,\tilde{L}}, \underline{\psi}_w^{1,\tilde{L}})$ to the coarse space as follows: let

$$\left(\underline{\psi}_w^{0,L}, \underline{\psi}_w^{1,L}\right) = I_{\tilde{L}}^{L}\left(\underline{\psi}_w^{0,\tilde{L}}, \underline{\psi}_w^{1,\tilde{L}}\right)$$

  and denote by (ψ_L^0, ψ_L^1) the corresponding finite element functions.
- Step 4. Finally, the operator $\Lambda_{L,\tilde{L}}$ acts as follows:

$$\langle \Lambda_{L,\tilde{L}}\{\phi_L^0, \phi_L^1\}, \{\bar{\phi}_L^0, \bar{\phi}_L^1\}\rangle = \left(\underline{\bar{\phi}}_w^{0,L}\right)^{\top} \mathbf{M}_w \underline{\psi}_w^{1,L} - \left(\underline{\bar{\phi}}_w^{1,L}\right)^{\top} \mathbf{M}_w \underline{\psi}_w^{0,L}.$$

  We observe that

$$\Lambda_{L,\tilde{L}} = P_L \Lambda_{\tilde{L}}.$$

- Step 5. Denote by $\mathbf{\Lambda}_{L,\tilde{L}}$ the wavelet basis matrix corresponding to $\Lambda_{L,\tilde{L}}$.

As an approximation of $\partial_x \phi(1,t)$ in $V_{\tilde{L}}$ below we take $\partial_x^h \phi_h(1,t)$ with $h = h_{\tilde{L}}$ as in (3.15):

$$\partial_x^h \phi_{\tilde{L}}(1,t) = -\frac{\phi_{N_{\tilde{L}}}^{\tilde{L}}}{h_{\tilde{L}}}(t).$$

In the hierarchical wavelet basis from Example 2 (see also Figure 3.1) $\underline{\partial}_x \phi_{\tilde{L}}(1,t)$ is given by

$$\partial_x^h \phi_{\tilde{L}}(1,t) = -\sum_{l=0}^{\tilde{L}} 2^{l-\tilde{L}} \frac{\phi_{N_{\tilde{L}}-1,l}^{\tilde{L}}}{h_{\tilde{L}}}(t), \tag{3.17}$$

with $\underline{\phi}_w^{\tilde{L}} = \{\phi_{j,l}^{\tilde{L}}\}$ being the coefficient vector with respect to the wavelet basis $\{\psi_j^l\}$.

3.5. Conjugate gradient solution of $\mathbf{\Lambda_{L,\tilde{L}} e_L = f_L}$, $\mathbf{\tilde{L} > L}$. The conjugate gradient algorithm for finding an approximate solution of the equation $\Lambda_{L,\tilde{L}} e_L = f_L$ with space discretization level $\tilde{L} > L$ for the wave equation is a Galerkin discretization of algorithm (2.19)–(2.35).

First of all, let us consider the systems obtained by a wavelet approximation of the semidiscretization of (1.1) and (2.1) in $V_{\tilde{L}}$.

We obtain the following conjugate gradient algorithm for the application of HUM based on hierarchical wavelet bases and a multigrid filtering technique.

- Step 0.
  (a)

$$\text{Given } e_0^0 \in V_L \text{ and } e_0^1 \in V_L \tag{3.18}$$

solve the homogeneous semidiscrete system

$$\mathbf{M}\phi_{0,\tilde{L}}''(t) + \mathbf{A}\phi_{0,\tilde{L}}(t) = 0, \qquad 0 < t < T, \tag{3.19}$$

in $V_{\tilde{L}}$, initialized by

$$\phi_{0,L}(0) = e_0^0, \quad \partial_t \phi_{0,L}(0) = e_0^1, \tag{3.20}$$

$$\phi_{0,\tilde{L}}(0) = I_L^{\tilde{L}} \phi_{0,L}(0), \quad \partial_t \phi_{0,\tilde{L}}(0) = I_L^{\tilde{L}} \partial_t \phi_{0,L}(0), \tag{3.21}$$

and denote by $\phi_{0,\tilde{L}}$ the corresponding solution.

  (b) Solve the backward wave equation in $V_{\tilde{L}}$,

$$\mathbf{M}\psi_{0,\tilde{L}}''(t) + \mathbf{A}\psi_{0,\tilde{L}}(t) = 0, \qquad 0 < t < T, \tag{3.22}$$

with terminal data $(0,0)$ in $t = T$ and Dirichlet control at $x = 1$ given by $\partial_x^h \phi_{0,\tilde{L}}(1,t)$, with ∂_x^h defined as in (3.17), and denote by $\psi_{0,\tilde{L}}$ the corresponding solution.

  (c) (Filtering.)
  Restrict $\psi_{0,\tilde{L}}$ to the coarse space,

$$\psi_{0,L} = I_{\tilde{L}}^{L} \psi_{0,\tilde{L}},$$

and denote by $\psi_{0,L}$ the restricted functions.

  (d) Compute $g_0 = \{g_0^0, g_0^1\} \in V_L \times V_L$, solving the discrete problem

$$\mathbf{A}(g_0^0) = (\partial_t \psi_{0,L})(0) - u^1 \text{ in } (0,1), \quad g_0^0 \in V_L, \tag{3.23}$$

and set

$$g_0^1 = u^0 - \psi_{0,L}(0), \quad g_0^1 \in V_L, \tag{3.24}$$

If $g_0 = 0$ or is , take $e_L = e_0$; if not, set

$$w_0 = g_0. \tag{3.25}$$

- Step 1.
  Then, for $k \geq 0$, assuming that e_k, g_k, w_k, $\phi_{k,L}$, $\psi_{k,L}$ are known, compute e_{k+1}, g_{k+1}, w_{k+1}, $\phi_{k+1,L}$, $\psi_{k+1,L}$ as follows:
  Solve

$$\mathbf{M}\phi_{k,\tilde{L}}''(t) + \mathbf{A}\phi_{k,\tilde{L}}(t) = 0, \qquad 0 < t < T, \tag{3.26}$$

in $V_{\tilde{L}}$ with sufficiently large $\tilde{L} > L$, initialized by

$$\phi_{k,L}(0) = w_k^0, \quad \partial_t \phi_{k,L}(0) = w_k^1, \tag{3.27}$$

$$\phi_{k,\tilde{L}}(0) = I_L^{\tilde{L}} \phi_{k,L}(0), \quad \partial_t \phi_{k,\tilde{L}}(0) = I_L^{\tilde{L}} \partial_t \phi_{k,L}(0), \tag{3.28}$$

and denote by $\phi_{k,\tilde{L}}$ the corresponding solution; solve in $V_{\tilde{L}}$ the backward time system

$$\mathbf{M}\psi''_{k,\tilde{L}}(t) + \mathbf{A}\psi_{k,\tilde{L}}(t) = 0, \qquad 0 < t < T, \tag{3.29}$$

with final data $(0,0)$ in $t = T$ and with Dirichlet control at $x = 1$ given by $\partial_x^h \phi_{k,\tilde{L}}(1,t)$ with ∂_x^h defined as in (3.17) and with L in place of $\tilde{L}$. This corresponds to truncation of the high-frequency components in the calculation of the numerical control. We denote by $\psi_{k,\tilde{L}}$ the corresponding solution.

Restrict $\psi_{k,\tilde{L}}$ to the coarse space: take

$$\psi_{k,L} = I_{\tilde{L}}^L \psi_{k,\tilde{L}}$$

and denote by $\psi_{k,L}$ the resulting functions.

Compute $g_k = \{g_k^0, g_k^1\} \in V_L \times V_L$, solving the discrete problem

$$A(g_k^0) = (\partial_t \psi_{k,L})(0) \text{ in } (0,1), \quad g_k^0 \in V_L, \tag{3.30}$$

and then

$$g_k^1 = -\psi_{k,L}(0), \quad g_k^1 \in V_L. \tag{3.31}$$

Compute

$$\rho_k = \frac{(|g_k^0|^2_{\hbar^{-1}} + |g_k^1|^2_{\ell^2})}{(g_k^0, w_k^0)_{\hbar^{-1} \times \hbar^1} + (g_k^1, w_k^1)_{\ell^2}}, \tag{3.32}$$

and

$$e_{k+1} = e_k - \rho_k w_k, \tag{3.33}$$

$$\phi_{k+1,L} = \phi_{k,L} - \rho_k \phi_{k,L}, \tag{3.34}$$

$$\psi_{k+1,L} = \psi_{k,L} - \rho_k \psi_{k,L}, \tag{3.35}$$

$$g_{k+1} = g_k - \rho_k g_k. \tag{3.36}$$

- Step 2. [illegible]
If $g_{k+1} = 0$ or is [illegible], take $e_L = e_{k+1}$, $\phi_L = \phi_{k+1,L}$, $\psi_L = \psi_{k+1,L}$; otherwise, compute

$$\gamma_n = \frac{(|g_{k+1}^0|^2_{\hbar^{-1}} + |g_{k+1}^1|^2_{\ell^2})}{(|g_k^0|^2_{\hbar^{-1}} + |g_k^1|^2_{\ell^2})}, \tag{3.37}$$

and set the new descent direction by

$$w_{k+1} = g_{k+1} + \gamma_k w_k. \tag{3.38}$$

Set $k := k + 1$ and go to (3.26).

3.6. Matrix representation of Λ. By (2.6) and (2.7), for $T \geq T_{\min} = 2$ the operator Λ is continuous and coercive in the $\|\cdot\|_E$-norm (2.8). If we expand both ϕ^0 and ϕ^1 in the wavelet series

$$\phi^0 = \sum_{l=0}^{\infty} \sum_{j=0}^{2^l-1} \phi^0_{j,l} \psi^l_j, \quad \phi^1 = \sum_{l=0}^{\infty} \sum_{j=0}^{2^l-1} \phi^1_{j,l} \psi^l_j$$

(prewavelet basis or the wavelets as in Example 2) and exploit the norm equivalencies as in (3.7), then (2.6)–(2.7) become

$$|\langle \Lambda\{\phi^0, \phi^1\}, \{\hat{\phi}^0, \hat{\phi}^1\}\rangle| \leq C \left[\sum_{l=0}^{\infty} \sum_{k=0}^{2^l-1} \left(\phi^0_{k,l}\right)^2 2^l + \sum_{l=0}^{\infty} \sum_{k=0}^{2^l-1} \left(\phi^1_{k,l}\right)^2 2^{-l} \right]^{1/2} \times \left[\sum_{l=0}^{\infty} \sum_{k=0}^{2^l-1} \left(\hat{\phi}^0_{k,l}\right)^2 2^l + \sum_{l=0}^{\infty} \sum_{k=0}^{2^l-1} \left(\hat{\phi}^1_{k,l}\right)^2 2^{-l} \right]^{1/2} \tag{3.39}$$

and

$$\langle \Lambda\{\phi^0, \phi^1\}, \{\phi^0, \phi^1\}\rangle \geq \gamma \left[\sum_{l=0}^{\infty} \sum_{k=0}^{2^l-1} \left(\phi^0_{k,l}\right)^2 2^l + \sum_{l=0}^{\infty} \sum_{k=0}^{2^l-1} \left(\phi^1_{k,l}\right)^2 2^{-l} \right]. \tag{3.40}$$

Denote by $\mathbf{\Lambda}_{L,\tilde{L}}$ the matrix corresponding to the wavelet representation of $\Lambda_{L,\tilde{L}}$. This matrix is an $\mathbb{R}^{2N_L \times 2N_L}$ block matrix of the form

$$\mathbf{\Lambda}_{L,\tilde{L}} = \begin{pmatrix} \mathbf{\Lambda}_1 & \mathbf{\Lambda}_2 \\ \mathbf{\Lambda}_3 & \mathbf{\Lambda}_4 \end{pmatrix},$$

with

$$(\mathbf{\Lambda}_1)_{(j,l),(\hat{\jmath},\hat{l})} = \langle \mathbf{\Lambda}_{L,\tilde{L}}\{\psi^l_j, 0\}, \{\psi^{\hat{l}}_{\hat{\jmath}}, 0\}\rangle, \quad (\mathbf{\Lambda_2})_{(j,l),(\hat{\jmath},\hat{l})} = \langle \mathbf{\Lambda}_{L,\tilde{L}}\{\psi^l_j, 0\}, \{0, \psi^{\hat{l}}_{\hat{\jmath}}\}\rangle,$$

$$(\mathbf{\Lambda}_3)_{(j,l),(\hat{\jmath},\hat{l})} = \langle \mathbf{\Lambda}_{L,\tilde{L}}\{0, \psi^l_j\}, \{\psi^{\hat{l}}_{\hat{\jmath}}, 0\}\rangle, \quad (\mathbf{\Lambda}_4)_{(j,l),(\hat{\jmath},\hat{l})} = \langle \mathbf{\Lambda}_{L,\tilde{L}}\{0, \psi^l_j\}, \{0, \psi^{\hat{l}}_{\hat{\jmath}}\}\rangle.$$

To simplify the notation we omit the level L (or $\tilde{L}$) and the index $_w$, unless necessary. By $\sum_{j,l}$ we will always understand $\sum_{l=0}^{L} \sum_{j \in \mathcal{I}_l}$, with $\mathcal{I}_l = \{0, 1, \ldots, 2^l - 1\}$. Then we have

$$\begin{aligned} \Lambda_{L,\tilde{L}}\{\phi^0, \phi^1\} &= \Lambda_{L,\tilde{L}} \left\{ \sum_{j,l} \phi^0_{j,l} \psi^l_j, \sum_{p,q} \phi^1_{p,q} \psi^q_p \right\} \\ &= \Lambda_{L,\tilde{L}} \left\{ \sum_{j,l} \phi^0_{j,l} \psi^l_j, 0 \right\} + \Lambda_{L,\tilde{L}} \left\{ 0, \sum_{p,q} \phi^1_{p,q} \psi^q_p \right\} \\ &= \sum_{j,l} \phi^0_{j,l} \Lambda_{L,\tilde{L}}\{\psi^l_j, 0\} + \sum_{p,q} \phi^1_{p,q} \Lambda_{L,\tilde{L}}\{0, \psi^q_p\}. \end{aligned}$$

Therefore,

(3.41)

$$= \begin{pmatrix} \underline{\hat{\phi}}^0 \\ \\ \underline{\hat{\phi}}^1 \end{pmatrix}^\top \mathbf{\Lambda}_{L,\tilde{L}} \begin{pmatrix} \underline{\phi}^0 \\ \\ \underline{\phi}^1 \end{pmatrix} = \langle \Lambda_{L,\tilde{L}}\{\phi^0,\phi^1\},\{\hat{\phi}^0,\hat{\phi}^1\}\rangle$$

$$= \left\langle \Lambda_{L,\tilde{L}} \left\{ \sum_{j,l} \phi^0_{j,l}\psi^l_j, \sum_{p,q} \phi_{p,q}\psi^q_p \right\}, \left\{ \sum_{\hat{j},\hat{l}} \hat{\phi}^0_{\hat{j},\hat{l}}\psi^{\hat{l}}_{\hat{j}}, \sum_{\hat{p},\hat{q}} \hat{\phi}^1_{\hat{p},\hat{q}}\psi^{\hat{q}}_{\hat{p}} \right\} \right\rangle$$

$$= \left\langle \sum_{j,l} \phi^0_{j,l}\Lambda_{L,\tilde{L}}\{\psi^l_j,0\} + \sum_{p,q} \phi^1_{p,q}\Lambda_{L,\tilde{L}}\{0,\psi^q_p\}, \sum_{\hat{j},\hat{l}} \hat{\phi}^0_{\hat{j},\hat{l}}\{\psi^{\hat{l}}_{\hat{j}},0\} + \sum_{\hat{p},\hat{q}} \hat{\phi}^1_{\hat{p},\hat{q}}\{0,\psi^{\hat{q}}_{\hat{p}}\} \right\rangle$$

$$= \sum_{j,l}\sum_{\hat{j},\hat{l}} \phi^0_{j,l}\hat{\phi}^0_{\hat{j},\hat{l}}\langle \Lambda_{L,\tilde{L}}\{\psi^l_j,0\},\{\psi^{\hat{l}}_{\hat{j}},0\}\rangle + \sum_{p,q}\sum_{\hat{j},\hat{l}} \phi^1_{p,q}\hat{\phi}^0_{\hat{j},\hat{l}}\langle \Lambda_{L,\tilde{L}}\{0,\psi^q_p\},\{\psi^{\hat{l}}_{\hat{j}},0\}\rangle$$

$$+ \sum_{j,l}\sum_{\hat{p},\hat{q}} \phi^0_{j,l}\hat{\phi}^1_{\hat{p},\hat{q}}\langle \Lambda_{L,\tilde{L}}\{\psi^l_j,0\},\{0,\psi^{\hat{q}}_{\hat{p}}\}\rangle + \sum_{p,q}\sum_{\hat{p}\hat{q}} \phi^1_{p,q}\hat{\phi}^1_{\hat{p},\hat{q}}\langle \Lambda_{L,\tilde{L}}\{0,\psi^q_p\},\{0,\psi^{\hat{q}}_{\hat{p}},\}\rangle$$

$$= \begin{pmatrix} \underline{\hat{\phi}}^0 \\ \\ \underline{\hat{\phi}}^1 \end{pmatrix}^\top \begin{pmatrix} \mathbf{\Lambda}_1 & \mathbf{\Lambda}_2 \\ \\ \mathbf{\Lambda}_3 & \mathbf{\Lambda}_4 \end{pmatrix} \begin{pmatrix} \underline{\phi}^0 \\ \\ \underline{\phi}^1 \end{pmatrix}.$$

We note that the matrix corresponding to $\Lambda_{L,\infty}$ is a finite section of the matrix $\mathbf{\Lambda}$.

3.7. Convergence of the discrete controls. In this section we prove the convergence of the two-grid filtering procedure for finite element controls.

THEOREM 3.4. [illegible] $\{\psi^l_j\}$ [illegible] $e_L := \{\phi^0_L,\phi^1_L\}$ $\bar{e}_L := \{\bar{\phi}^0_L,\bar{\phi}^1_L\} \in E$ [illegible] $L > 0$

$$|\langle (\Lambda_{L,\tilde{L}} - \Lambda)e_L,\bar{e}_L\rangle| \le o(1)\|e_L\|_E\|\bar{e}_L\|_E, \tag{3.42}$$

[illegible] $o(1)$ [illegible] $\tilde{L} \to \infty$ [illegible]. For every $\tilde{L} > L > 0$ we have

$$\begin{aligned} \langle (\Lambda_{L,\tilde{L}} - \Lambda)e_L,\bar{e}_L\rangle &= \langle (P_L\Lambda_{\tilde{L}} - \Lambda_{\tilde{L}} + \Lambda_{\tilde{L}} - \Lambda_\infty)e_L,\bar{e}_L\rangle \\ &= \langle (P_L\Lambda_{\tilde{L}} - \Lambda_{\tilde{L}})e_L,\bar{e}_L\rangle + \langle (\Lambda_{\tilde{L}} - \Lambda_\infty)e_L,\bar{e}_L\rangle. \end{aligned} \tag{3.43}$$

We discuss the two terms separately.

For the first term, we use the first representation in (2.5) of the inner product $\langle\cdot,\cdot\rangle$. We note that for every $e_L \in (V_L)^2$, $\Lambda_{\tilde{L}}e_L \in (V_{\tilde{L}})^2$. More precisely, the solution $\psi_{\tilde{L}}(x,t)$ of the backward controlled, space semidiscrete wave equation can be written as

$$\psi_{\tilde{L}}(x,t) = \sum_{l=0}^{\tilde{L}} \sum_{j\in\mathcal{I}_l} c^l_j(t)\psi^l_j(x),$$

where the $c_j^l(t)$ are obtained by solving the ODEs for the space semidiscrete wave equation in the definition of the operator $\Lambda_{\tilde{L}}$. By (3.16), we then find

$$\Lambda_{\tilde{L}} e_L = \left\{ \sum_{l=0}^{\tilde{L}} \sum_{j\in\mathcal{I}_l} (c_j^l)'(0)\psi_j^l(x),\ -\sum_{l=0}^{\tilde{L}} \sum_{j\in\mathcal{I}_l} c_j^l(0)\psi_j^l(x) \right\},$$

and by the $L^2(0,1)$-orthogonality of the prewavelets between scales,

$$\langle \Lambda_{\tilde{L}} e_L - P_L \Lambda_{\tilde{L}} e_L, \bar{e}_L \rangle = 0, \tag{3.44}$$

and by the L^2-orthogonality of the prewavelets,

$$\langle (P_L \Lambda_{\tilde{L}} - \Lambda_{\tilde{L}}) e_L, \bar{e}_L \rangle \le o(1) \|e_L\|_E \|\bar{e}_L\|_E. \tag{3.45}$$

$o(1)$ tends to 0 when $\tilde{L} \to \infty$ and it is independent of e_L and $\bar{e}_L$.

Let $T > 0$. Denote by $\phi^L(x,t)$, $\bar{\phi}^L(x,t)$ the solutions of the continuous wave equation (1.4) with initial data $e_L := \{\phi_L^0, \phi_L^1\}$, $\bar{e}_L := \{\bar{\phi}_L^0, \bar{\phi}_L^1\} \in E$, respectively. By definition we have $\Lambda_\infty = \Lambda$ and

$$\langle \Lambda e_L, \bar{e}_L \rangle = \int_0^T \frac{\partial \phi^L}{\partial x}(1,t) \frac{\partial \bar{\phi}^L}{\partial x}(1,t) dt.$$

Let $\phi^{L,\tilde{L}}(h,t)$, $\bar{\phi}^{L,\tilde{L}}(h,t) \in V_{\tilde{L}}$ be the solutions of the semidiscrete homogeneous wave equation by finite element method in the fine space $V_{\tilde{L}}$ with initial data e_L, $\bar{e}_L \in (V_L)^2 \subset (V_{\tilde{L}})^2$, $N_{\tilde{L}} = 2^{\tilde{L}} - 1$, and $h = 2^{-\tilde{L}}$. The corresponding coefficient vectors in the hat basis are $\underline{\phi}^{L,\tilde{L}}(h,t) = (\phi_j^{L,\tilde{L}}(h,t))_{j=1,\dots,N_{\tilde{L}}}$, $\underline{\bar{\phi}}^{L,\tilde{L}}(h,t) = (\bar{\phi}_j^{L,\tilde{L}}(h,t))_{j=1,\dots,N_{\tilde{L}}}$, respectively. Following [4, section 6] the operator $\Lambda_{\tilde{L}}$ satisfies

$$\langle \Lambda_{\tilde{L}} e_L, \bar{e}_L \rangle = \int_0^T \frac{\phi_{N_{\tilde{L}}}^{L,\tilde{L}}(h,t)}{h} \frac{\bar{\phi}_{N_{\tilde{L}}}^{L,\tilde{L}}(h,t)}{h} dt.$$

Therefore we get

$$\begin{aligned}
\langle (\Lambda_{\tilde{L}} - \Lambda_\infty) e_L, \bar{e}_L \rangle &= \int_0^T \frac{\phi_{N_{\tilde{L}}}^{L,\tilde{L}}(h,t)}{h} \frac{\bar{\phi}_{N_{\tilde{L}}}^{L,\tilde{L}}(h,t)}{h} dt - \int_0^T \frac{\partial \phi^L(1,t)}{\partial x} \frac{\partial \bar{\phi}^L(1,t) dt}{\partial x} dt \\
&= \int_0^T \frac{\phi_{N_{\tilde{L}}}^{L,\tilde{L}}(h,t)}{h} \left(\frac{\bar{\phi}_{N_{\tilde{L}}}^{L,\tilde{L}}(h,t)}{h} + \frac{\partial \bar{\phi}^L(1,t) dt}{\partial x} \right) dt \\
&\quad - \int_0^T \frac{\partial \bar{\phi}^L(1,t) dt}{\partial x} \left(\frac{\phi_{N_{\tilde{L}}}^{L,\tilde{L}}(h,t)}{h} + \frac{\partial \phi^L(1,t)}{\partial x} \right) dt.
\end{aligned}$$

On the other hand, for every $T > 0$ we have

$$\int_0^T \left| \frac{\phi_{N_{\tilde{L}}}^{L,\tilde{L}}(h,t)}{h} \right|^2 dt \le C(T) \|e_L\|_E, \qquad \int_0^T \left| \frac{\partial \bar{\phi}^L(1,t) dt}{\partial x} \right|^2 dt \le C(T) \|\bar{e}_L\|_E$$

(see inequality (2.72) in [6] and Corollaire 4.1. in [7]).

Clearly, to obtain (3.42) it suffices to prove that for every e_L and $\bar{e}_L$ with $\|e_L\|_E = \|\bar{e}_L\|_E = 1$, as $\tilde{L} \to \infty$,

(3.46)
$$\int_0^T \left| \frac{\phi_{N_{\tilde{L}}}^{L,\tilde{L}}(h,t)}{h} + \frac{\partial \phi^L(1,t)}{\partial x} \right|^2 dt \le o(1), \qquad \int_0^T \left| \frac{\bar{\phi}_{N_{\tilde{L}}}^{L,\tilde{L}}(h,t)}{h} + \frac{\partial \bar{\phi}^L(1,t)}{\partial x} \right|^2 dt \le o(1).$$

Since $\dim V_L < \infty$, it suffices to prove the strong convergence in $L^2(0,T)$ of the discrete controls

$$-\frac{\phi_{N_{\tilde{L}}}^{L,\tilde{L}}(h,t)}{h} \underset{\tilde{L}\to\infty}{\to} \frac{\partial \phi^L(1,t)}{\partial x}, \qquad -\frac{\bar{\phi}_{N_{\tilde{L}}}^{L,\tilde{L}}(h,t)}{h} \underset{\tilde{L}\to\infty}{\to} \frac{\partial \bar{\phi}^L(1,t)}{\partial x} \tag{3.47}$$

for every fixed e_L and $\bar{e}_L$, respectively. This fact is a consequence of the following theorem.

THEOREM 3.5. [illegible] L

$$-\frac{\phi_{N_{\tilde{L}}}^{L,\tilde{L}}(h,t)}{h} \to \frac{\partial \phi^L(1,t)}{\partial x} \quad \text{[illegible]} \quad L^2(0,T) \tag{3.48}$$

[illegible] $\tilde{L} \to \infty$

The proof of Theorem 3.5 is given in the appendix. As an immediate consequence of (3.48) we deduce that

$$\forall L, \quad \forall \tilde{e}_L, e_L \in V_L, \quad |\langle (\Lambda_{\tilde{L}} - \Lambda)\, e_L, \bar{e}_L \rangle| \le o(1) \|e_L\|_E \|\bar{e}_L\|_E \quad \text{as } \tilde{L} \to \infty. \tag{3.49}$$

Therefore, combining (3.44) and (3.49) we deduce that (3.42) holds with $o(1)$ tending to 0 when $\tilde{L} \to \infty$, it is independent of e_L and $\bar{e}_L$, and the proof of Theorem 3.4 is now complete. ∎

3.8. Multilevel preconditioning of $\mathbf{\Lambda_{L,\tilde{L}}}$. Since the HUM algorithm is essentially a conjugate gradient iteration on the operator $\Lambda_{L,\tilde{L}}$, with each iteration amounting to the solution of two wave equations, the number of iteration steps should be kept small. We will precondition the discretized operator Λ by a diagonal wavelet preconditioner of the form $\mathbf{D}_L^{-1/2} \mathbf{\Lambda}_{L,\tilde{L}} \mathbf{D}_L^{-1/2}$. Based on the continuity and coercivity of Λ, (2.6) and (2.7), and using the norm equivalence (3.7), the preconditioned matrix $\mathbf{\Lambda}_{L,\tilde{L}}$ is given by

$$\mathbf{D}_L = \begin{pmatrix} \mathbf{D}_L^1 & 0 \\ 0 & \mathbf{D}_L^2 \end{pmatrix}, \tag{3.50}$$

where $\mathbf{D}_L^1 = \mathrm{diag}(1 : 2^l,\, l = 0, \ldots, L)$ and $\mathbf{D}_L^2 = \mathrm{diag}(1 : 2^{-l},\, l = 0, \ldots, L)$. Based on the norm equivalences, we obtain the following.

THEOREM 3.6. [illegible] $\mathbf{D}_L^{-1/2} \mathbf{\Lambda}_{L,\tilde{L}} \mathbf{D}_L^{-1/2}$ [illegible] L [illegible]. The proof is a direct consequence of (2.6)–(2.7), of the norm equivalence (3.7) as expressed in (3.39)–(3.40), and of the approximation result of Theorem 3.4. Indeed, given $C, \gamma > 0$, and given L, by (3.42) there exists $\tilde{L} > L$ (see the uniform observability inequality in [8] and [11] for every $T \ge 2$) such that

$$|\langle (\Lambda_{L,\tilde{L}} - \Lambda) e_L, \bar{e}_L \rangle| \le C \|e_L\|_E \|\bar{e}_L\|_E \quad \forall\, e_L, \bar{e}_L \in (V_L)^2,$$
$$\langle (\Lambda_{L,\tilde{L}} - \Lambda) e_L, e_L \rangle \ge \frac{\gamma}{2} \|e_L\|_E^2 \quad \forall\, e_L \in (V_L)^2.$$

With the constants $C > 0$, $\gamma > 0$ being now as in (3.39)–(3.40) (in particular, C, γ are independent of L) we infer

$$|\langle \Lambda_{L,\tilde{L}} e_L, \bar{e}_L \rangle| \le 2C \|e_L\|_E \|\bar{e}_L\|_E \quad \forall\, e_L, \bar{e}_L \in (V_L)^2, \tag{3.51}$$

$$\langle \Lambda_{L,\tilde{L}} e_L, e_L \rangle = \langle \Lambda e_L, e_L \rangle + \langle (\Lambda_{L,\tilde{L}} - \Lambda) e_L, e_L \rangle \ge \frac{\gamma}{2} \|e_L\|_E^2 \quad \forall\, e_L \in (V_L)^2. \tag{3.52}$$

By the norm equivalence (3.7), there exist constants C_1, $C_2 > 0$ independent of L such that

$$C_1 \|\mathbf{D}_L^{1/2} \begin{pmatrix} \underline{\phi}^0 \\ \underline{\phi}^1 \end{pmatrix} \|_{\ell^2}^2 \le \|e_L\|_E^2 \le C_2 \|\mathbf{D}_L^{1/2} \begin{pmatrix} \underline{\phi}^0 \\ \underline{\phi}^1 \end{pmatrix} \|_{\ell^2}^2 \quad \forall\, e_L = (\phi_L^0, \phi_L^1) \in (V_L)^2, \tag{3.53}$$

where we denoted by $\underline{\phi}^0$, $\underline{\phi}^1$ the wavelet coefficients of ϕ_L^0, ϕ_L^1 in the prewavelet basis. By (3.51)–(3.52) and from inequality (3.53) we conclude that the condition number of $\mathbf{D}_L^{-1/2} \mathbf{\Lambda}_{L,\tilde{L}} \mathbf{D}_L^{-1/2}$ is bounded independent of L. □

4. Numerical results. From the theoretical point of view, it is convenient to choose $\tilde{L}$ possibly much larger than L since $\tilde{L} = \infty$ corresponds to an exact solution of the wave equations. In this case the operator Λ is linear, continuous, and coercive and bounded uniformly with respect to L. If the time-stepping scheme used for the numerical solution of the wave equations is convergent (i.e., in particular, consistent and stable as $\tilde{L} \to \infty$ and as $\Delta t \to 0$), a Strang-type perturbation argument will yield smallness of the resulting perturbation of Λ_L so that $\Lambda_{L,\tilde{L}}$ will also be uniformly bounded with respect to L [illegible] $\tilde{L}$ [illegible] L. This will imply in particular the convergence of the HUM controls as $\tilde{L} \to \infty$ under the condition that $\tilde{L}/L > 1$ be fixed sufficiently large, independently of L.

In this section, we present numerical results of the wavelet-based HUM applied to (1.1). For this example, we found $\tilde{L} = L + 1$ to be sufficient for stability and convergence (with the controllability time $T \ge 2$).

4.1. Wavelet preconditioning and compression of Λ. First we study numerically wavelet preconditioning, the sparsity pattern of the matrix $\mathbf{\Lambda}_{L,\tilde{L}}$, and the sparsity pattern of the matrix $\mathbf{\Lambda}_{L,\tilde{L}}$.

In Figures 4.1 and 4.2 we present the sparsity pattern of the matrix $\mathbf{\Lambda}_{L,L+1}$ for the case when $L \in \{6, 7\}$ and $\alpha = \Delta t/\Delta x \in \{1, 1/5\}$, respectively. Notice that the "off-diagonal" blocks in the first case ($\alpha = 1$) are zeros, as expected from theoretical considerations.

[illegible] 4.1. In the continuous case, the solution of (1.4) admits a Fourier development on the basis of eigenvectors $\{\sin(k\pi x)\}_k$. More precisely, every solution $\phi = \phi(x,t)$ can be written as

$$\phi(x,t) = \sum_{k=0}^{\infty} \left(a_k \cos(k\pi t) + \frac{b_k}{k\pi} \sin(k\pi t) \right) \sin(k\pi x) \tag{4.1}$$

for suitable coefficients a_k, $b_k \in \mathbb{R}$, $k > 0$, that can be computed explicitly in terms of the initial data:

$$\phi^0(x) = \phi(x,0) = \sum_{k=0}^{\infty} a_k \sin(k\pi x), \quad \phi^1(x) = \phi_t(x,0) = \sum_{k=0}^{\infty} b_k \sin(k\pi x).$$

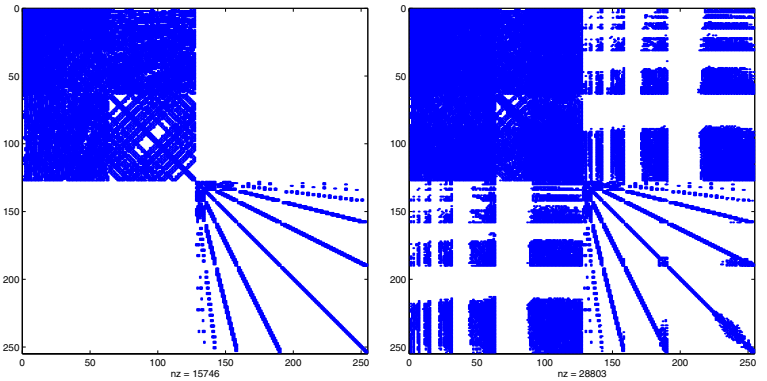


FIG. 4.1. *Sparsity pattern of the matrix* $\mathbf{\Lambda}_{L,L+1}$ *with* $L = 6$, $\alpha = \Delta t/\Delta x = 1$ (*left*) *and* $\alpha = \Delta t/\Delta x = 1/5$ (*right*)*; truncation threshold* $\epsilon = 10^{-10}$.

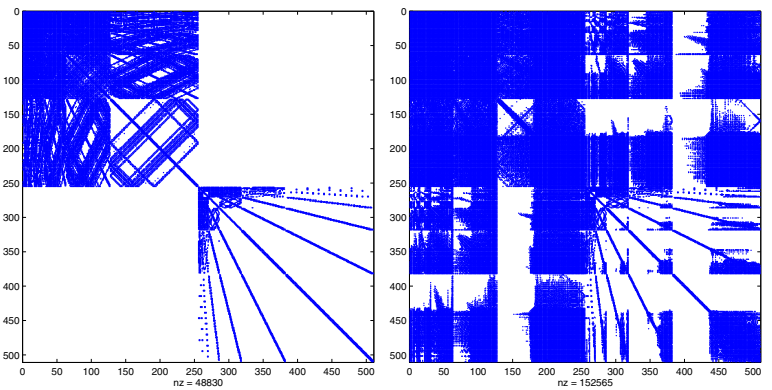


FIG. 4.2. *Sparsity pattern of the matrix* $\mathbf{\Lambda}_{L,L+1}$ *with* $L = 7$, $\alpha = \Delta t/\Delta x = 1$ (*left*) *and* $\alpha = \Delta t/\Delta x = 1/5$ (*right*)*; truncation threshold* $\epsilon = 10^{-10}$.

To define $\mathbf{\Lambda}_2$, we use the solutions

$$\phi(x,t) = \sum_{k=0}^{\infty} a_k \cos(k\pi t)\sin(k\pi x), \quad \hat{\phi}(x,t) = \sum_{k=0}^{\infty} \frac{\hat{b}_k}{k\pi}\sin(k\pi t)\sin(k\pi x)$$

corresponding to the initial data $(\phi^0, 0)$ and $(0, \hat{\phi}^1)$, respectively.

In this case we have

$$\begin{aligned}\int_0^T \frac{\partial\phi(1,t)}{\partial x}\frac{\partial\hat{\phi}(1,t)}{\partial x}dt &= \int_0^T \left(\sum_k k\pi(-1)^k a_k\cos(k\pi t)\right)\left(\sum_l (-1)^l \hat{b}_l \sin(l\pi t)\right)dt \\ &= \sum_{k,l} k\pi(-1)^{k+l} a_k \hat{b}_l \int_0^T \cos(k\pi t)\sin(l\pi t)dt.\end{aligned}$$

For $T = T_{\min} = 2$ it holds that $\int_0^T \cos(k\pi t)\sin(l\pi t)dt = 0$ for all k, l. Consequently, $\mathbf{\Lambda}_2 = 0$.

On the other hand, according to (4.1), the solutions of the homogeneous wave

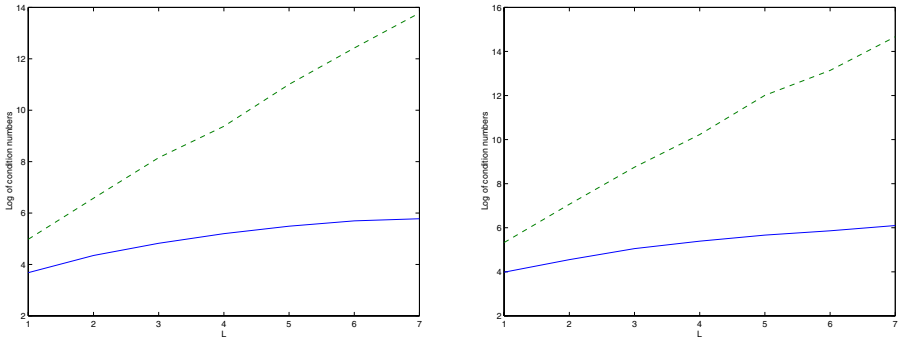


FIG. 4.3. *Natural logarithm of condition numbers of* $\mathbf{\Lambda}_{L,L+1}$ *versus* L *for* $\alpha = 1$ (*left*) *and* $\alpha = 1/5$ (*right*) *with* (*solid line*) *and without* (*dashed line*) *wavelet preconditioner.*

equation with initial data $(0, \phi^1)$ and $(\hat{\phi}^0, 0)$ are

$$\phi(x,t) = \sum_{k=0}^{\infty} \frac{b_k}{k\pi} \sin(k\pi t) \sin(k\pi x), \quad \hat{\phi}(x,t) = \sum_{k=0}^{\infty} \hat{a}_k \cos(k\pi t) \sin(k\pi x),$$

respectively. We deduce that

$$\int_0^T \frac{\partial \phi(1,t)}{\partial x} \frac{\partial \hat{\phi}(1,t)}{\partial x} dt = \sum_{k,l} l\pi(-1)^{k+l} b_k \hat{a}_l \int_0^T \sin(k\pi t) \cos(l\pi t) dt = 0$$

for all $k, l > 0$ and $T = T_{\min} = 2$. Thus, the other off-diagonal block $\mathbf{\Lambda}_4$ satisfies $\mathbf{\Lambda}_4 = 0$.

In Figure 4.3 we illustrate the effect of the wavelet preconditioners on the condition numbers of the matrix $\mathbf{\Lambda}_{L,L+1}$ for the values $\alpha = 1$ and $\alpha = 1/5$ for $L = 1, \ldots, 7$: the diagonal wavelet preconditioner reduces the condition of $\mathbf{\Lambda}_{L,L+1}$ by several orders of magnitude. We have used in our computations the spline-wavelet basis of Example 2; see Figure 3.1.

4.2. Numerical convergence study of the controls. We present here numerical experiments in which we analyze the convergence of the numerical control in conjugate gradient iterations versus L.

The above numerical experiments have been done with $T = 2$. In Table 4.1 ($\alpha = 1$) and in Table 4.2 ($\alpha = 1/5$), $L =$ level and it$(\cdot)$ is the number of conjugate gradient iterations necessary to achieve convergence with

$$(1) \begin{cases} u^0(x) = \sin(5\pi x), \\ u^1(x) = \begin{cases} x, & x \in [0, 1/2), \\ 0, & x \in (1/2, 1], \end{cases} \end{cases}$$

for it(1) and with

$$(2) \begin{cases} u^0(x) = \begin{cases} 0, \\ \text{for } i = 1:10,\ u^0(x) = u^0(x) + \sin(i\pi x), \\ u^0(x) = u^0(x)/10, \end{cases} \\ u^1(x) = 0 \end{cases}$$

for it(2).

TABLE 4.1
$\alpha = 1$, $\varepsilon = 10^{-8}$.

L	it (1)	it (2)	L^2-norm (1)	L^2-norm (2)
1	6	7	3.6486	1.8615
2	7	5	3.7242	1.9231
3	7	6	3.6749	1.8780
4	6	7	3.7202	1.9021
5	6	7	3.6998	1.8812
6	6	6	3.6849	1.8901
7	5	6	3.7001	1.8889

TABLE 4.2
$\alpha = 1/5$, $\varepsilon = 10^{-6}$.

L	it (1)	it (2)	L^2-norm (1)	L^2-norm (2)
1	11	9	3.6591	1.8567
2	10	6	3.6961	1.9078
3	9	6	3.6752	1.7986
4	8	6	3.7989	1.9053
5	8	7	3.6873	1.8932
6	9	5	3.7012	1.8891
7	8	5	3.6978	1.8903

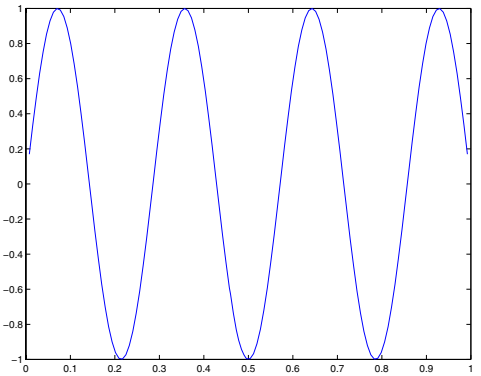
4.3. Numerically exact controllability. To obtain a fully discrete realization of Algorithm 3.3, we have to discretize with respect to time in (3.2). Perhaps the most widely used family of direct methods for solving (3.2) is that of Newmark [14]. By this way, the evolution equations are solved exactly in time. The convergence analysis of the Newmark family of methods is similar to the classical convergence analysis. CFL-type stability conditions must be satisfied for each mode in the discretized system: $\Delta t = \alpha\, h_L$, $\alpha \leq 1$, where the time step is Δt.

We implement this algorithm with $\tilde{L} = L+1$ and test the efficiency of the discrete HUM control v by visualizing and checking how far $u(\cdot, T)$ and $u_t(\cdot, T)$ differ from $(0, 0)$.

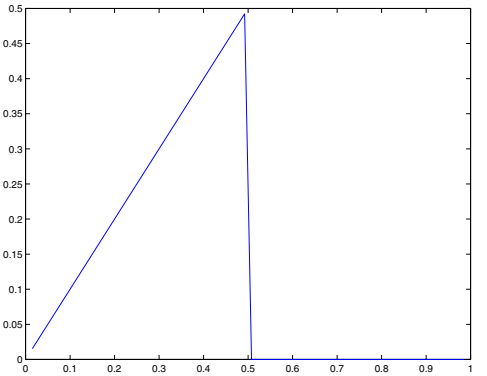
In Figures 4.4 and 4.5 we present two examples corresponding to (1) and (2) with $L = 7$. In each example, (a)–(b) describe the initial data (u^0, u^1) to be controlled along the mesh, (c) and (c′) correspond to the numerical solution u and derivative u' with numerical control which confirm the efficiency of the numerical control, and (d) shows the computed control.

4.4. Further comments and conclusions. We have discussed the numerical implementation of HUM, introduced in [7], applied to the exact controllability of the one-dimensional wave equation when the control is achieved by appropriate Dirichlet boundary conditions. It has been shown that, combining a multilevel finite element discretization and a two-grid filtering technique in the spline wavelet basis, we obtain quite effective numerical methods for approximating the boundary control. The results of Tables 4.1 and 4.2 compare favorably with those obtained using the Tychonoff regularization procedure in [5].

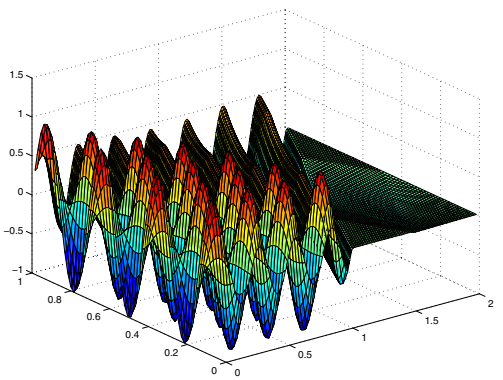
The following facts appear quite clearly: (a) the filtering method described in section 3.4 has been a very effective cure to the ill-posedness of the approximate problem (3.16); (b) as in [4, 5], the number of conjugate gradient iterations necessary to achieve the convergence in the optimal time $T = 2$ is essentially independent of Δx (and L).



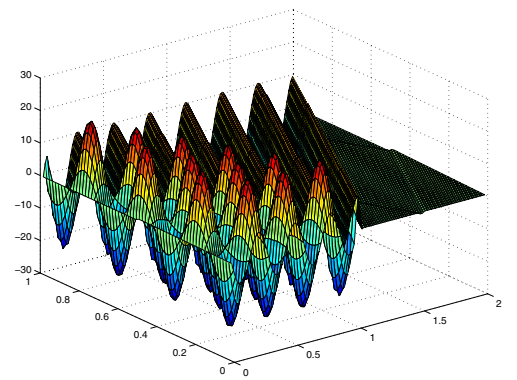
(a) Initial position (u^0).



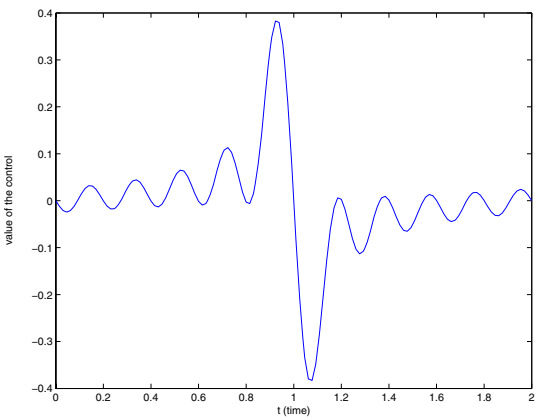
(b) Initial velocity (u^1).



(c) Numerical displacement (u).



(c′) Numerical velocity (u').

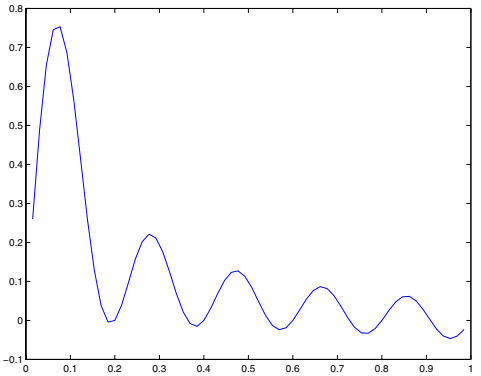


(d) Numerical control.

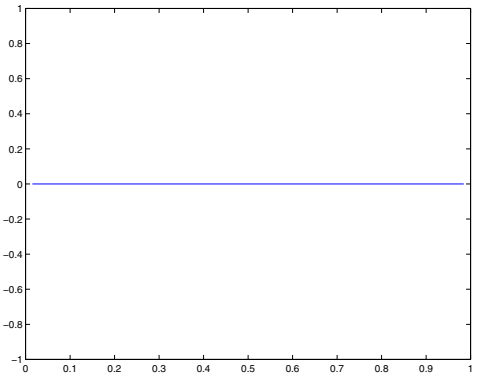
FIG. 4.4. (a)–(b) *Variation of initial data corresponding to* (1). (c)–(c′) *Variation of numerical displacement and numerical velocity with* (d) *numerical control.*

The numerical experiments presented are in very good agreement with the theory [8, 9, 10, 11]. Concerning the CPU time, the scheme appears comparable to the Tychonoff regularization method.

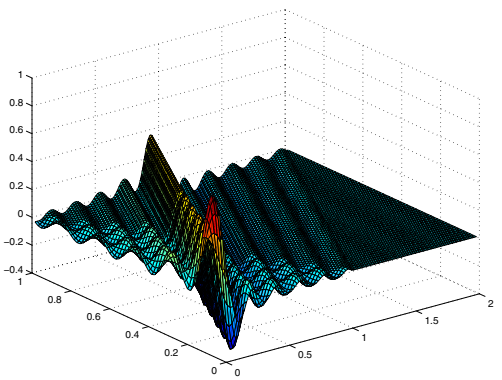
It is the intention of the authors to study whether or not it is possible to extend



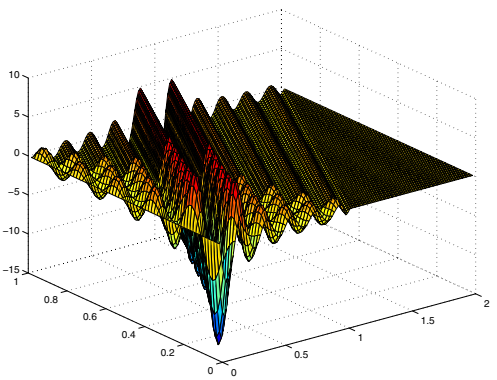
(a) Initial position (u^0).



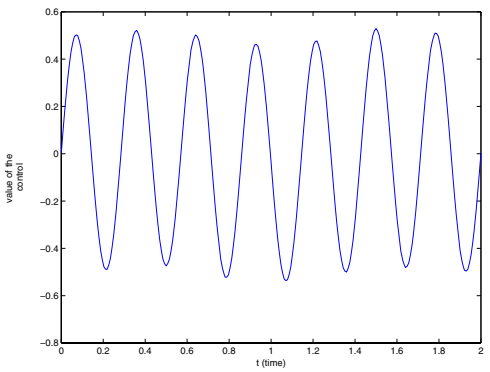
(b) Initial velocity (u^1).



(c) Numerical displacement (u).



(c′) Numerical velocity (u').



(d) Numerical control.

FIG. 4.5. *Variation of initial data* (a)–(b) *corresponding to* (2). (c)–(c′) *Variation of numerical displacement and numerical velocity with* (d) *numerical control.*

or adapt the present analysis to other types of boundary controls, to more complicated and complex time-dependent multidimensional models, such as, for instance, the equations modeling the vibrations of elastic plates. Indeed there are many ways to include multigrid procedures, and investigating them could in itself be an interesting study.

Appendix. Proof of Theorem 3.5. In this part we prove some general results. The convergences obtained in Lemmas A.1 and A.2 were proved in [17] for any bounded family of finite difference space semidiscretization solutions of the two-dimensional uncontrolled wave equation in the square. Here we prove similar results for the finite element method.

Given $\Omega = (0,1)$, $N \in \mathbb{N}$, we set $h = 1/(N+1)$ and denote by $\phi(h,t)$ the solution of the finite element space semidiscretization of the uncontrolled wave equation (3.1).

Let $\varphi_i(x) = \max(1 - |x - x_i|/h, 0)$ denote the hat function basis of V_h, and let $\underline{\phi}(h,t) = \{\phi_i(h,t)\}_{i=1}^N$ be the coefficient vector of $\phi(h,t)$ with respect to $\{\varphi_i\}$. The space semidiscrete wave equation (3.1) can be rewritten as

$$\mathbf{M}\frac{d^2}{dt^2}\underline{\phi}(h,t) + \mathbf{A}\underline{\phi}(h,t) = 0, \qquad 0 < t < T. \tag{A.1}$$

We adopt $\partial_x^h \phi(1,t) = -\frac{\phi_N(h,t)}{h}$ as an approximation of the normal derivative $\partial_x \phi(1,t)$. We consider the family $\{\phi(h,t)\}_h$ of solutions of (3.1) depending on the parameter h and study their limit behavior as $h \to 0$. The solution of (3.1) admits the Fourier expansion

$$\underline{\phi}(h,t) = \sum_{k \geq 1} m_k(h,t)\bar{\varphi}^k(h), \tag{A.2}$$

where $\bar{\varphi}^k(h) = (\varphi_1^k, \dots, \varphi_N^k)$ are the discrete eigenvectors $\varphi_j^k(h) = \sin(jk\pi h)$ and $m_k(h,t)$ are the time-dependent Fourier coefficients: $m_k(h,t) = 0$ for all $t > 0$ and $k \geq N+1$. In particular,

$$m_k(h,t) = m_k(h)e^{i\sqrt{\mu_k(h)}t},$$

with $\mu_k(h)$ being the discrete eigenvalues: $\mu_k(\mathbf{M}^{-1}\mathbf{A}) = \frac{6}{h^2}\frac{1-\cos(k\pi h)}{2+\cos(k\pi h)}$.

The conserved energy of system (3.1) is given by

$$E_h(0) = \frac{h}{6}\sum_{j=1}^{N}\left|\phi_j'(h,t)\right|^2 + \frac{h}{12}\sum_{j=0}^{N}\left|\phi_j'(h,t) + \phi_{j+1}'(h,t)\right|^2 + \frac{h}{2}\sum_{j=0}^{N}\left|\frac{\phi_j(h,t) - \phi_{j+1}(h,t)}{h}\right|^2. \tag{A.3}$$

In the following lemmas we prove that the bounded family of solutions of (3.1) weakly converges as $h \to 0$ to a solution of the continuous homogeneous wave equation (Lemma A.1). Moreover we analyze the behavior of the normal derivative of the solution of the semidiscrete problem (Lemma A.2).

LEMMA A.1. [illegible] $\{\phi(h,t)\}_h$ [illegible] (3.1) [illegible] h $h = 1/(N+1)$ $N \in \mathbb{N}$ [illegible] E_h [illegible]

$$E_h(0) \leq C \quad \forall h > 0. \tag{A.4}$$

[illegible]

$$m_k(h,\cdot) \underset{h\to 0}{\rightharpoonup} m_k(\cdot) \text{ [illegible] } H^1(0,T) \text{ [illegible] } h \to 0 \quad \forall k \in \mathbb{N}. \tag{A.5}$$

[illegible]

$$\phi = \sum_{k \geq 1} m_k(t)\sin(k\pi x), \tag{A.6}$$

[illegible] ϕ [illegible]

$$\phi \in \mathcal{C}([0,T]; H_0^1(0,1)) \cap \mathcal{C}^1([0,T]; L^2(0,1)), \tag{A.7}$$

$$E(0) \le \liminf_{h\to 0} E_h(0); \tag{A.8}$$

$$h\sum_{j=1}^{N} |\phi_j(h,t)|^2 \underset{h\to 0}{\to} \|\phi(t)\|^2_{L^2(0,1)} \text{ [illegible] } L^\infty(0,T). \tag{A.9}$$

[illegible]. Let us introduce the Hilbert spaces ℓ^2, $\hbar^1$, and $\hbar^{-1}$:

$$\ell^2 = \left\{ \{m_k\} \ : \ \|m_k\|^2_{\ell^2} = \sum_{k\in\mathbb{N}} |m_k|^2 < \infty \right\}, \tag{A.10}$$

$$\hbar^1 = \left\{ \{m_k\} \in \ell^2 \ : \ \|m_k\|^2_{\hbar^1} = \sum_{k\in\mathbb{N}} k^2 |m_k|^2 < \infty \right\}, \tag{A.11}$$

$$\hbar^{-1} = \left\{ \{m_k\} \in \ell^2 \ : \ \|m_k\|^2_{\hbar^{-1}} = \sum_{k\in\mathbb{N}} k^{-2} |m_k|^2 < \infty \right\}, \tag{A.12}$$

endowed with their canonical norms.

Due to the fact that the eigenvectors $\bar{\varphi}^k$ are orthogonal in the discrete L^2-norm, for all $k \in \{1, \dots, N\}$, we have

$$\begin{aligned} E_h(t) &= \frac{1}{2}\sum_k \left[|m_k'(h,t)|^2 + \mu_k(h) |m_k(h,t)|^2 \right] \frac{1 + 2\cos^2 \frac{k\pi h}{2}}{6} \\ &\ge \frac{1}{12} \left[\sum_k |m_k'(h,t)|^2 + k^2 |m_k(h,t)|^2 \right]. \end{aligned} \tag{A.13}$$

Then, from the uniform boundedness estimate (A.4) and the conservation of the energy, we deduce that

$$\vec{m}\,(h,\cdot) = \{m_k(h,\cdot)\}_{k\in\mathbb{N}} \text{ is uniformly bounded in } W^{1,\infty}(0,T;\ell^2) \text{ as } h \to 0. \tag{A.14}$$

Thus, extracting subsequences, it follows that

$$\vec{m}\,(h,t) \rightharpoonup \vec{m}\,(t) = \{m_k(t)\}_{k\ge 1} \text{ weakly in } H^1(0,T;\ell^2) \text{ as } h \to 0. \tag{A.15}$$

It is easy to see that $\hbar^1 \hookrightarrow \ell^2$ with compact embedding. According to (A.13) and (A.4) we have

$$\left\{ \vec{m}\,(h,\cdot) \right\} \text{ is uniformly bounded in } \text{[illegible]}\,([0,T];\hbar^1). \tag{A.16}$$

Using the classical Lions–Aubin compactness lemma, we deduce that

$$\left\{ \vec{m}\,(h,\cdot) \right\} \text{ is relatively compact in } \text{[illegible]}\,([0,T];\ell^2), \tag{A.17}$$

$$\vec{m}\,(h,\cdot) \rightharpoonup \vec{m}\,(\cdot) \quad \text{weakly in } H^1(0,T;\ell^2) \cap L^2(0,T;\hbar^1) \text{ as } h \to 0, \tag{A.18}$$

and

$$\vec{m}\,(h,\cdot) \rightharpoonup \vec{m}\,(\cdot) \quad \text{strongly in } \mathcal{C}([0,T];\ell^2). \tag{A.19}$$

We now observe that

$$h\sum_{j=0}^{N+1} |\phi_j(h,t)|^2 = \frac{1}{2}\sum_k |m_k(h,t)|^2 \tag{A.20}$$

($\bar{\varphi}^k$ is orthogonal to $\bar{\varphi}^{k'}$ in the discrete L^2-norm when $k \neq k'$) and therefore

$$h\sum_{j=1}^{N} |\phi_j(h,t)|^2 \underset{h\to 0}{\rightharpoonup} \|\phi(t)\|^2_{L^2(0,1)} \quad \text{in } L^\infty(0,T) \tag{A.21}$$

follows immediately from (A.19), since $\vec{m}\,(\cdot) = \{m_k(\cdot)\}_{k\in\mathbb{N}}$ are precisely the Fourier coefficients of ϕ (A.6) in the basis $\{\sin(k\pi x)\}_k$ of $L^2(0,1)$.

Observe that according to the bounds (A.14) and (A.16) the subsequence may be extracted so that

$$\vec{m}\,(h,0) \rightharpoonup \vec{m}^0 \quad \text{weakly in } \hbar^1, \tag{A.22}$$

$$\vec{m}'\,(h,0) \rightharpoonup \vec{m}^1 \quad \text{weakly in } \ell^2 \tag{A.23}$$

for some $(\vec{m}^0, \vec{m}^1) \in \hbar^1 \times \ell^2$. In view of (A.19) we have

$$\vec{m}\,(0) = \vec{m}^0\,. \tag{A.24}$$

We use the fact that

$$m_k''(h,t) + \mu_k(h) m_k(h,t) = 0,$$

which combined with

$$\mu_k(h) \leq 3k^2\pi^2 \quad \forall h, k > 0$$

implies that

$$\vec{m}''\,(h,\cdot) \text{ is uniformly bounded in } \mathcal{C}([0,\mathrm{T}];\hbar^{-1}). \tag{A.25}$$

Combining (A.14), (A.25), and the compactness results mentioned above we deduce that

$$\vec{m}'\,(h,\cdot) \rightharpoonup \vec{m}'\,(\cdot) \quad \text{strongly in } \mathcal{C}([0,\mathrm{T}],\hbar^{-1}), \tag{A.26}$$

and therefore by (A.23)

$$\vec{m}'\,(0) = \vec{m}^1\,. \tag{A.27}$$

$(\vec{m}^0, \vec{m}^1) \in \hbar^1 \times \ell^2$ is equivalent to saying that the limit initial data are of finite energy. Then ϕ, defined as in (A.6), is the solution of the continuous wave equation with initial data:

$$\phi^0 = \sum_k m_k^0 \sin(k\pi x) \in H_0^1(0,1), \quad \phi^1 = \sum_k m_k^1 \sin(k\pi x) \in L^2(0,1).$$

By uniqueness of solutions we deduce that (A.7) holds. On the other hand,

$$h\sum_{j=1}^{N}|\phi_j(h,t)|^2 \;=\frac{1}{2}\sum_k|m_k(h,t)|^2\to\frac{1}{2}\sum_k|m_k(t)|^2 \;=\|\phi(t)\|^2_{L^2(0,1)}\ \text{in}\,\mathcal{C}(0,T).$$

We conclude the proof by checking the property of lower semicontinuity of the energy (A.8).

In view of (A.22) and (A.23) we deduce that

$$\sum_k|m_k'(0)|^2+\sum_k k^2\pi^2|m_k(0)|^2\le\liminf_{h\to0}\left[\sum_k|m_k'(h,0)|^2+\sum_k\mu_k(h)|m_k(h,0)|^2\right].$$

Obviously

$$E(0)=\frac{1}{2}\sum_k\left[k^2\pi^2|m_k(0)|^2+|m_k'(0)|^2\right].$$

Thus, we deduce (A.8). ∎

In the following lemma we study the convergence of the normal derivatives of the solutions of the semidiscrete system.

LEMMA A.2. $\{\phi(h,t)\}_h$ [illegible] $h\to0$ [illegible] (A.4) [illegible] ϕ [illegible] $h\to0$ [illegible] $\{\phi(h,t)\}_h$ [illegible] A.1 [illegible]

$$\int_0^T|\partial_x\phi(1,t)|^2dt\le\liminf_{h\to0}\int_0^T\left|\frac{\phi_N(h,t)}{h}\right|^2dt, \tag{A.28}$$

[illegible]

$$-\frac{\phi_N(h,t)}{h}\rightharpoonup\partial_x\phi(1,t)\ \text{[illegible]}\ h\to0\ \text{[illegible]}\quad L^2(0,T).$$

[illegible]. Let $\phi(h,t)$ be a solution of (A.1) as in (A.2). In view of the decomposition (A.2) we have

$$\int_0^T\left|\frac{\phi_N(h,t)}{h}\right|^2dt=\left|\sum_k m_k(h,t)\frac{\sin(Nk\pi h)}{h}\right|^2_{L^2(0,T)}. \tag{A.29}$$

For all the solutions of the semidiscrete system, it holds that

$$C(h)\int_0^T\left|\frac{\phi_N(h,t)}{h}\right|^2dt\le E_h(0)\quad\forall h\in(0,1), \tag{A.30}$$

with $C(h)$ uniformly bounded from below with respect to h. Recall that

$$\frac{\sin(Nk\pi h)}{h}=\frac{\sin(k\pi-k\pi h)}{h}=-\cos(k\pi)\frac{\sin(k\pi h)}{h}\to-k\pi\cos(k\pi)$$

as $h\to0$ for all $k\ge0$.

We prove now that

(A.31)
$$-\sum_k m_k(h,t)\frac{\sin(Nk\pi h)}{h} \underset{h\to 0}{\rightharpoonup} \sum_k k\pi\cos(k\pi)m_k(t) \quad \text{weakly in } L^2(0,T) \text{ as } h\to 0.$$

In view of (A.30) we have

$$\sum_k m_k(h,t)\frac{\sin(Nk\pi h)}{h} \quad \text{is bounded in } L^2(0,T). \tag{A.32}$$

For any function $\psi \in \quad (0,T)$ we set

$$v_h = \sum_{k=1}^{N} \langle m_k(h),\psi\rangle_{D'(0,T)\times D(0,T)}\,\varphi^k(h), \tag{A.33}$$

$$v = \sum_{k\in\mathbb{N}} \langle m_k,\psi\rangle_{D'(0,T)\times D(0,T)}\,\varphi^k(x), \quad \varphi^k(x)=\sin(k\pi x). \tag{A.34}$$

For the semidiscrete problem we have

$$-\mathbf{A}\phi_h = \mathbf{M}\phi_h'' = \sum_{k=1}^{N} h\frac{2+\cos(k\pi h)}{3} m_k''(h,t)\bar{\varphi}^k(h), \tag{A.35}$$

and for the continuous problem we have

$$\partial_x^2\phi = \sum_{k\in\mathbb{N}} m_k''(t)\varphi^k = \sum_{k\in\mathbb{N}} m_k(t)(\varphi^k)''. \tag{A.36}$$

As a consequence of Lemma A.1 we have that

$$m_k''(h,t)+\mu_k(h)m_k(h,t) \rightharpoonup m_k''(t)+k^2\pi^2 m_k(t) \tag{A.37}$$

weakly in $H^{-1}(0,T;\ell^2)$ as $h\to 0$. We have

$$\begin{aligned} -(\mathbf{A}v_h)_j &= -\sum_{k=1}^{N} \langle m_k(h),\psi\rangle_{D'(0,T)\times D(0,T)} \left(\mathbf{A}\bar{\varphi}^k(h)\right)_j \\ &= \sum_{k=1}^{N} h\left\langle \frac{2+\cos(k\pi h)}{3} m_k''(h),\psi\right\rangle_{D'(0,T)\times D(0,T)} \varphi_j^k(h) \end{aligned} \tag{A.38}$$

and

$$v'' = \sum_{k\in\mathbb{N}} \langle m_k,\psi\rangle_{D'(0,T)\times D(0,T)}\,(\varphi^k)'' = \sum_{k\in\mathbb{N}} \langle m_k'',\psi\rangle_{D'(0,T)\times D(0,T)}\,\varphi^k. \tag{A.39}$$

Therefore, v_h and v satisfy the elliptic problems

$$\begin{cases} -\left(\mathbf{A}v_h\right)_j = f_j(h), & j=1,\dots,N,\\ v_{h,0}=v_{h,N+1}=0, \end{cases} \tag{A.40}$$

with $f_j(h)=\sum_{k=1}^{N} f_h^k\varphi_j^k(h)$ and $f_h^k = h\left\langle \frac{2+\cos(k\pi h)}{3} m_k''(h,\cdot),\psi(\cdot)\right\rangle_{D'(0,T)\times D(0,T)}$ and

$$\begin{cases} v''=f,\\ v(0)=v(1)=0, \end{cases} \tag{A.41}$$

with $f = \sum_{k\geq 1} f^k \varphi^k = \sum_{k\geq 1} f^k \sin(k\pi x)$ and $f^k = \langle m_k''(\cdot), \psi(\cdot)\rangle_{D'(0,T)\times D(0,T)}$.

Statement (A.31) is equivalent to showing that

$$-\frac{v_{h,N}(h)}{h} \underset{h\to 0}{\to} \frac{\partial v}{\partial x}(1,t). \tag{A.42}$$

We obtained in Lemma A.1

$$\vec{m}''(h,\cdot) \underset{h\to 0}{\rightharpoonup} \vec{m}''(\cdot) \text{ weakly in } H^{-1}(0,T;\ell^2) \text{ as } h\to 0, \tag{A.43}$$

and therefore

$$\left\{\frac{1}{h} f_h^k\right\}_{k=1}^N \underset{h\to 0}{\rightharpoonup} \left\{f^k\right\}_{k\in\mathbb{N}} \text{ weakly in } \ell^2 \text{ as } h\to 0. \tag{A.44}$$

The solutions of the elliptic problems can be written explicitly as

$$v_{h,j} = -\sum_{k=1}^{N} \frac{f_h^k}{h} \sin(k\pi h j) \frac{1}{\lambda_k(h)} \tag{A.45}$$

and

$$v(x) = -\sum_{k\geq 1} \frac{f^k \sin(k\pi x)}{k^2\pi^2}, \tag{A.46}$$

respectively, ($\lambda_k(h) = \frac{4}{h^2}\sin^2\frac{k\pi h}{2}$).

Thus, statement (A.42) is equivalent to

$$\sum_{k=1}^{N} \frac{f_h^k}{\lambda_k(h)} \frac{\sin(Nk\pi h)}{h^2} \to -\sum_{k\geq 1} \frac{f^k}{k\pi}\cos(k\pi) \text{ as } h\to 0. \tag{A.47}$$

In view of (A.44) it is easy to see that for any $M>0$ fixed

$$\sum_{k=1}^{M} \left[\frac{f_h^k}{\lambda_k(h)} \frac{\sin(k\pi h)}{h^2} - \frac{f^k}{k\pi}\right]\cos(k\pi) \to 0 \quad \text{as } h\to 0 \tag{A.48}$$

and

$$f_h^k = 0 \quad \forall\, k \geq N+1.$$

(Note that $\sin(Nk\pi h) = -\sin(k\pi h)\cos(k\pi)$.)

Taking into account that the convergence of each term of the series for k fixed holds, it is enough to prove that for every $\epsilon>0$, there exists $M(\epsilon)>0$ such that

$$\left|\sum_{k\geq M(\epsilon)+1} \left[\frac{f_h^k}{\lambda_k(h)} \frac{\sin(k\pi h)}{h^2} - \frac{f^k}{k\pi}\right]\cos(k\pi)\right| \leq \epsilon \tag{A.49}$$

uniformly as $h\to 0$.

Taking into account that $\left\{f^k\right\}_{k\in\mathbb{N}} \in \ell^2$, it is easy to see that

$$\left|\sum_{k\geq M(\epsilon)+1} f^k \frac{\cos(k\pi)}{k\pi}\right| \leq \frac{\epsilon}{2} \tag{A.50}$$

for $M(\epsilon) > 0$ large enough.

Let us analyze the remainder

(A.51)
$$\left| \sum_{k \geq M(\epsilon)+1} \frac{f_h^k}{\lambda_k(h)} \frac{\sin(k\pi h)}{h^2} \cos(k\pi) \right| \leq \sum_{k \geq M(\epsilon)+1} \frac{|f_h^k|}{h\,\lambda_k(h)} k\pi$$
$$\leq \left(\sum_{k \geq M(\epsilon)+1} \left| \frac{f_h^k}{h} \right|^2 \right)^{\frac{1}{2}} \left(\sum_{k \geq M(\epsilon)+1} \left| \frac{k\pi}{\lambda_k(h)} \right|^2 \right)^{\frac{1}{2}}.$$

In view of (A.44) and (A.51) we observe that it is sufficient to show that

$$\sum_{k \geq M(\epsilon)+1} \left| \frac{k\pi}{\lambda_k(h)} \right|^2 \to 0 \tag{A.52}$$

as $M(\epsilon) \to \infty$ uniformly on h (we take into account that $\lambda_k(h) \geq ck^2$ for some $C > 0$ for all $h > 0$ and $k = 1, \ldots, N$).

This completes the proof of Lemma A.2. □

The following result provides the strong convergence of the normal derivative under suitable assumptions on the initial data.

LEMMA A.3. [illegible] $T > 0$ [illegible] $h \in (0,1)$ [illegible] $(\phi_h^0, \phi_h^1) \in \hbar^1 \times \ell^2$ [illegible] $(\phi^0, \phi^1) \in H_0^1(0,1) \times L^2(0,1)$ [illegible] (A.1) ([illegible]) [illegible] [illegible]

$$(\phi_h^0, \phi_h^1) = \sum_{k=1}^{N} (m_k^0(h), m_k^1(h)) \bar{\varphi}^k(h),$$
$$(\phi^0, \phi^1) = \sum_{k \in \mathbb{N}} (m_k^0, m_k^1) \varphi^k.$$

[illegible] $m_k^0(h) = m_k^1(h) = 0$ [illegible] $k \geq N+1$ [illegible]

$$\left\{ m_k^0(h) \right\}_{k \in \mathbb{N}} \underset{h \to 0}{\to} \left\{ m_k^0 \right\}_{k \in \mathbb{N}} \quad \text{[illegible]} \quad \hbar^1, \tag{A.53}$$

$$\left\{ m_k^1(h) \right\}_{k \in \mathbb{N}} \underset{h \to 0}{\to} \left\{ m_k^1 \right\}_{k \in \mathbb{N}} \quad \text{[illegible]} \quad l^2, \tag{A.54}$$

[illegible]

$$-\frac{\phi_N(h,t)}{h} \underset{h \to 0}{\to} \partial_x \phi(1,t) \quad \text{[illegible]} \quad h \to 0 \quad \text{[illegible]} \quad L^2(0,T). \tag{A.55}$$

[illegible]. As a consequence of Lemma A.2 we have

$$-\frac{\phi_N(h,t)}{h} \rightharpoonup \partial_x \phi(1,t) \ \text{ weakly as } \ h \to 0 \ \text{ in } \ L^2(0,T),$$

where

$$\frac{\phi_N(h,t)}{h} = -\sum_{k=1}^{N} m_k(h,t) \frac{\sin(k\pi h)}{h} \cos(k\pi)$$

and

$$\partial_x \phi(1,t) = \sum_{k\in\mathbb{N}} k\pi\, m_k(t)\cos(k\pi).$$

To conclude the proof of (A.55) it is sufficient to show that the following holds:

$$\int_0^T \left| \sum_{k=1}^{N} m_k(h,t)\frac{\sin(k\pi h)}{h}\cos(k\pi) - \sum_{k\in\mathbb{N}} m_k(t) k\pi \cos(k\pi)\right|^2 dt \to 0 \ \text{ as } \ h\to 0.$$

Thus, it is sufficient to check, for every $\varepsilon > 0$, that there exists $M_\varepsilon > 0$ such that

$$\int_0^T \left| \sum_{k\geq M_\varepsilon} \left(m_k(h,t)\frac{\sin(k\pi h)}{h} - m_k(t)k\pi\right)\cos(k\pi)\right|^2 dt \leq \varepsilon. \tag{A.56}$$

In view of the convergences (A.53)–(A.55) and the conservation of the energies E_h, we deduce that

$$\overrightarrow{m}(h,t) = \{m_k(h,t)\}_{k\in\mathbb{N}} \underset{h\to 0}{\to} \overrightarrow{m}(t) = \{m_k(t)\}_{k\in\mathbb{N}} \quad \text{strongly in } \ \hbar^1 \tag{A.57}$$

for all $0 \leq t \leq T$. Moreover, for $m_k(h,t) = m_k(h)e^{i\sqrt{\mu_k(h)}t}$ and $m_k(t) = m_k e^{ik\pi t}$, as a consequence of inequality (A.30) (see [17]), we deduce that

$$\int_0^T \left| \sum_{k\geq M} m_k(h,t)\frac{\sin(k\pi h)}{h}\cos(k\pi)\right|^2 dt \leq A\sum_{k\geq M} |m_k(h)|^2 k^2\pi^2 \quad \forall M \geq 1. \tag{A.58}$$

Due to the conservation of energy and taking hypotheses (A.53)–(A.54) into account, we have that

$$\sum_{k\geq M_\varepsilon} |m_k(h)|^2 k^2\pi^2 \leq \frac{\varepsilon}{2} \quad \text{for } M_\varepsilon \text{ large enough.} \tag{A.59}$$

On the other hand, by the same argument [7]

$$\int_0^T \left| \sum_{k\geq M_\varepsilon} m_k(t) k\pi\cos(k\pi)\right|^2 dt \leq \hat{A}\sum_{k\geq M_\varepsilon} |m_k|^2 k^2\pi^2 \leq \frac{\varepsilon}{2}. \tag{A.60}$$

Combining inequality (A.60) with (A.58)–(A.59), the convergence (A.55) follows immediately. □

In what follows we obtain the convergence results which have been used in the paper.

3.5. For every $e_L = (\phi_L^0, \phi_L^1) \in H_0^1(0,1)\times L^2(0,1)$ fixed, we denote by (m_k^0, m_k^1) the Fourier coefficients and by $(m_{k,\tilde{L}}^0, m_{k,\tilde{L}}^1)$ the Fourier coefficients in every $V_{\tilde{L}}$ of $e_L \in (V_L)^2 \subset (V_{\tilde{L}})^2$. Trivially we have

$$\left(m_{k,\tilde{L}}^0, m_{k,\tilde{L}}^1\right) \underset{\tilde{L}\to\infty}{\to} \left(m_k^0, m_k^1\right) \ \text{ strongly in } \ \hbar^1\times l^2,$$

and therefore, by Lemma A.3, we obtain the strong convergence (3.48). □

REFERENCES

[1] M. Asch and G. Lebeau, *Geometrical aspects of exact boundary controllability of the wave equation. A numerical study*, ESAIM Control Optim. Calc. Var., 3 (1998), pp. 163–212.

[2] S. Beuchler, R. Schneider, and C. Schwab, *Multiresolution weighted norm equivalences and applications*, Numer. Math., 98 (2004), pp. 67–97.

[3] W. Dahmen, *Wavelet and Multiscale methods for operator equations*, Acta Numer., 6 (1997), pp. 55–228.

[4] R. Glowinski, *Ensuring well-posedness by analogy: Stokes problem and boundary control of the wave equation*, J. Comput. Phys., 103 (1992), pp. 189–221.

[5] R. Glowinski, C. H. Li, and J.-L. Lions, *A numerical approach to the exact boundary controllability of the wave equation* I. *Dirichlet controls: Description of the numerical methods*, Japan J. Math. Pures Appl., 68 (1990), pp. 1–76.

[6] J. A. Infante and E. Zuazua, *Boundary observability of the space semi-discretizations of the* 1*-D wave equation*, M2AN Math. Model. Numer. Anal., 33 (1999), pp. 407–438.

[7] J.-L. Lions, *Contrôlabilité exacte, stabilisation et perturbations du systemes distribués. Tome* 1*: Contrôlabilité exacte*, Rech. Math. Appl. 8, Masson, Paris, 1988.

[8] P. Loreti and M. Mehrenberger, *An Ingham type proof for a two-grid observability theorem*, ESAIM Control Optim. Calc. Var., to appear.

[9] M. Negreanu and E. Zuazua, *Uniform boundary controllability of a discrete* 1*-D wave equation*, Systems Control Lett., 48 (2003), pp. 261–279.

[10] M. Negreanu and E. Zuazua, *Convergence of a multigrid method for the controllability of a* 1*-d wave equation*, C. R. Math. Acad. Sci. Paris, 338 (2004), pp. 413–418.

[11] M. Negreanu, *Métodos numéricos para el análisis de la propagación, observación y control de ondas*, Ph.D. Thesis, Departamento de Matemática Aplicada, Universidad Complutense de Madrid, Spain, 2004.

[12] M. Negreanu and E. Zuazua, *Discrete Ingham inequalities and applications*, C. R. Math. Acad. Sci. Paris, 338 (2004), pp. 281–286.

[13] M. Negreanu and E. Zuazua, *Discrete Ingham inequalities and applications*, SIAM J. Numer. Anal., 44 (2006), pp. 412–448.

[14] N. M. Newmark, *A method of computation for structural dynamics*, J. Engrg. Mech. Division, ASCE, 1959, pp. 67–94.

[15] L. N. Trefethen, *Group velocity in finite difference schemes*, SIAM Rev., 24 (1982), pp. 113–136.

[16] R. Vichnevetsky and J. B. Bowles, *Fourier Analysis of Numerical Approximations of Hyperbolic Equations*, SIAM Stud. Appl. Math. 5, SIAM, Philadelphia, 1982.

[17] E. Zuazua, *Boundary observability for the finite-difference space semi-discretizations of the* 2*-d wave equation in the square*, J. Math. Pures Appl. (9), 78 (1999), pp. 523–563.

SIAM J. SCI. COMPUT.
Vol. 28, No. 5, pp. 1886–1906
© 2006 Society for Industrial and Applied Mathematics

A PARALLEL IMPLEMENTATION OF DUAL-PRIMAL FETI METHODS FOR THREE-DIMENSIONAL LINEAR ELASTICITY USING A TRANSFORMATION OF BASIS*

AXEL KLAWONN† AND OLIVER RHEINBACH†

Abstract. Dual-primal FETI methods for linear elasticity problems in three dimensions are considered. These are nonoverlapping domain decomposition methods where some primal continuity constraints across subdomain boundaries are required to hold throughout the iterations, whereas most of the constraints are enforced by Lagrange multipliers. An algorithmic framework for dual-primal FETI methods is described together with a transformation of basis to implement the primal constraints. Numerical results obtained from a parallel implementation of these algorithms applied to a model benchmark problem with structured meshes and to problems with more complicated geometries from industrial and biological applications using unstructured meshes are provided. These results show that the presented dual-primal FETI algorithms are numerical and parallel scalable.

Key words. domain decomposition, Lagrange multipliers, FETI, preconditioners, elliptic systems, elasticity, finite elements, parallel computing

AMS subject classifications. 65F10, 65N30, 65N55

DOI. 10.1137/050624364

1. Introduction.

Dual-primal FETI (FETI-DP) methods are the most recent members of the family of finite element tearing and interconnecting (FETI) domain decomposition methods. The FETI methods, introduced by Farhat and Roux [18], are all dual iterative substructuring methods for partial differential equations. In these methods the original domain, on which the given partial differential equation has to be solved, is decomposed into nonoverlapping subdomains. The intersubdomain continuity is then enforced by Lagrange multipliers across the interface defined by the subdomain boundaries. For further results and references, see, e.g., [17, 13, 5, 34, 24, 41]. In FETI-DP methods, some continuity constraints on the primal displacement variables are forced to hold throughout iterations, as in primal substructuring algorithms, while the other constraints are enforced by the use of Lagrange multipliers, as in standard one-level FETI methods. The primal constraints have to be chosen such that the local subproblems become invertible and such that a parallel scalable method is obtained; the primal constraints provide a coarse problem for these domain decomposition methods. Different choices of primal constraints are possible; see the discussion below. Each of these choices defines a different coarse space and thus a different member of the family of FETI-DP algorithms.

FETI-DP algorithms were introduced by Farhat et al. in [14] for linear elasticity problems in the plane and then extended by Farhat, Lesoinne, and Pierson [15] to three-dimensional elasticity problems; see also Pierson [37]. The first theoretical analysis for two-dimensional, scalar elliptic partial differential equations of second and fourth order with only small coefficient jumps across the subdomain boundaries was given by Mandel and Tezaur [35]; it was shown that the condition number is bounded polylogarithmically as a function of the dimension of the individual subregion

*Received by the editors February 14, 2005; accepted for publication (in revised form) April 25, 2006; published electronically November 16, 2006.
http://www.siam.org/journals/sisc/28-5/62436.html
†Fachbereich Mathematik, Universität Duisburg-Essen, Universitätsstraße 3, 45117 Essen, Germany (axel.klawonn@uni-due.de, http://www.uni-essen.de/numa; oliver.rheinbach@uni-due.de).

problems. The family of FETI-DP algorithms for scalar, second order elliptic problems in three dimensions was extended by Klawonn and Widlund [23] and Klawonn, Widlund, and Dryja [27, 28]; see also [41]. There, constraints on averages over subdomain edges and faces are used to obtain algorithms with an improved scalability for three-dimensional problems. We note that such averages were already used in Dryja, Sarkis, and Widlund [11] and Dryja, Smith, and Widlund [12] to construct primal iterative substructuring methods which are scalable in three dimensions. In [15, 37] similar constraints on averages are used to define FETI-DP methods which are scalable for three-dimensional linear elasticity problems, although the definition of subdomain edges is different from the one used in [23, 27, 28]; see also the discussion at the end of section 2. We also note that an important feature of the algorithms presented in [23, 27, 28] was to develop variants of FETI-DP with smaller coarse problems while still maintaining scalability and a good convergence bound. In fact, in [27, 28] a theory was provided which shows that the condition number in three dimensions can again be bounded polylogarithmically as a function of the dimension of the individual subdomain problems and that the bounds can otherwise be made independent of the number of subdomains, the mesh size, and the jumps in the coefficients.

More recently, new variants of FETI-DP for three-dimensional, linear elasticity problems were provided by Klawonn and Widlund [25, 26] together with a theoretical analysis proving a polylogarithmic condition number estimate as in the scalar case, which is also robust with respect to discontinuities in the material coefficients. For benign elasticity problems, it is shown that selecting an appropriate set of edge averages as primal constraints is sufficient for obtaining good polylogarithmic bounds. For arbitrary coefficient distributions, certain first order moments on selected edges have to be added as primal constraints as well as constraints at some of the vertices in order to obtain robust, polylogarithmic bounds. An implementational variant of FETI-DP is presented in [25], using a transformation of basis for the primal edge averages and first order moments. Originally, optional Lagrange multipliers are used for the implementation of constraints of this type; cf. [15].

Strongly related to FETI-DP methods are the more recently developed Neumann–Neumann methods with constraints, also known as balancing domain decomposition methods by constraints (BDDC); cf. [10, 32, 33]. Neumann–Neumann methods with primal vertex constraints were also developed independently by Cros [7] and Fragakis and Papadrakakis [19]; in the latter, this approach is called P-FETIDP. In [19], it was also experimentally observed that for the balancing Neumann–Neumann and one-level FETI methods, the eigenvalues, which are different from one and zero, are the same. It was first proved in Mandel, Dohrmann, and Tezaur [33] that the eigenvalues of BDDC and FETI-DP, which are not one or zero, are the same; see also Li and Widlund [31] for another approach.

Recently, new results for inexact FETI-DP and BDDC algorithms have been obtained for the inexact solution of the subdomain or coarse problems. For inexact BDDC methods, see Tu [42, 43], Dohrmann [9], and Li and Widlund [30], and for inexact FETI-DP algorithms, see Klawonn and Rheinbach [22].

The purpose of this article is to present results obtained from a parallel implementation of some of the algorithms developed in Klawonn and Widlund [25] using only average constraints over selected edges and to show numerical and parallel scalability of these algorithms in three dimensions. Here, we restrict ourselves to homogeneous elasticity problems without any jumps in the material coefficients; numerical experiments for heterogeneous problems are part of ongoing research; see Klawonn and

Rheinbach [21]. We note that the implementation using a transformation of basis leads to a symmetric, positive definite coarse problem in contrast to the approach using optional Lagrange multipliers, where the coarse problem is symmetric and indefinite; cf. [15, 25]. Another interesting feature is the fact that for homogeneous problems, no vertex constraints have to be used and only edge averages over selected edges are needed. Our experiments show that using a transformation of basis for the implementation of the primal edge average constraints leads to a parallel scalable domain decomposition method. We first report on experiments for model benchmark problems with structured meshes. It is shown that the experimental results confirm the theory in [25] in the case of structured domain decompositions using cubic subdomains. In scientific computing, solving real-world, three-dimensional problems from structural mechanics applications, usually unstructured meshes occur and the decomposition into subdomains is obtained by using automatic graph partitioners. In general, this leads to irregularly shaped subdomains with jagged boundaries. To be of relevance in scientific computing, it is of utmost importance for a domain decomposition method to show applicability to and scalability in such situations. We note that such a situation is not covered by theory anymore. Thus, it is important to provide numerical results showing scalability experimentally. We see that our algorithms can also be successfully applied to cases which are not covered by the theory anymore, such as unstructured domain decompositions using mesh partitioners or faces, edges, and vertices defined by equivalence classes, which can, e.g., lead to curved or jagged edges. We also show that the parallel implementation of our FETI-DP method applied to some three-dimensional, industrial applications, in almost all cases, shows better scalability and uses less CPU time compared to a parallel direct solver. Furthermore, we successfully considered an elasticity problem from a biological application. We note that the bone structure considered in this example is a very difficult and challenging domain since it contains many holes and thin bridges. We also carried out numerical experiments which show the robustness of our method with respect to reduction of the relative residual; it is shown that it can be reduced to 10^{-14} without stagnation of the method.

The remainder of this article is organized as follows. In section 2, we introduce the equations of linear elasticity in three dimensions, the discretization by finite elements and the partition of the domain into substructures. In section 3, we present the algorithmic framework for our FETI-DP methods and in section 4 we describe how to perform the transformation of basis to implement the primal constraints. Finally, in section 5 we present the numerical results obtained from a parallel implementation of our methods. First, we apply it to a benchmark problem on the unit cube, testing for numerical and parallel scalability, confirming the theoretical findings in [25]. We then show the parallel performance of our algorithms applied to geometries obtained from industrial and biological applications. Finally, we show that the approach presented in this article is very stable with respect to the residual reduction.

2. The equations of linear elasticity, finite elements, and geometry. The equations of linear elasticity model the displacement of a linear elastic material under the action of external and internal forces. The elastic body occupies a domain $\Omega \subset \mathbb{R}^3$, which is assumed to be polyhedral and of diameter one. We denote its boundary by $\partial\Omega$ and assume that one part of it, $\partial\Omega_D$, is clamped, i.e., with homogeneous Dirichlet boundary conditions, and that the rest, $\partial\Omega_N := \partial\Omega \setminus \partial\Omega_D$, is subject to a surface force $\mathbf{g}$, i.e., a natural boundary condition. We can also introduce a body force $\mathbf{f}$, e.g., gravity. With $\mathbf{H^1}(\Omega) := (H^1(\Omega))^3$, the appropriate space for a variational

formulation is the Sobolev space $\mathbf{H_0^1}(\Omega, \partial\Omega_D) := \{\mathbf{v} \in \mathbf{H^1}(\Omega) : \mathbf{v} = \mathbf{0} \text{ on } \partial\Omega_D\}$. The linear elasticity problem consists in finding the displacement $\mathbf{u} \in \mathbf{H_0^1}(\Omega, \partial\Omega_D)$ of the elastic body Ω, such that

(2.1)
$$\int_\Omega G(\mathbf{x})\varepsilon(\mathbf{u}) : \varepsilon(\mathbf{v})d\mathbf{x} + \int_\Omega G(\mathbf{x})\,\beta(\mathbf{x})\,\mathrm{div}\,\mathbf{u}\,\mathrm{div}\,\mathbf{v}\,d\mathbf{x} = \langle \mathbf{F}, \mathbf{v}\rangle \quad \forall \mathbf{v} \in \mathbf{H_0^1}(\Omega, \partial\Omega_D).$$

Here G and β are material parameters which depend on the Young modulus $E > 0$ and the Poisson ratio $\nu \in (0, \frac{1}{2}]$; we have $G = E/(1+\nu)$ and $\beta = \nu/(1-2\nu)$. In this article, we consider only the case of compressible elasticity, which means that the Poisson ratio ν is bounded away from $\frac{1}{2}$. Furthermore, $\varepsilon_{ij}(\mathbf{u}) := \frac{1}{2}(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i})$ is the linearized strain tensor, and

$$\varepsilon(\mathbf{u}) : \varepsilon(\mathbf{v}) = \sum_{i,j=1}^{3} \varepsilon_{ij}(\mathbf{u})\varepsilon_{ij}(\mathbf{v}), \quad \langle \mathbf{F}, \mathbf{v}\rangle := \int_\Omega \mathbf{f}^T\mathbf{v}\,d\mathbf{x} + \int_{\partial\Omega_N} \mathbf{g}^T\mathbf{v}\,d\sigma.$$

For convenience, we also introduce the notation

$$(\varepsilon(\mathbf{u}), \varepsilon(\mathbf{v}))_{L_2(\Omega)} := \int_\Omega \varepsilon(\mathbf{u}) : \varepsilon(\mathbf{v})d\mathbf{x}.$$

The bilinear form associated with linear elasticity is then

$$a(\mathbf{u}, \mathbf{v}) = (G\,\varepsilon(\mathbf{u}), \varepsilon(\mathbf{v}))_{L_2(\Omega)} + (G\,\beta\,\mathrm{div}\,\mathbf{u}, \mathrm{div}\,\mathbf{v})_{L_2(\Omega)}.$$

The wellposedness of the linear system (2.1) follows immediately from the continuity and ellipticity of the bilinear form $a(\cdot,\cdot)$, where the first follows from elementary inequalities and the latter from Korn's first inequality; see, e.g., Ciarlet [6].

The null space $\mathbf{ker}\,(\varepsilon)$ of ε is the space of rigid body motions which is spanned by the three translations

$$\mathbf{r}_1 := \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{r}_2 := \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \quad \mathbf{r}_3 := \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{2.2}$$

and the three rotations

$$\mathbf{r}_4 := \begin{bmatrix} x_2 - \hat{x}_2 \\ -x_1 + \hat{x}_1 \\ 0 \end{bmatrix}, \quad \mathbf{r}_5 := \begin{bmatrix} -x_3 + \hat{x}_3 \\ 0 \\ x_1 - \hat{x}_1 \end{bmatrix}, \quad \mathbf{r}_6 := \begin{bmatrix} 0 \\ x_3 - \hat{x}_3 \\ -x_2 + \hat{x}_2 \end{bmatrix}. \tag{2.3}$$

Here $\hat{\mathbf{x}} \in \Omega$ to shift the origin to a point in Ω.

We will consider only compressible elastic materials. It is therefore sufficient to discretize our elliptic problem (2.1) by low order, conforming finite elements, e.g., linear or trilinear elements.

Let us assume that a triangulation τ^h of Ω is given which is shape regular and has a typical diameter of h. We denote by $\mathbf{W}^h := \mathbf{W}^h(\Omega) \subset \mathbf{H_0^1}(\Omega, \partial\Omega_D)$ the corresponding conforming finite element space of finite element functions. The corresponding discrete problem is then

$$a(\mathbf{u}_h, \mathbf{v}_h) = \langle \mathbf{F}, \mathbf{v}_h\rangle \quad \forall \mathbf{v}_h \in \mathbf{W}^h. \tag{2.4}$$

When there is no risk of confusion, we will drop the subscript h.

Let the domain $\Omega \subset \mathbb{R}^3$ be decomposed into nonoverlapping subdomains $\Omega_i, i = 1, \ldots, N$, each of which is the union of finite elements with matching finite element nodes on the boundaries of neighboring subdomains across the interface Γ. The interface Γ is the union of three different groups of open sets, namely, subdomain faces, edges, and vertices. We denote individual faces, edges, and vertices by $\mathcal{F}, \mathcal{E}$, and $\mathcal{V}$, respectively. To define faces, edges, and vertices, we introduce certain equivalence classes. Let us denote the sets of nodes on $\partial\Omega$, $\partial\Omega_i$, and Γ by $\partial\Omega_h$, $\partial\Omega_{i,h}$, and Γ_h, respectively. For any interface nodal point $x \in \Gamma_h$, we define

$$\mathcal{N}_x := \{j \in \{1, \ldots, N\} : x \in \partial\Omega_j\},$$

i.e., $\mathcal{N}_x$ is the set of indices of all subdomains with x in the closure of the subdomain. For a node x we define the multiplicity as $|N_x|$.

Associated with the nodes of the finite element mesh, we have a graph, the nodal graph, which represents the node-to-node adjacency. For a given node $x \in \Gamma_h$, we denote by $\mathcal{C}_{con}(x)$ the connected component of the nodal subgraph, defined by $\mathcal{N}_x$, to which x belongs. For two interface points $x, y \in \Gamma_h$, we introduce an equivalence relation by

$$x \sim y :\Longleftrightarrow \mathcal{N}_x = \mathcal{N}_y \quad \text{and} \quad y \in \mathcal{C}_{con}(x).$$

We can now describe faces, edges, and vertices using their equivalence classes. Here, $|G|$ denotes the cardinality of the set G. We define the following.

Definition 2.1.

$$\begin{aligned} x \in \mathcal{F} &:\Longleftrightarrow |\mathcal{N}_x| = 2. \\ x \in \mathcal{E} &:\Longleftrightarrow |\mathcal{N}_x| \geq 3 \quad \exists y \in \Gamma_h, y \neq x, \quad y \sim x. \\ x \in \mathcal{V} &:\Longleftrightarrow |\mathcal{N}_x| \geq 3 \quad \nexists y \in \Gamma_h, \quad x \sim y. \end{aligned}$$

In the case of a decomposition into regular substructures, e.g., cubes or tetrahedrons, our definition of faces, edges, and vertices conforms to our basic geometric intuition; see Figure 2.2. On the other hand, for subdomains generated by an automatic mesh partitioner, the situation can be quite complicated. We can, e.g., have several edges with the same index set $\mathcal{N}_x$ or an edge and a vertex with the same $\mathcal{N}_x$. In practice, we can also have situations when there are not enough edges and potential edge constraints for some subdomains. Then we have to use constraints on some extra edges on $\partial\Omega_N$, which otherwise would be regarded as part of a face. A similar problem might occur for flat structures for which additional constraints might be required for each subdomain. Therefore, we introduce an alternative definition of edges.

Definition 2.2. N_x $|N_x| \geq 3$ $|N_x| \geq 2$ x $\partial\Omega_N$

If needed, we will increase the number of edges in unstructured cases by switching locally from the definition of edges given in Definition 2.1 to Definition 2.2 and by splitting edges into several edges.

Let us note that sometimes the term is used differently in the literature; cf., e.g., [15, 37]. Also, instead of vertices, the term is often used although usually with a different meaning than vertices in our terminology; cf. [15, 37, 29]. A simple choice is to define corners in three dimensions as nodes which either belong to more than three subdomains (see Pierson [37, p. 24]) or to more than four subdomains (see Lesoinne [29]). A more elaborate algorithmic choice, which can reduce the number

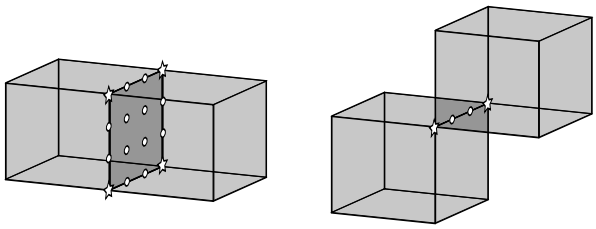


FIG. 2.1. *Definition of edges in* [15, 36, 37]. *When the nodes, denoted by stars, are selected as corners, the set of edge nodes is given by those nodes denoted by circles. Left: The intersection of the closure of the two subdomains consists of one edge (which looks like a geometric face) and four corners based on this definition of edges. Right: The intersection of the closure of the two subdomains consists of one edge and two corners.*

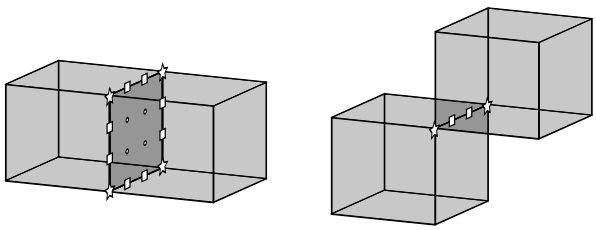


FIG. 2.2. *Definition* 2.1 *of vertices, edges, and faces. Vertices are denoted by stars, edge nodes by squares, and face nodes by small circles. Left: The intersection of the closure of the two subdomains consists of one face, four edges, and four vertices. Right: The intersection of the closure of the two subdomains consists of one edge and two vertices.*

of corners, is given in [29]. For completeness, we also give the definition of edges in three dimensions as used in [15, 37]; see Pierson [36]. There, an edge between two subdomains is the set of interface nodes, excluding corners, which is common to the boundaries of both subdomains. To illustrate the subtle difference between this and our definition, we consider a structured decomposition of the unit cube into cubic subdomains. For two different situations of interior neighboring subdomains, we illustrate the different definitions in Figures 2.1 and 2.2. Here, an edge in Figure 2.1 can be, for example, either an edge as in Definition 2.1 or the union of, in our terminology, a face with the edges belonging to its closure; cf. also Figure 2.2.

Let us note that in the algorithms in this article, we exclusively use constraints over edges; no vertices or faces are needed.

3. The FETI-DP algorithms. In this section, we introduce a matrix formulation of the FETI-DP algorithms considered in this article. For each subdomain Ω_i, $i = 1, \ldots, N$, we assemble local stiffness matrices K_i and local load vectors $\mathbf{f}_i$ of the elasticity problem (2.4). Furthermore, we denote by $\mathbf{u}_i$ the local solution vectors of nodal values. In the dual-primal FETI methods, we distinguish between dual and primal displacement variables according to the way the continuity of the solution in those variables is established. Dual displacement variables are those for which the continuity is enforced by a continuity constraint and Lagrange multipliers $\boldsymbol{\lambda}$, and thus continuity is not established until convergence of the iterative method is reached, as in the classical one-level FETI methods. On the other hand, continuity of the primal displacement variables is enforced explicitly in each iteration step by subassembly of the local stiffness matrices K_i at the primal displacement variables. This subassembly yields a symmetric, positive definite stiffness matrix $\widetilde{K}$ which is not block diagonal

anymore but is coupled at the primal displacement variables. Let us note that this coupling yields a global problem which is necessary to obtain a parallel and numerical scalable algorithm. An obvious choice of primal variables are (selected) vertices for which a partial assembly is then carried out. We will see in section 4 that there are other possible choices, such as edge averages over selected edges; cf. Klawonn and Widlund [25] for other possible choices. If a transformation of basis is used, then the edge average constraints can be treated algorithmically exactly in the same way as primal vertices; cf. section 4.

We will use the subscripts I, Δ, and Π to denote the interior, dual, and primal displacement variables, respectively, and obtain for the local stiffness matrices, load vectors, and solution vectors of nodal values

$$K^{(i)} = \begin{bmatrix} K_{II}^{(i)} & K_{\Delta I}^{(i)T} & K_{\Pi I}^{(i)T} \\ K_{\Delta I}^{(i)} & K_{\Delta\Delta}^{(i)} & K_{\Pi\Delta}^{(i)T} \\ K_{\Pi I}^{(i)} & K_{\Pi\Delta}^{(i)} & K_{\Pi\Pi}^{(i)} \end{bmatrix}, \quad \mathbf{u}^{(i)} = \begin{bmatrix} \mathbf{u}_I^{(i)} \\ \mathbf{u}_\Delta^{(i)} \\ \mathbf{u}_\Pi^{(i)} \end{bmatrix}, \quad \mathbf{f}^{(i)} = \begin{bmatrix} \mathbf{f}_I^{(i)} \\ \mathbf{f}_\Delta^{(i)} \\ \mathbf{f}_\Pi^{(i)} \end{bmatrix}.$$

By subassembly in the primal displacement variables, we obtain

$$\widetilde{K} = \begin{bmatrix} K_{BB} & \widetilde{K}_{\Pi B}^T \\ \widetilde{K}_{\Pi B} & \widetilde{K}_{\Pi\Pi} \end{bmatrix},$$

where a tilde indicates the subassembled matrices and where

$$K_{BB} = \begin{bmatrix} K_{BB}^{(1)} & O & \cdots & O \\ O & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & O \\ O & \cdots & O & K_{BB}^{(N)} \end{bmatrix}, \quad K_{BB}^{(i)} = \begin{bmatrix} K_{II}^{(i)} & K_{\Delta I}^{(i)T} \\ K_{\Delta I}^{(i)} & K_{\Delta\Delta}^{(i)} \end{bmatrix},$$

and

$$\widetilde{K}_{\Pi B} = [\widetilde{K}_{\Pi B}^{(1)} \cdots \widetilde{K}_{\Pi B}^{(N)}].$$

Introducing local assembly operators $R_\Pi^{(i)T}$ which map from the local primal displacement variables $\mathbf{u}_\Pi^{(i)}$ to the global, assembled $\widetilde{\mathbf{u}}_\Pi$, we have

$$\widetilde{K}_{\Pi B}^{(i)} = R_\Pi^{(i)T} K_{\Pi B}^{(i)}, \qquad \mathbf{u}_\Pi^{(i)} = R_\Pi^{(i)} \widetilde{\mathbf{u}}_\Pi, \quad i = 1, \ldots, N,$$

and

$$\widetilde{K}_{\Pi\Pi} = \sum_{i=1}^{N} R_\Pi^{(i)T} K_{\Pi\Pi}^{(i)} R_\Pi^{(i)}.$$

We also introduce the notation

$$\mathbf{u}_B^T = [\mathbf{u}_I^T \ \mathbf{u}_\Delta^T], \quad \mathbf{f}_B^T = [\mathbf{f}_I^T \ \mathbf{f}_\Delta^T], \mathbf{u}_B^{(i)T} = [\mathbf{u}_I^{(i)T} \ \mathbf{u}_\Delta^{(i)T}], \quad \mathbf{f}_B^{(i)T} = [\mathbf{f}_I^{(i)T} \ \mathbf{f}_\Delta^{(i)T}].$$

To guarantee continuity at the dual displacement variables, we introduce a jump operator B, which is constructed from $\{0, 1, -1\}$, in such a way that the values of the solution $\mathbf{u}_\Delta$, associated with more than one subdomain, coincide when $B\mathbf{u}_B = 0$; the

interior variables $\mathbf{u}_I$ remain unchanged and thus the corresponding entries in B are zero. These constraints are very simple and just express that the nodal values coincide across the interface; clearly, we do not need any constraints associated with the primal displacement variables. However, we will otherwise use all possible constraints and thus work with a fully redundant set of Lagrange multipliers as in [24, section 5]; cf. also [38]. Thus, for an edge node common to four subdomains, we will use six constraints rather than choosing as few as three.

We can now reformulate the finite element discretization of (2.4) as

$$\begin{bmatrix} K_{BB} & \widetilde{K}_{\Pi B}^T & B^T \\ \widetilde{K}_{\Pi B} & \widetilde{K}_{\Pi\Pi} & O \\ B & O & O \end{bmatrix} \begin{bmatrix} \mathbf{u}_B \\ \widetilde{\mathbf{u}}_\Pi \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \mathbf{f}_B \\ \widetilde{\mathbf{f}}_\Pi \\ \mathbf{0} \end{bmatrix}.$$

Elimination of the interior and dual displacement variables $\mathbf{u}_B$ and of the primal displacement variables $\widetilde{\mathbf{u}}_\Pi$ leads to

$$\begin{bmatrix} K_{BB} & \widetilde{K}_{\Pi B}^T & B^T \\ O & \widetilde{S}_{\Pi\Pi} & -\widetilde{S}_{\Lambda\Pi}^T \\ O & O & -F \end{bmatrix} \begin{bmatrix} \mathbf{u}_B \\ \widetilde{\mathbf{u}}_\Pi \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \mathbf{f}_B \\ \widetilde{\mathbf{f}}_\Pi - \widetilde{K}_{\Pi B} K_{BB}^{-1} \mathbf{f}_B \\ -\mathbf{d} \end{bmatrix}$$

with

$$\begin{aligned} \widetilde{S}_{\Pi\Pi} &= \widetilde{K}_{\Pi\Pi} - \widetilde{K}_{\Pi B} K_{BB}^{-1} \widetilde{K}_{\Pi B}^T, \qquad \widetilde{S}_{\Lambda\Pi}^T = \widetilde{K}_{\Pi B} K_{BB}^{-1} B^T, \\ F &= B K_{BB}^{-1} B^T + B K_{BB}^{-1} \widetilde{K}_{\Pi B}^T \widetilde{S}_{\Pi\Pi}^{-1} \widetilde{K}_{\Pi B} K_{BB}^{-1} B^T, \\ \mathbf{d} &= B K_{BB}^{-1} \mathbf{f}_B + B K_{BB}^{-1} \widetilde{K}_{\Pi B}^T \widetilde{S}_{\Pi\Pi}^{-1} \left(\widetilde{\mathbf{f}}_\Pi - \widetilde{K}_{\Pi B} K_{BB}^{-1} \mathbf{f}_B \right). \end{aligned}$$

Since K_{BB} is a block-diagonal matrix, we obtain

$$\begin{aligned} F = &\sum_{i=1}^{N} B^{(i)} \left(K_{BB}^{(i)} \right)^{-1} B^{(i)T} \\ &+ \left(\sum_{i}^{N} B^{(i)} (K_{BB}^{(i)})^{-1} \widetilde{K}_{\Pi B}^{(i)T} \right) \widetilde{S}_{\Pi\Pi}^{-1} \left(\sum_{j}^{N} \widetilde{K}_{\Pi B}^{(j)} (K_{BB}^{(j)})^{-1} B^{(j)T} \right). \end{aligned} \tag{3.1}$$

We see that in a matrix-vector multiplication with F, we only need to solve local linear systems in the first sum. To apply the second term in (3.1) to a vector, we first apply the right sum in it to that vector; this represents a gather operation. Next, we have to solve a linear system with the matrix $\widetilde{S}_{\Pi\Pi}$, which yields our global problem needed for scalability. Finally, we apply the left sum of the second term in (3.1) to the result; this represents a scatter operation. A similar form can be derived for the right-hand side $\mathbf{d}$.

To define the FETI-DP Dirichlet preconditioner M^{-1}, we first need to introduce a scaled jump operator B_D; this is done by scaling the contributions of B associated with the dual displacement variables from individual subdomains. We can write

$$B_D = [B_D^{(1)}, \ldots, B_D^{(N)}],$$

where the $B_D^{(i)}$ are defined as follows: Each row of $B^{(i)}$ with a nonzero entry corresponds to a Lagrange multiplier connecting the subdomain Ω_i with a neighboring

subdomain Ω_j at a point $x \in \partial\Omega_{i,h} \cap \partial\Omega_{j,h}$. We obtain $B_D^{(i)}$ by multiplying each such row of $B^{(i)}$ with $G_j(x)/\sum_{k\in\mathcal{N}_x} G_k(x)$, where $G_l(x) := G(x)$ for $x \in \partial\Omega_l, l = 1, \ldots, N$. In the homogeneous case, this results in scaling each such row with the reciprocal of the multiplicity $|\mathcal{N}_x|$ of the node $x \in \partial\Omega_{i,h} \cap \partial\Omega_{j,h}$; thus it is also called multiplicity scaling.

Let us now consider the matrix representation of our preconditioner M^{-1}. For this, we need certain Schur complement matrices $S^{(i)}$, $i = 1, \ldots, N$. Each $S^{(i)}$ is obtained from the local stiffness matrices $K^{(i)}$ by eliminating the interior displacement variables. Denoting the displacement variables on the local interface $\partial\Omega_i$ by the subscript Γ, we then have

$$K^{(i)} = \begin{bmatrix} K_{II}^{(i)} & K_{\Gamma I}^{(i)T} \\ K_{\Gamma I}^{(i)} & K_{\Gamma\Gamma}^{(i)} \end{bmatrix}$$

and the local Schur complements $S^{(i)}$ are obtained as

$$S^{(i)} = K_{\Gamma\Gamma}^{(i)} - K_{\Gamma I}^{(i)}(K_{II}^{(i)})^{-1}K_{\Gamma I}^{(i)T}.$$

We now define the block-diagonal Schur complement matrix S as

$$S = \begin{bmatrix} S^{(1)} & & O \\ & \ddots & \\ O & & S^{(N)} \end{bmatrix}.$$

Our preconditioner is then given in matrix form by

$$M^{-1} = B_D R_\Gamma^T S R_\Gamma B_D^T = \sum_{i=1}^{N} B_D^{(i)} R_\Gamma^{(i)T} S^{(i)} R_\Gamma^{(i)} B_D^{(i)T}. \tag{3.2}$$

Here, $R_\Gamma^{(i)}$ are restriction matrices that restrict the degrees of freedom of a subdomain to its interface and $R_\Gamma = \operatorname{diag}_i(R_\Gamma^{(i)})$.

Let us note that generally the local Schur complements $S^{(i)}$ are never built explicitly; instead, in each iteration step and each application of M^{-1} to a vector, local linear systems with $K^{(i)}$ are solved. This corresponds to the solution of local Dirichlet problems; thus, this preconditioner is also known as the Dirichlet preconditioner. Clearly, a matrix-vector multiplication with M^{-1} requires only the solution of local linear systems and can be carried out completely in parallel. The Dirichlet preconditioner without scaling was first introduced in Farhat, Mandel, and Roux [16]. For early references to the scaled version used in (3.2), see, e.g., Rixen and Farhat [39] and Klawonn and Widlund [24]. Let us note that appropriate scaling can be important to obtain convergence results which are independent of jumps in the coefficients of the partial differential equation; see [24, 27, 25]. But also for homogeneous problems, scaling can be important to improve convergence and also in order to obtain a condition number result for the one-level FETI method which is of the order of $(1+\log(H/h))^2$ instead of $(1+\log(H/h))^3$, where H is the subdomain diameter; cf. [34] and [24].

The FETI-DP method can now be described as the standard preconditioned conjugate gradient (pcg) method for solving the preconditioned linear system

$$M^{-1}F\boldsymbol{\lambda} = M^{-1}\mathbf{d}.$$

4. Primal constraints and a change of basis. We have to decide how to choose the primal displacement variables. The simplest choice is to choose them as certain selected primal vertices of the subdomains; see Farhat et al. [14], where this approach was first considered. In this section, we show that edge average constraints can be treated the same way by using a transformation of basis. In the following, we will denote the FETI-DP algorithm which exclusively uses selected vertices as primal displacement constraints as Algorithm A; cf. Klawonn, Widlund, and Dryja [27], where this notation was introduced. In Lesoinne [29], an algorithm for the choice of selected vertices as primal variables is suggested in the context of choosing a relatively small number of vertices to guarantee invertibility of the subproblems and otherwise enhance the coarse problem by face averages using optional Lagrange multipliers.

For scalar, second order, elliptic differential equations in three dimensions, certain edge and face averages were suggested in [23, 27] as additional primal constraints. For linear elasticity problems in three dimensions, Farhat, Lesoinne, and Pierson [15] considered similar averages as primal constraints. Let us note that there, edges (faces) are defined differently from our definition; cf. the discussion at the end of section 2. In Klawonn and Widlund [25], edge averages and first order edge moments were introduced and analyzed in order to obtain scalable algorithms which are also robust for arbitrarily large jumps of the material coefficients across the subdomain interface. The algorithms described in this article are based on [25].

There are two different approaches to implement the edge and face constraints: one using optional Lagrange multipliers, which form a part of the global, coarse problem; cf. [15] and [25, section 6.1]; and the other using a change of basis; cf. [25, section 6.2]. The latter approach generally leads to smaller and computationally more efficient coarse problems. In our transformation of basis, we introduce edge averages explicitly as new (primal) variables into the FETI-DP system. Then we subassemble in these degrees of freedom. This leads to a symmetric, positive definite coarse problem. Using this approach, the invertibility of the local problems and the positive definiteness of the entire problem can be guaranteed without any vertex constraints; the rigid body modes of a subdomain can be controlled if a sufficient number of edge averages is chosen. For homogeneous materials or problems with only small jumps in the material coefficients across the interface, this approach is scalable without any vertex constraints; see [25] for a theoretical analysis and section 5 for numerical results. In fact, vertex constraints are needed only for problems with very special distributions of the material coefficients and large jumps of these coefficients across the interface; they are necessary to obtain a convergence rate which is uniformly bounded with respect to these jumps. A complete theoretical treatment of these cases is given in Klawonn and Widlund [25, section 8.3]; experimental results which confirm these theoretical considerations are presented in Klawonn and Rheinbach [21].

We now describe the approach using an explicit change of basis. As a result, the finite element functions associated with dual displacement vectors will have zero edge averages over primal edges. In addition, we introduce these averages as primal variables. Here and in the following, we define an edge as primal if at least one component of the displacements has the same edge average across the interface on this edge.

The transformation matrix T_E performs the desired change of basis from the new basis to the original nodal basis, and we then iterate in the space defined by the new basis. Denoting the edge unknowns in the new basis by $\widehat{\mathbf{u}}_E$, we have

$$\mathbf{u}_E = T_E \widehat{\mathbf{u}}_E.$$

Such a transformation matrix T_E can be constructed separately for each edge with three primal edge constraints. Ordering the three edge averages last, a possible choice of T_E is

$$T_E = \begin{bmatrix} I_3 & & O & I_3 \\ & \ddots & & \vdots \\ O & & I_3 & I_3 \\ -I_3 & \cdots & -I_3 & I_3 \end{bmatrix}, \tag{4.1}$$

where I_3 is the 3×3 identity matrix. We denote the resulting transformation, which operates on all relevant edges of $\partial\Omega_i$, by $T_E^{(i)}$. The transformation for all variables of one subdomain Ω_i is then of the form

$$T^{(i)} = \begin{bmatrix} I_I & O & O \\ O & I_{\overline{\Gamma}} & O \\ O & O & T_E^{(i)} \end{bmatrix},$$

where I_I and $I_{\overline{\Gamma}}$ are identity matrices. Here, we assume that the variables are ordered interior variables first, interface variables not related to the primal edges second, and the variables on the primal edges last, i.e., a typical vector of nodal unknowns is of the form $[\mathbf{u}_I^{(i)T}, \mathbf{u}_{\overline{\Gamma}}^{(i)T}, \mathbf{u}_E^{(i)T}]^T$. Here, we denote the interface variables not related to a primal edge by the subscript $\overline{\Gamma}$. We note that $T_E^{(i)}$ is a direct sum of the relevant transformation matrices associated with the primal edges of that subdomain; $T_E^{(i)}$ is a block-diagonal matrix where each block represents the transformation of a component of a primal edge.

Decomposing the subdomain stiffness matrices $K^{(i)}$ in the same manner, we obtain

$$K^{(i)} = \left[\begin{array}{cc|c} K_{II}^{(i)} & K_{I\overline{\Gamma}}^{(i)} & K_{IE}^{(i)} \\ K_{\overline{\Gamma}I}^{(i)} & K_{\overline{\Gamma}\overline{\Gamma}}^{(i)} & K_{\overline{\Gamma}E}^{(i)} \\ \hline K_{EI}^{(i)} & K_{E\overline{\Gamma}}^{(i)} & K_{EE}^{(i)} \end{array}\right].$$

Using the transformation $\mathbf{u}^{(i)} = T^{(i)}\widehat{\mathbf{u}}^{(i)}$, we obtain

$$T^{(i)T}K^{(i)}T^{(i)} = \left[\begin{array}{cc|c} K_{II}^{(i)} & K_{I\overline{\Gamma}}^{(i)} & K_{IE}^{(i)}T_E^{(i)} \\ K_{\overline{\Gamma}I}^{(i)} & K_{\overline{\Gamma}\overline{\Gamma}}^{(i)} & K_{\overline{\Gamma}E}^{(i)}T_E^{(i)} \\ \hline T_E^{(i)T}K_{EI}^{(i)} & T_E^{(i)T}K_{E\overline{\Gamma}}^{(i)} & T_E^{(i)T}K_{EE}^{(i)}T_E^{(i)} \end{array}\right], \tag{4.2}$$

where the upper left 2×2 block matrix is not affected by the basis transformation. The primal variables in the new basis consist now of averages, but we note that we might also have selected primal vertices as additional primal variables. The primal variables belonging to Ω_i are denoted by $\mathbf{u}_\Pi^{(i)}$ and the remaining, dual displacement variables by $\mathbf{u}_\Delta^{(i)}$. By construction, the basis functions associated with the new dual displacement variables have zero edge average over primal edges.

In the same manner the indices Δ_E and Π_E indicate the dual and primal displacement variables associated with the primal edge constraints. Denoting the transformed

matrices by an overline and ordering the primal edge variables last, we obtain

$$T^{(i)T}K^{(i)}T^{(i)} = \left[\begin{array}{cc|cc} K_{II}^{(i)} & K_{I\overline{\Gamma}}^{(i)} & \overline{K}_{I\Delta_E}^{(i)} & \overline{K}_{I\Pi_E}^{(i)} \\ K_{\overline{\Gamma}I}^{(i)} & K_{\overline{\Gamma}\overline{\Gamma}}^{(i)} & \overline{K}_{\overline{\Gamma}\Delta_E}^{(i)} & \overline{K}_{\overline{\Gamma}\Pi_E}^{(i)} \\ \hline \overline{K}_{\Delta_E I}^{(i)} & \overline{K}_{\Delta_E\overline{\Gamma}}^{(i)} & \overline{K}_{\Delta_E\Delta_E}^{(i)} & \overline{K}_{\Delta_E\Pi_E}^{(i)} \\ \overline{K}_{\Pi_E I}^{(i)} & \overline{K}_{\Pi_E\overline{\Gamma}}^{(i)} & \overline{K}_{\Pi_E\Delta_E}^{(i)} & \overline{K}_{\Pi_E\Pi_E}^{(i)} \end{array}\right].$$

Denoting the primal vertices by a subscript Π_V and the remaining dual displacement variables by a subscript Δ, we can then write $\mathbf{u}_{\overline{\Gamma}}^{(i)} = [\mathbf{u}_{\Delta}^{(i)T}\ \mathbf{u}_{\Pi_V}^{(i)T}]^T$. Using this splitting for the local stiffness matrices $K^{(i)}$ accordingly, ordering the primal variables $\mathbf{u}_{\Pi_V}^{(i)}$ and $\mathbf{u}_{\Pi_E}^{(i)}$ last, and combining them as primal variables $\mathbf{u}_{\Pi}^{(i)} = [\mathbf{u}_{\Pi_V}^{(i)T}, \mathbf{u}_{\Pi_E}^{(i)T}]^T$, we obtain

$$T^{(i)T}K^{(i)}T^{(i)} = \left[\begin{array}{ccc} K_{II}^{(i)} & \overline{K}_{\Delta I}^{(i)T} & \overline{K}_{\Pi I}^{(i)T} \\ \overline{K}_{\Delta I}^{(i)} & \overline{K}_{\Delta\Delta}^{(i)} & \overline{K}_{\Pi\Delta}^{(i)T} \\ \overline{K}_{\Pi I}^{(i)} & \overline{K}_{\Pi\Delta}^{(i)} & \overline{K}_{\Pi\Pi}^{(i)} \end{array}\right].$$

Assembling the primal contributions of each transformed $K^{(i)}$ and ordering the primal variables last, we obtain

$$\widetilde{K} := \left[\begin{array}{ccccccc} K_{II}^{(1)} & \overline{K}_{I\Delta}^{(1)} & & & & & \widetilde{K}_{\Pi I}^{(1)T} \\ \overline{K}_{\Delta I}^{(1)} & \overline{K}_{\Delta\Delta}^{(1)} & & & & & \widetilde{K}_{\Pi\Delta}^{(1)T} \\ & & \ddots & & & & \vdots \\ & & & K_{II}^{(N)} & \overline{K}_{I\Delta}^{(N)} & & \widetilde{K}_{\Pi I}^{(N)T} \\ & & & \overline{K}_{\Delta I}^{(N)} & \overline{K}_{\Delta\Delta}^{(N)} & & \widetilde{K}_{\Pi\Delta}^{(N)T} \\ \widetilde{K}_{\Pi I}^{(1)} & \widetilde{K}_{\Pi\Delta}^{(1)} & \cdots & \widetilde{K}_{\Pi I}^{(N)} & \widetilde{K}_{\Pi\Delta}^{(N)} & & \widetilde{K}_{\Pi\Pi} \end{array}\right] =: \left[\begin{array}{cc} \overline{K}_{BB} & \widetilde{K}_{\Pi B}^T \\ \widetilde{K}_{\Pi B} & \widetilde{K}_{\Pi\Pi} \end{array}\right].$$

The transformation of basis changes the sparsity pattern of the transformed matrices $T^{(i)T}K^{(i)}T^{(i)}$ compared to that of the original local stiffness matrices $K^{(i)}$, but only the matrix blocks related to the edge degrees of freedom are affected; cf. (4.2). We note that the set of edge degrees of freedom is only a small subset of the overall set of interface degrees of freedom. Thus, the transformation of basis only slightly affects the sparsity pattern. Using local Lagrange multipliers, the sparsity of the transformed matrices can be further improved; cf. [25, section 6.2]. To illustrate the change of sparsity of our local, transformed matrices $T^{(i)T}K^{(i)}T^{(i)}$ using the transformation (4.1), we compared the increase of nonzero entries in $T^{(i)T}K^{(i)}T^{(i)}$ to that of $K^{(i)}$ for the problems in Tables 5.1 and 5.2 in section 5. We found an increase between 4 and 12 percent of nonzero entries compared to the original local matrices $K^{(i)}$.

In our FETI-DP algorithm described in section 3, we always assume that we have performed an appropriate change of basis. If there is no danger of confusion, we will drop the overline notation which indicates the dual displacement variables in the transformed basis.

Using the transformation of basis, we again obtain

$$\left[\begin{array}{ccc} K_{BB} & \widetilde{K}_{\Pi B}^T & B^T \\ \widetilde{K}_{\Pi B} & \widetilde{K}_{\Pi\Pi} & O \\ B & O & O \end{array}\right] \left[\begin{array}{c} \mathbf{u}_B \\ \tilde{\mathbf{u}}_{\Pi} \\ \boldsymbol{\lambda} \end{array}\right] = \left[\begin{array}{c} \mathbf{f}_B \\ \tilde{\mathbf{f}}_{\Pi} \\ \mathbf{0} \end{array}\right].$$

We note that after the change of basis has been carried out, we can always use the same implementation as for Algorithm A since the algorithmic description in section 3 does not depend on a specific choice of primal and dual variables. Note that the local problems as well as the Schur complement $\tilde{S}_{\Pi\Pi}$ remain symmetric positive definite.

The following condition number estimate for FETI-DP algorithms using edge averages on selected edges and/or selected vertices as primal constraints can be deduced from Klawonn and Widlund [25, Theorem 1].

THEOREM 4.1.

$$\kappa(M^{-1}F) \leq C\,(1+\log(H/h))^2,$$

H h C h H

For homogeneous materials or problems with discontinuities of the material coefficients which are not very large, it follows from the results in Klawonn and Widlund [25] that primal vertices are not needed to obtain a good condition number bound. We thus introduce a new notation and denote the algorithm using edge averages on selected primal edges as the only primal constraints by Algorithm D_E.

We note that in [25] a variant of Algorithm D_E is considered which in addition uses first order moments on selected edges and possibly selected primal vertices. In this case, a condition number estimate is shown which is also independent of discontinuities of the material coefficients across the interface.

5. Numerical results. We apply our new implementation of FETI-DP using a basis transformation to different problems. First, we consider the standard benchmark problem of an elastic cube divided into smaller regular cubes as subdomains; as a second example we consider three different mechanical parts from an industrial application; and finally, we apply our algorithm to a cancellous bone geometry; see subsections 5.1, 5.2, and 5.3. The domains considered in the second and third subsections are decomposed in irregularly shaped substructures by using ParMetis [20]. In all problems, we use linear tetrahedral finite elements and, for simplicity, a Young modulus of $E = 210$ and a Poisson ratio of $\nu = 0.29$ throughout. In all of our experiments we use Algorithm D_E, i.e., our choice of primal constraints is given by edge averages on selected edges, without any primal vertices. Here, we always constrain all three averages on a primal edge. In all of our computations, we make all edges obtained by Definition 2.1 primal; additionally, in subsections 5.2 and 5.3, we add some edges obtained by Definition 2.2. All computations of subsections 5.1 and 5.2 were carried out on , a 350-node computing cluster operated by the Mathematics and Computer Science Division at Argonne National Laboratory. The cluster consists of 2.4 GHz Xeon processors with 1 or 2 GBytes of memory each and uses a Myrinet connection. The numerical results given in Table 5.1, Figure 5.6, and in subsection 5.3 have been carried out on a 16-processor computing cluster in Essen with eight dual 2.2 GHz Opteron nodes and 4 GBytes of memory for each processor. This cluster uses a Gigabit ethernet connection.

5.1. Model problem. In this section, as a benchmark model problem, we consider a homogeneous, isotropic, linearly elastic cube which is clamped at one side, while all other parts of the boundary have homogeneous natural boundary conditions. A volume force is applied which defines the right-hand side.

In order to analyze the numerical and parallel scalability of our FETI-DP algorithm we report on two different series of experiments. In our first set of runs, we

TABLE 5.1
Results for a cube divided into smaller cubes with a fixed subdomain size of 8 232 *degrees of freedom (d.o.f.) using PETSc as subdomain and coarse grid sparse direct solver.*

Proc.	N	$1/h+1$	d.o.f.	Coarse	$\lambda_{\min}$	$\lambda_{\max}$	Iter	Time
1	8	27	59 049	18	1.03	12.94	18	137s
8	64	53	446 631	324	1.03	10.45	23	171s
27	216	79	1 479 117	1 350	1.04	10.32	23	188s
64	512	105	3 472 875	3 528	1.04	10.31	23	192s
125	1 000	131	6 744 273	7 290	1.04	10.30	23	226s

TABLE 5.2
Results for a cube divided into smaller cubes with a fixed subdomain size of 8 232 *degrees of freedom (d.o.f.) using MUMPS as subdomain and coarse grid sparse direct solver.*

Proc.	N	$1/h+1$	d.o.f.	Coarse	$\lambda_{\min}$	$\lambda_{\max}$	Iter	Time
4	64	53	446 631	324	1.03	10.45	23	60.3s
32	512	105	3 472 875	3 528	1.04	10.31	23	69.2s
108	1 728	157	11 609 679	13 068	1.04	10.30	23	78.8s

keep the dimension of the local problems, and H/h, fixed and increase the number N of subdomains and thus the overall problem size; see Tables 5.1 and 5.2. In a second series of experiments, we keep a fixed number N of subdomains and increase the size of the local problems, and H/h, resulting in a smaller h and thus a larger overall problem size. The results of these experiments are given in Table 5.3 and Figure 5.1. In all sets of experiments, we use as a stopping criterion the relative reduction of the preconditioned dual residual by 10^{-7}.

In the numerical results obtained for a fixed subdomain size, reported in Tables 5.1 and 5.2, the cube is partitioned into smaller cubes with $H/h = 13$ which results in 8 232 degrees of freedom for each subdomain. In Tables 5.1 and 5.2, we denote the degrees of freedom of the original, assembled problem by "d.o.f." and those of the coarse problem by "Coarse". We first present results using the sparse direct solver built into PETSc [3, 2, 4] for solving the coarse problem and the local problems; cf. Table 5.1. Next, we present a set of experiments using the sparse direct solver package MUMPS 4.3.2 [1] together with the optimized BLAS libraries for the Xeon architecture to solve the coarse problem and the local problems; cf. Table 5.2. Both sets of experiments show that our FETI-DP algorithm using only edge averages as primal constraints yields a numerical and parallel scalable domain decomposition method. As a further result of this comparison, we see that using MUMPS as a direct solver accelerates our method by almost a factor of three in terms of CPU time.

The numerical results for a fixed number of subdomains with increasing size are obtained for the unit cube divided into $N = 4 \times 4 \times 4 = 64$ subdomains. We keep $H = 1/4$ fixed and vary H/h between 3 and 23. Algorithm D_E has a coarse problem size of 324. The growth of the number of iterations and of the largest eigenvalue of Algorithms D_E are shown in Figure 5.1.

All of the numerical results in this subsection confirm the theoretical condition number estimate given in Theorem 4.1; cf. also [25, Theorem 1].

TABLE 5.3
Results for $4 \times 4 \times 4$ subdomains of increasing size. The iteration count is given for a relative reduction of the preconditioned dual residual by 10^{-7}.

d.o.f.	$H/h+1$	Iter	$\lambda_{\min}$	$\lambda_{\max}$
6 591	4	14	1.03	**4.11**
27 783	6	17	1.03	**5.70**
73 167	8	18	1.03	**7.10**
273 375	12	22	1.03	**9.45**
680 943	16	24	1.03	**11.36**
1 369 599	20	25	1.04	**12.97**
2 413 071	24	26	1.04	**14.35**

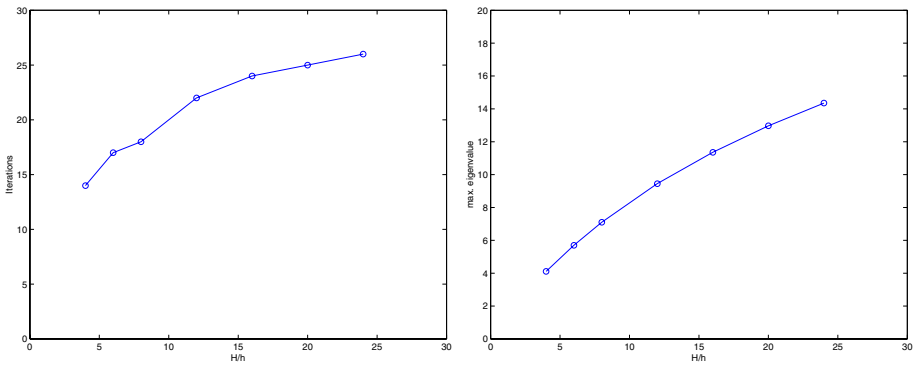


FIG. 5.1. *Growth of number of iterations (left) and λ_{max} (right) of Algorithm D_E for the unit cube with variable H/h and $4 \times 4 \times 4$ subdomains.*

5.2. Industrial applications. In this section, we apply our FETI-DP algorithm to three different industrial finite element problems, denoted by mechanical parts A, B, and C.

The direct subdomain solves as well as the coarse problem solve are performed by again using the sparse direct solver MUMPS and the optimized BLAS libraries for the Xeon architecture. To compare the parallel performance of our FETI-DP domain decomposition method Algorithm D_E to that of the parallel sparse direct solver provided by MUMPS, we also provide CPU timings for MUMPS applied to the assembled and undecomposed problem on the same machine.

The first problem, mechanical part A (cf. Figure 5.2) has been discretized by 208 536 linear tetrahedral finite elements yielding a global number of 187 539 degrees of freedom. In the reported experiments the mechanical part A is partitioned into $N = 16$ subdomains using ParMetis. The FETI-DP algorithm using only edge averages as primal constraints needed 29 iterations for a relative reduction of the preconditioned dual residual by 10^{-7}. The size of the coarse problem is 393 degrees of freedom. The smallest and largest eigenvalues are $\lambda_{\min} = 1.02$ and $\lambda_{\max} = 19.63$, respectively. The parallel scalability results on 2 to 16 processors are given in Table 5.4.

The second problem in this subsection, mechanical part B (cf. Figure 5.3) is discretized by 581 394 linear tetrahedral finite elements resulting in a global number of 380 709 degrees of freedom. It is partitioned into $N = 64$ subdomains using ParMetis. In all of the experiments reported in Table 5.5, our FETI-DP algorithm using only edge averages as primal constraints needed 29 iterations for a relative reduction of the preconditioned dual residual by 10^{-7}. The size of the coarse problem is 1 020. The

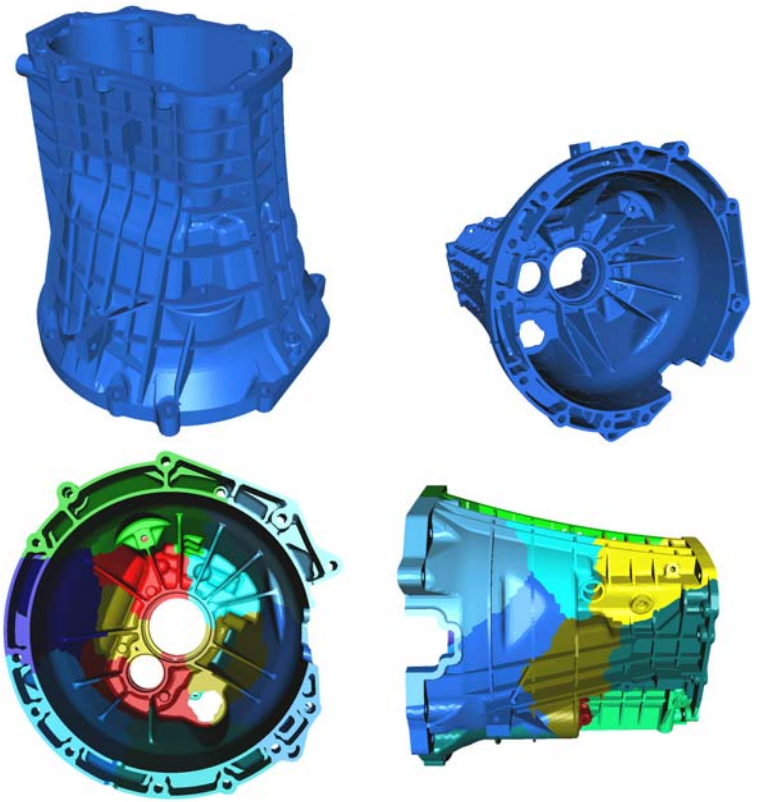


FIG. 5.2. *Mechanical part A; courtesy of GETRAG FORD Transmissions GmbH, Cologne, Germany.*

TABLE 5.4
Parallel scalability results for mechanical part A: CPU times for FETI-DP using MUMPS as local and coarse sparse direct solver and for MUMPS applied to the undecomposed problem.

Proc.	2	4	8	16
FETI-DP D_E/MUMPS	62s	33s	**20s**	**11s**
MUMPS	**47s**	**27s**	26s	19s

smallest and largest eigenvalues are $\lambda_{\min} = 1.03$ and $\lambda_{\max} = 33.85$, respectively. The parallel scalability results on 4 to 64 processors are given in Table 5.5.

The third problem in this subsection, mechanical part C (cf. Figure 5.4) is discretized by 1 291 933 linear tetrahedral finite elements yielding a global number of 841 836 degrees of freedom. It is partitioned into $N = 64$ subdomains using ParMetis. In all of the experiments reported in Table 5.6, the number of iterations is 32 for a relative reduction of the preconditioned dual residual by 10^{-7}. The size of the coarse problem is 957 degrees of freedom. The smallest and largest eigenvalues are $\lambda_{\min} = 1.03$ and $\lambda_{\max} = 27.77$, respectively. The parallel scalability results on 16 to 64 processors are given in Table 5.6.

In Table 5.7 we also report results for mechanical part C decomposed into 512 subdomains. In these experiments the largest eigenvalue is $\lambda_{\max} = 21.36$ and the smallest eigenvalue $\lambda_{\min} = 1.03$. The coarse problem size was chosen as 7 500 degrees

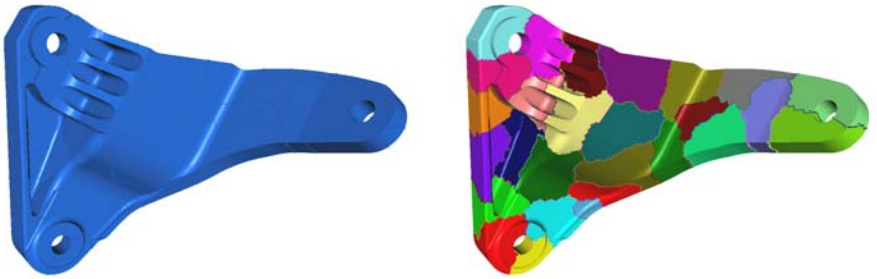


FIG. 5.3. *Mechanical part B; courtesy of GETRAG FORD Transmissions GmbH, Cologne, Germany.*

TABLE 5.5
Parallel scalability results for mechanical part B: CPU times for FETI-DP using MUMPS as local and coarse sparse direct solver and for MUMPS applied to the undecomposed problem.

Proc.	4	8	16	32	64
FETI-DP D_E/MUMPS	**60s**	**33s**	**18s**	**11s**	**6s**
MUMPS	113s	156s	103s	86s	90s

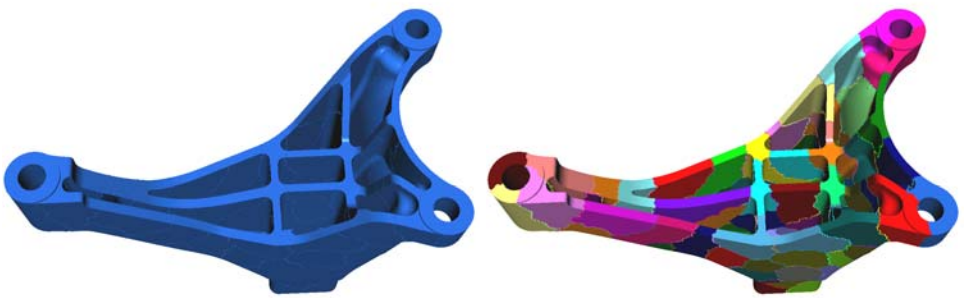


FIG. 5.4. *Mechanical part C; courtesy of GETRAG FORD Transmissions GmbH, Cologne, Germany.*

TABLE 5.6
Parallel scalability results for mechanical part C: CPU times for FETI-DP with 64 *subdomains using MUMPS as local and coarse sparse direct solver and for MUMPS applied to the undecomposed problem.*

Proc.	16	32	64
FETI-DP D_E/MUMPS	**49s**	**26s**	**17s**
MUMPS	failed	1 177s	889s

TABLE 5.7
Parallel scalability results for mechanical part C: CPU times for FETI-DP with 512 *subdomains using MUMPS as local and coarse sparse direct solver.*

Proc.	8	16	32	64
Time	58s	29s	15.4s	9.3s

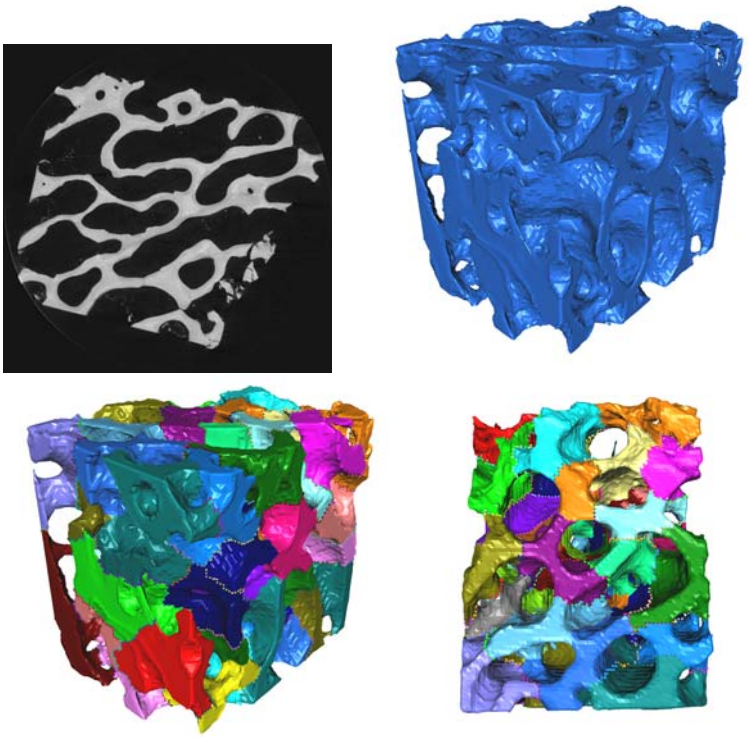


FIG. 5.5. *Cancellous bone. Upper left: cross section of X-ray Computer Tomography. Upper right: finite element discretization with* 907 609 *tetrahedrons and* 620 730 *degrees of freedom. Lower left and right: different views of a decomposition of the bone into subdomains by ParMetis.*

of freedom. The number of iterations is 32 for a relative reduction of the preconditioned dual residual of 10^{-7}.

From our numerical experiments we see that the FETI-DP algorithms using edge averages only as primal constraints yield a parallel scalable domain decomposition method also for problems from industrial applications using irregularly shaped substructures. The FETI-DP algorithm is, in terms of CPU time, always faster than the sparse direct solver applied to the undecomposed problem, except for the smallest problem, mechanical part A, and there only for 2 and 4 processors.

5.3. A cancellous bone geometry. In this subsection, we present results of our FETI-DP algorithm applied to a linearly elastic domain which has a cancellous bone geometry. We show that our method is robust also for complicated geometries with many holes and thin structures, as usually appear in cancellous/trabecular bone; see Figure 5.5. For simplicity, we consider only isotropic linear elasticity. For a full study of cancellous bone, we should use linearly elastic orthotropic or even nonlinear material models; this will be considered in forthcoming work. The bone geometry is discretized by 907 609 linear tetrahedral finite elements resulting in a global number of 620 730 degrees of freedom. The mesh is obtained using a marching cubes algorithm to generate a surface mesh and Netgen [40] to produce a volume mesh from it. It is partitioned into $N = 96$ subdomains using ParMetis. In all of our experiments reported in Table 5.8, the number of iterations is 51 for a relative reduction of the

TABLE 5.8
Parallel scalability of FETI-DP for cancellous bone geometry.

Proc.	1	2	4	8	16
Time	524s	312s	170s	90s	46s

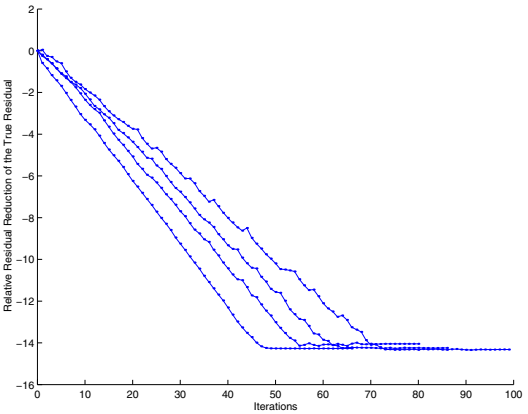


FIG. 5.6. *History of relative residual reductions of the true residual. From left to right: cube benchmark problem (105^3), mechanical part A, mechanical part B, cancellous bone geometry.*

preconditioned dual residual by 10^{-10}. The size of the coarse problem is 1 602 degrees of freedom. The smallest and largest eigenvalues are $\lambda_{\min} = 1.02$ and $\lambda_{\max} = 43.57$, respectively. The parallel scalability results on 1 to 16 processors are given in Table 5.8. As sparse direct solvers for the coarse and the local problems, we use UMFPACK 4.3 [8].

In Figure 5.6 we show the convergence history for the cancellous bone geometry as well as for mechanical parts A and B and for the cube benchmark problem with $H/h = 14$ and $h = 1/105$. We compute the true relative residual $\|F\boldsymbol{\lambda}^n - \mathbf{d}\|_2/\|F\boldsymbol{\lambda}^0 - \mathbf{d}\|_2$ explicitly in each step. Let us note that the true residual is only used in the experiments for the results shown in Figure 5.6. Here, we use the true residual for the computation of the results to avoid possible inaccuracies in the recursive computation used in the conjugate gradient algorithm when the residual is approaching machine precision.

Acknowledgments. The authors gratefully acknowledge the use of , a 350-node computing cluster operated by the Mathematics and Computer Science Division at Argonne National Laboratory as part of its Laboratory Computing Resource Center. The authors also gratefully acknowledge the X-ray computer tomography cross sections which were obtained from Matthias Epple, Institute of Inorganic Chemistry, University of Duisburg-Essen, within a common research collaboration. The second author acknowledges many fruitful discussions with Barry Smith and Matthew Knepley while visiting the Mathematics and Computer Science Division at Argonne National Laboratory. The authors wish to thank Hans-Jürgen Philipsenburg, Holger Hein, and GETRAG FORD Transmissions GmbH, Cologne, Germany. For discussions on the revised version, the authors are grateful to Kendall Pierson, Sandia National Laboratories. Last but not least, the authors have had many interesting discussions with Olof Widlund, Courant Institute of Mathematical Sciences.

REFERENCES

[1] P. R. Amestoy, I. S. Duff, J.-Y. L'Excellent, and J. Koster, *A fully asynchronous multifrontal solver using distributed dynamic scheduling*, SIAM J. Matrix Anal. Appl., 23 (2001), pp. 15–41.

[2] S. Balay, K. Buschelman, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, L. Curfman McInnes, B. F. Smith, and H. Zhang, *PETSc Users Manual*, Technical report ANL-95/11, Revision 2.1.5, Argonne National Laboratory, Argonne, IL, 2004.

[3] S. Balay, K. Buschelman, W. D. Gropp, D. Kaushik, M. Knepley, L. Curfman McInnes, B. F. Smith, and H. Zhang, *PETSc Home Page*, http://www.mcs.anl.gov/petsc, 2001.

[4] S. Balay, V. Eijkhout, W. D. Gropp, L. Curfman McInnes, and B. F. Smith, *Efficient management of parallelism in object oriented numerical software libraries*, in Modern Software Tools in Scientific Computing, E. Arge, A. M. Bruaset, and H. P. Langtangen, eds., Birkhäuser, Boston, 1997, pp. 163–202.

[5] M. Bhardwaj, D. Day, C. Farhat, M. Lesoinne, K. Pierson, and D. Rixen, *Application of the FETI method to ASCI problems: Scalability results on one thousand processors and discussion of highly heterogeneous problems*, Internat. J. Numer. Methods Engrg., 47 (2000), pp. 513–535.

[6] P. G. Ciarlet, *Mathematical Elasticity Volume* I: *Three-Dimensional Elasticity*, North–Holland, Amsterdam, 1988.

[7] J.-M. Cros, *A preconditioner for the Schur complement domain decomposition method*, in Domain Decomposition Methods in Science and Engineering, O. Widlund, I. Herrera, D. Keyes and R. Yates, eds., National Autonomous University of Mexico (UNAM), Mexico City, Mexico, 2003, p. 373–380.

[8] T. A. Davis, *A column pre-ordering strategy for the unsymmetric-pattern multifrontal method*, ACM Trans. Math. Software, 30 (2004), pp. 165–195.

[9] C. Dohrmann, *An Approximate BDDC Preconditioner*, Technical report SAND 2005-5424, Sandia National Laboratories, Albuquerque, NM, 2005.

[10] C. R. Dohrmann, *A preconditioner for substructuring based on constrained energy minimization*, SIAM J. Sci. Comput., 25 (2003), pp. 246–258.

[11] M. Dryja, M. V. Sarkis, and O. B. Widlund, *Multilevel Schwarz methods for elliptic problems with discontinuous coefficients in three dimensions*, Numer. Math., 72 (1996), pp. 313–348.

[12] M. Dryja, B. F. Smith, and O. B. Widlund, *Schwarz analysis of iterative substructuring algorithms for elliptic problems in three dimensions*, SIAM J. Numer. Anal., 31 (1994), pp. 1662–1694.

[13] C. Farhat, P.-S. Chen, F. Risler, and F.-X. Roux, *A unified framework for accelerating the convergence of iterative substructuring methods with Lagrange multipliers*, Internat. J. Numer. Methods Engrg., 42 (1998), pp. 257–288.

[14] C. Farhat, M. Lesoinne, P. LeTallec, K. Pierson, and D. Rixen, *FETI-DP: A dual-primal unified FETI method. Part* I: *A faster alternative to the two-level FETI method*, Internat. J. Numer. Methods Engrg., 50 (2001), pp. 1523–1544.

[15] C. Farhat, M. Lesoinne, and K. Pierson, *A scalable dual-primal domain decomposition method*, Numer. Linear Algebra Appl., 7 (2000), pp. 687–714.

[16] C. Farhat, J. Mandel, and F.-X. Roux, *Optimal convergence properties of the FETI domain decomposition method*, Comput. Methods Appl. Mech. Engrg., 115 (1994), pp. 367–388.

[17] C. Farhat and F.-X. Roux, *Implicit parallel processing in structural mechanics*, Comput. Mech. Adv., 2 (1994), pp. 1–124.

[18] C. Farhat and F.-X. Roux, *A method of finite element tearing and interconnecting and its parallel solution algorithm*, Internat. J. Numer. Methods Engrg., 32 (1991), pp. 1205–1227.

[19] Y. Fragakis and M. Papadrakakis, *The mosaic of high performance domain decomposition methods for structural mechanics: Formulation, interrelation and numerical efficiency of primal and dual methods*, Comput. Methods Appl. Mech. Engrg, 192 (2003), pp. 3799–3830.

[20] G. Karypis, K. Schloegel, and V. Kumar, *ParMetis—Parallel Graph Partitioning and Sparse Matrix Ordering*, Version 3.1, Technical report, Department of Computer Science and Engineering, University of Minnesota, 2003.

[21] A. Klawonn and O. Rheinbach, *Robust FETI-DP Methods for Heterogeneous Three Dimensional Linear Elasticity Problems*, Technical report TR-SM-E-607, Department of Mathematics, University of Duisburg-Essen, Germany, 2005, revised 2006. Comput. Methods Appl. Mech. Engrg., to appear.

[22] A. Klawonn and O. Rheinbach, *Inexact FETI-DP Methods*, Technical report TR-SM-E-609, Department of Mathematics, University of Duisburg-Essen, Germany, 2005, revised 2006. Internat. J. Numer. Methods Engrg., to appear.

[23] A. Klawonn and O. B. Widlund, *Dual and dual-primal FETI methods for elliptic problems with discontinuous coefficients in three dimensions*, in Domain Decomposition Methods, Proceedings of the 12th International Conference on Domain Decomposition Methods, Chiba, Japan, 1999, DDM.org, Augsburg, 2001, pp. 29–39.

[24] A. Klawonn and O. B. Widlund, *FETI and Neumann–Neumann iterative substructuring methods: Connections and new results*, Comm. Pure Appl. Math., 54 (2001), pp. 57–90.

[25] A. Klawonn and O. B. Widlund, *Dual-primal FETI methods for linear elasticity*, Comm. Pure Appl. Math., 59 (2006), pp. 1523–1572.

[26] A. Klawonn and O. B. Widlund, *Selecting constraints in Dual-Primal FETI methods for elasticity in three dimensions*, in Domain Decomposition Methods in Science and Engineering, R. Kornhuber, R. H. W. Hoppe, D. E. Keyes, J. Périaux, O. Pironneau, and J. Xu, eds., Lecture Notes in Comput. Sci. Engrg., Springer-Verlag, Berlin, 2005, pp. 67–81.

[27] A. Klawonn, O. B. Widlund, and M. Dryja, *Dual-Primal FETI methods for three-dimensional elliptic problems with heterogeneous coefficients*, SIAM J. Numer. Anal., 40 (2002), pp. 159–179.

[28] A. Klawonn, O. B. Widlund, and M. Dryja, *Dual-Primal FETI methods with face constraints*, in Recent Developments in Domain Decomposition Methods, L. F. Pavarino and A. Toselli, eds., Lecture Notes in Comput. Sci. Engrg. 23, Springer-Verlag, Berlin, 2002, pp. 27–40.

[29] M. Lesoinne, *A FETI-DP corner selection algorithm for three-dimensional problems*, in Domain Decomposition Methods in Science and Engineering, I. Herrera, D. E. Keyes, O. B. Widlund, and R. Yates, eds., Proceedings of the 14th International Conference on Domain Decomposition Methods, Cocoyoc in Morelos, Mexico, 2003, pp. 217–223.

[30] J. Li and O. B. Widlund, *On the use of inexact subdomain solvers for BDDC algorithms*, Technical report TR2005-871, Department of Computer Science, Courant Institute of Mathematical Sciences, New York University, New York, 2005. Comput. Methods Appl. Mech. Engrg., to appear.

[31] J. Li and O. B. Widlund, *FETI-DP, BDDC, and block Cholesky methods*, Internat. J. Numer. Methods Engrg., 66 (2006), pp. 250–271.

[32] J. Mandel and C. R. Dohrmann, *Convergence of a balancing domain decomposition by constraints and energy minimization*, Numer. Linear Algebra Appl., 10 (2003), pp. 639–659.

[33] J. Mandel, C. R. Dohrmann, and R. Tezaur, *An algebraic theory for primal and dual substructuring methods by constraints*, Appl. Numer. Math., 54 (2005), pp. 167–193.

[34] J. Mandel and R. Tezaur, *Convergence of a substructuring method with Lagrange multipliers*, Numer. Math., 73 (1996), pp. 473–487.

[35] J. Mandel and R. Tezaur, *On the convergence of a dual-primal substructuring method*, Numer. Math., 88 (2001), pp. 543–558.

[36] K. H. Pierson, *Private communication*, 2005.

[37] K. H. Pierson, *A Family of Domain Decomposition Methods for the Massively Parallel Solution of Computational Mechanics Problems*, Ph.D. thesis, Aerospace Engineering, University of Colorado at Boulder, 2000.

[38] D. Rixen and C. Farhat, *A simple and efficient extension of a class of substructure based preconditioners to heterogeneous structural mechanics problems*, Internat. J. Numer. Methods Engrg., 44 (1999), pp. 489–516.

[39] D. Rixen and C. Farhat, *A simple and efficient extension of a class of substructure based preconditioners to heterogeneous structural mechanics problems*, Internat. J. Numer. Methods Engrg., 44 (1999), pp. 489–516.

[40] J. Schöberl, *Netgen: An advancing front 2d/3d-mesh generator based on abstract rules*, Comput. Visual. Sci., 1 (1997), pp. 41–52.

[41] A. Toselli and O. B. Widlund, *Domain Decomposition Methods: Algorithms and Theory*, Springer Ser. Comput. Math. 34, Springer-Verlag, Berlin, Heidelberg, New York, 2005.

[42] X. Tu, *Three-Level BDDC in Two Dimensions*, Technical report TR2004-856, Department of Computer Science, Courant Institute, New York University, New York, 2004.

[43] X. Tu, *Three-Level BDDC in Three Dimensions*, Technical report TR2005-862, Department of Computer Science, Courant Institute, New York University, New York, 2005.

SIAM J. SCI. COMPUT.
Vol. 28, No. 5, pp. 1907–1926
© 2006 Society for Industrial and Applied Mathematics

PARALLEL GUARANTEED QUALITY DELAUNAY UNIFORM MESH REFINEMENT*

ANDREY N. CHERNIKOV† AND NIKOS P. CHRISOCHOIDES†

Abstract. We present a theoretical framework for developing parallel guaranteed quality Delaunay mesh generation software that allows us to use commercial off-the-shelf sequential Delaunay meshers for two-dimensional geometries. In this paper, we describe our approach for constructing uniform meshes, that is, the meshes in which all elements have approximately the same size. Our uniform distributed- and shared-memory implementations are based on a simple (block) coarse-grained mesh decomposition. Our method requires only local communication, which is bulk and structured as opposed to fine and unpredictable communication of the other existing practical parallel guaranteed quality mesh generation and refinement techniques. Our experimental data show that on a cluster of more than 100 workstations we can generate about 0.9 billion elements in less than 5 minutes in the absence of work-load imbalances. Preliminary results for this paper were presented in [A. N. Chernikov and N. P. Chrisochoides, "Practical and efficient point insertion scheduling method for parallel guaranteed quality Delaunay refinement," in Proceedings of the 18th Annual International Conference on Supercomputing, ACM Press, New York, 2004, pp. 48–57]. Our work in progress includes extending the presented approach, which can efficiently generate only uniform meshes, to nonuniform graded meshes.

Key words. Delaunay triangulation, mesh generation, parallel refinement

AMS subject classifications. 65D18, 68W05, 68W10, 68N19

DOI. 10.1137/050625886

1. Introduction.

With the development of very efficient sequential guaranteed quality mesh generators, e.g., the "Triangle" [33], the construction of very large meshes that cannot fit into the memory of a single machine still remains a challenging problem. These large meshes are required, for instance, to conduct direct numerical simulations of turbulence in cylinder flows with very large Reynolds numbers; see [14]. With the increase of the Reynolds number, the size of the mesh grows in the order of $Re^{9/4}$ (see [23]), which motivates the use of a parallel distributed-memory mesh generation algorithm.

There are four important requirements that are usually imposed on the parallel mesh generation algorithms: , , , and refers to the fact that distributed meshes retain the same quality of the elements and partition properties as the sequentially generated and partitioned meshes; see [10]. is understood as the ability of the algorithm and software to achieve the speedup proportional to the number of processors; see [25]. The can be measured in terms of its computation cost and the appropriateness of the resulting subdomains for the selected mesh generation algorithm. Finally, allows one to take advantage of the ever-evolving basic sequential meshing libraries which are now mature and highly optimized. In this

*Received by the editors March 3, 2005; accepted for publication (in revised form) May 15, 2006; published electronically November 16, 2006. This work was supported by National Science Foundation grants ACI-0085969, EIA-9972853, EIA-0203974, and ACI-0312980 and was performed using computational facilities at the College of William and Mary which were enabled by grants from Sun Microsystems, the National Science Foundation, and Virginia's Commonwealth Technology Research Fund.

http://www.siam.org/journals/sisc/28-5/62588.html

†Department of Computer Science, College of William and Mary, Williamsburg, VA 23185 (ancher@cs.wm.edu, nikos@cs.wm.edu).

paper, we present an algorithm that conforms to all four of these requirements, while the previously published results address only some of them.

The sophisticated domain decomposition methods like the medial axis domain decomposition (MADD) [27] allow one to completely decouple the subproblems, and therefore eliminate the communication, but they are very expensive and difficult to construct for three-dimensional domains. In this paper, we settle for a simpler (block) domain decomposition at the cost of some bulk communication and work-load imbalances. Fortunately, the bulk communication in our method is structured and can be tolerated. To cope with the work-load imbalances, we developed a runtime system [2] that successfully addresses the dynamic load balancing problem for mesh generation.

Nave, Chrisochoides, and Chew [30] presented the first practical provably good parallel mesh refinement algorithm for polyhedral domains. They addressed the stability, and to some success the scalability, but not the code reuse requirement. The algorithm in [30] guarantees stability by simultaneously partitioning and refining the interface surfaces and the volumes of the subdomains—the changes caused by a point insertion can propagate across the processor boundaries—at the cost of high communication and of the restructuring of the Delaunay kernel. In [10], Chrisochoides and Nave restructured the sequential Bowyer–Watson (B-W) Delaunay kernel [5, 39] using the speculative execution model (see [20]) in order to tolerate high communication latencies. The speculative execution model allows setbacks to occur if the changes caused by the concurrent insertion of two or more points are incompatible. Although more than 80% of the overhead due to the remote data gather operations is masked, the experimental data in [10] suggest only $\mathcal{O}(\log P)$ speedup, P being the number of processors. In this paper, we provide a sufficient condition, which (in two dimensions) guarantees the independence of the simultaneously inserted points, eliminates the setbacks and the need to restructure the mesh generation kernel, and does not induce intensive communication. Moreover, our condition is not based on the coloring techniques, which require high preprocessing time, since they need first to construct a graph (cavity graph[1]).

Blelloch et al. [4] describe a divide-and-conquer projection-based algorithm for constructing Delaunay triangulations of predefined point sets in parallel. The work by Kadow [21] and Kadow and Walkington [22] extended [4] for parallel mesh generation and further eliminated the sequential step (contrary to our proposed method) for an initial mesh, but did not address the issue of code reuse.

An idea of updating partition boundaries when inserted points happen to be close to them was presented by Chew, Chrisochoides, and Sukup [8] as a parallel constrained Delaunay meshing (PCDM) algorithm. In PCDM, the edges on the boundaries of submeshes are fixed (constrained), and if a new point encroaches upon a constrained edge, another point is inserted in the middle of this edge instead. As a result, a "split" message is sent to the neighboring processor, notifying that it also has to insert the midpoint of the shared edge. This approach requires the construction of the separators that will not compromise the quality of the final mesh, and permits the reuse of sequential codes only to a limited degree, since one has to handle "split" messages and partial cavity expansions within the B-W kernel.

[1]When a new point is inserted, all tetrahedra that have this point within their circumspheres are eliminated. This creates a *cavity*. In the cavity graph each cavity is represented by a vertex and two adjacent cavities are represented by an edge. Chew et al. [9] presented a method to improve the efficiency of this method, but its computation requirements and possible optimizations are subject to additional research.

The theoretical analysis of the idea of the concurrent insertion of independent points has been started by several authors. Edelsbrunner and Guoy [15] define the prestar of point x as the difference between the closure of the set of tetrahedra whose circumspheres enclose x and the closure of the set of remaining tetrahedra. Further, they define the points x and y as independent if the closures of their prestars are disjoint. We prove a similar condition of point independence. The difference between the independence condition in [15] and in this paper is that our formulation is less restrictive: it allows the prestars (we use the word cavity) to share a point. However, computing the prestars (cavities) and their intersections for all candidate points is very expensive. That is why we do not use coloring methods that are based on the cavity graphs and continue by proving a theorem, which allows us to use only the distance between the points to check whether they are independent. The minimum separation distance argument in [15] is used to derive the upper bound on the number of inserted vertices and prove termination, but does not ensure point independence. In addition, we propose a simple block decomposition scheme for scheduling parallel point insertion for both the distributed- and the shared-memory implementations. In [15], a shared-memory algorithm based on finding the maximal independent set of points is used. Also, Spielman, Teng, and Üngör [36] presented the first theoretical analysis of the complexity of the parallel Delaunay refinement algorithms. However, the assumption is that the global mesh is completely retriangulated each time a set of independent points is inserted [38]. In [37] a more practical algorithm was developed.

Besides the Delaunay-based algorithms, a number of other parallel mesh generation algorithms have been published. De Cougny, Shephard, and Ozturan [12] base the parallel mesh construction on an underlying octree. Löhner and Cebral [28] and Ito et al. [19] developed parallel advancing front schemes. Globisch [17, 18] presented a parallel mesh generator which uses a sequential frontier algorithm. A detailed review of many more methods appears in [11].

The rest of the paper is organized as follows. Section 2 presents the theoretical framework for our parallel guaranteed quality Delaunay refinement algorithm. After introducing the notation, we describe the sequential Delaunay refinement algorithm and emphasize the Delaunay triangulation loop invariant, which has to be maintained in order to ensure the correctness of the result. In subsection 2.3.1, we show how the concurrent point insertion can violate this invariant and lead to either nonconforming or non-Delaunay mesh. We prove a theorem which states that if the distance between two points is greater than four times the upper bound on triangle circumradius, then these points can be safely inserted in parallel. We continue by proving in subsection 2.3.2 that the maximal circumradius is not increased in the course of the execution of the B-W algorithm, which makes our condition applicable at all times during the mesh refinement. Next, we show in subsection 2.3.3 how to select the preprocessing parameters so that we can efficiently use all processors and at the same time have minimal sequential overhead. Our coarse-grained partitioning scheme is described in section 3. Finally, in section 4 we present the implementation issues and the experimental results. We conclude with a summary of our results and future work directions.

2. Parallel Delaunay refinement.

2.1. Notation. We will denote a point with index i as p_i and the triangle with vertices p_i, p_j, and p_k as $\triangle p_i p_j p_k$. When the vertices of a triangle are irrelevant, we will denote such a triangle with a single letter t. We will use the small letters a, b, and c to refer to the side lengths of a triangle, adopting the convention $a \leq b \leq c$,

and the capital letters A, B, and C to represent the measures of the three angles in the triangle, where we suppose the relation $A \leq B \leq C$.

The term [illegible] p [illegible] r refers to the set of points in $\mathbb{R}^2$ whose Euclidean distance to p is exactly r. We will use the term [illegible] for the set of points in $\mathbb{R}^2$ whose distance from p is strictly less than r. The symbol $\|\cdot\|$ will represent the conventional Euclidean distance (the L_2 norm).

Let the [illegible] of triangle $\triangle p_i p_j p_k$ be the unique circle that passes through p_i, p_j, and p_k. The center and the radius of a triangle's circumcircle are called its [illegible] and [illegible], respectively. Let us call the open disk corresponding to a triangle's circumcircle its [illegible]. We will use the symbols $\bigcirc(t)$, $r(t)$, and $\odot(t)$ to represent the circumdisk, circumradius, and the circumcenter of the triangle t, respectively.

The input to a planar triangular mesh generation algorithm includes a description of [illegible] $\Omega \subset \mathbb{R}^2$, which is permitted to contain holes or have more than one connected component. We will use a [illegible] (PSLG) [33] to delimit Ω from the rest of the plane. A PSLG consists of a set P of points and a set S of segments connecting the points in P. The segments are not allowed to intersect in points that are not in P. Each segment in S is considered [illegible] and must appear (possibly as a union of smaller segments) in the final mesh.

DEFINITION 1. [illegible] V [illegible] $\Omega \subset \mathbb{R}^2$ [illegible] T [illegible] V [illegible] $\mathcal{T} = (V, T)$ [illegible] conformal[2] triangulation ([illegible] [16]) [illegible]

1. [illegible] T [illegible] V
2. [illegible] T [illegible] Ω
3. [illegible] ([illegible]) [illegible] T
4. [illegible]

We will use the terms [illegible] and [illegible] interchangeably, depending on the context.

Also, let $\bar{r}$, $\bar{\rho}$, and $\bar{\Delta}$ denote the upper bounds on the circumradius, the circumradius-to-shortest edge ratio, and the area of the triangles, respectively.

2.2. Sequential Delaunay refinement. The applications that use Delaunay meshes usually impose two constraints on the quality of the mesh elements: the upper bound $\bar{\Delta}$ on the element area and the lower bound $\bar{A}$ on the smallest angle. As shown by Miller et al. [29] and Shewchuk [34], in two dimensions the circumradius-to-shortest edge ratio ρ of a triangle can be expressed in terms of its minimal angle A as

$$\rho = \frac{1}{2 \sin A}. \tag{2.1}$$

As the circumradius-to-shortest edge ratio is the metric that is naturally optimized by the Delaunay refinement (see [29]), we will consider $\bar{\Delta}$ and $\bar{\rho}$, along with a PSLG $\mathcal{X}$, as the parameters to the mesh generation algorithm.

Typically, a mesh generation procedure starts by constructing an initial mesh which conforms to the input vertices and segments, and then refines this mesh until the constraints $\bar{\Delta}$ and $\bar{\rho}$ are met. For sufficiently small values of $\bar{\Delta}$, resulting in large fine meshes, most of the time is spent reducing the area of the triangles. Although

[2] A term which sounds similar, but has a different meaning, is *conforming triangulation*, which is used in the context of preserving constrained edges [32].

asymptotically $\mathcal{O}(n \log n)$ time is required to triangulate n points, while refining an existing mesh is on average linear in the number of the triangles produced, for fairly large final meshes the latter time significantly dominates. In this paper, we focus on parallelizing the Delaunay refinement stage, which is usually the most memory- and computation-expensive. The general idea of the Delaunay refinement is to insert points in the circumcenters of the triangles that violate the required bounds, until there are no such triangles left. There are two main variations of the point insertion procedure: Lawson's algorithm [26], which uses edge flips, and the B-W algorithm [5, 39], which is based on deleting the triangles that are no longer Delaunay and inserting new triangles that satisfy the Delaunay property. These algorithms produce equivalent results (see [34]); thus, we will use only the B-W algorithm for our analysis. The following definition, adapted from [16], plays a major role in the study of the B-W algorithm.

DEFINITION 2. [illegible] cavity $\mathcal{C}_{\mathcal{M}}(p)$ of point p with respect to mesh $\mathcal{M}$ [illegible] $\mathcal{M}$ [illegible] p. $\mathcal{C}_{\mathcal{M}}(p) = \{t \in \mathcal{M} \mid p \in \bigcirc(t)\}$

We will write simply $\mathcal{C}(p)$ when $\mathcal{M}$ is clear from the context. Also, we will denote $\mathcal{B}_{\mathcal{M}}(p)$ to be the set of the edges which belong to only one triangle in $\mathcal{C}_{\mathcal{M}}(p)$, i.e., the external edges of $\mathcal{C}_{\mathcal{M}}(p)$.

The sequential Delaunay algorithms treat constrained (boundary) segments differently from triangle edges in the interior of the mesh; see [31, 34]. For a segment s, the unique circle with s as a diameter is called the [illegible] of s. A vertex p is said to [illegible] s if it lies within the open diametral disk of s. When a new point is about to be inserted and it happens to encroach upon a constrained segment s, another point is inserted in the middle of s instead, and the cavity of the segment's midpoint is constructed and triangulated in a similar way.

For a thorough description of the sequential Delaunay refinement algorithms and the proofs of their termination, as well as the proofs of good grading and size optimality of the result, the reader can consult [16, 31, 34, 35] and the references therein. For our analysis, it is sufficient to consider the algorithm in Figure 1. The while-loop of the algorithm maintains the following invariant.

LOOP INVARIANT 1 (Delaunay triangulation). $\mathcal{M}$ [illegible] (i) [illegible] (ii) [illegible]

2.3. Theoretical framework. The parallelization of the sequential Delaunay refinement algorithm can be achieved by inserting the circumcenters of multiple triangles concurrently by several processors. In this subsection, we propose a theoretical framework which allows us to select the candidate circumcenters in such a way that the Delaunay triangulation loop invariant holds throughout the run of the parallel algorithm, and, thus, all the quality guarantees of the sequential algorithm remain valid.

2.3.1. Maintaining the Delaunay triangulation loop invariant. We expect a parallel Delaunay refinement algorithm to maintain Loop Invariant 1. The reason for a parallel algorithm to produce a nonconformal mesh can be a concurrent retriangulation of two or more cavities that have common triangles. Consider, for example, Figure 2(a). However, even if the cavities of two points do not have common triangles, Loop Invariant 1 can still be violated due to the creation of non-Delaunay triangles, i.e., triangles whose circumdisks include mesh vertices. This can happen if two cavities have common edges; see Figure 2(b). The following lemma provides a sufficient condition for the simultaneous retriangulation of two cavities to yield a conformal Delaunay mesh.

DelaunayRefinement($\mathcal{X}$, $\mathcal{M}$, $\bar{\Delta}$, $\bar{\rho}$)
Input: $\mathcal{X}$ is a PSLG which defines Ω
$\mathcal{M}$ is some Delaunay mesh over Ω
$\bar{\Delta}$ and $\bar{\rho}$ are desired upper bounds
Output: a conforming Delaunay mesh respecting $\bar{\Delta}$ and $\bar{\rho}$

```
1  Q ← {t ∈ M | (ρ(t) ≥ ρ̄) ∨ (Δ(t) ≥ Δ̄)}
2  while Q ≠ ∅
3      Let t ∈ Q
4      p ← ⊙(t)
5      Find S, a set of segments encroached upon by p
6      if S ≠ ∅
7          SplitSegments(S)
8      else
9          M ← M \ C_M(p) ∪ {△puv | (u, v) ∈ B_M(p)}
10     endif
11     Update Q
12 endwhile
13 return M
```

FIG. 1. *The sequential Delaunay refinement algorithm. A more detailed description along with the segment splitting strategies, which always guarantee termination, can be found in* [35].

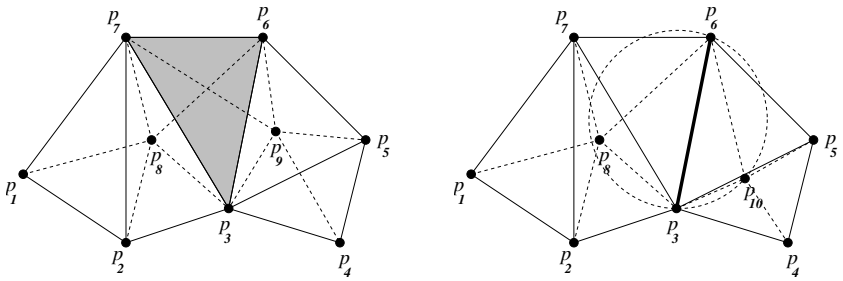


FIG. 2. (a) *If* $\triangle p_3p_6p_7 \in \mathcal{C}(p_8) \cap \mathcal{C}(p_9)$, *where* $\mathcal{C}(p_8) = \{\triangle p_1p_2p_7, \triangle p_2p_3p_7, \triangle p_3p_6p_7\}$ *and* $\mathcal{C}(p_9) = \{\triangle p_3p_6p_7, \triangle p_3p_5p_6, \triangle p_3p_4p_5\}$, *then the concurrent insertion of* p_8 *and* p_9 *results in a nonconformal mesh. The solid lines represent the edges of the initial triangulation, and the dashed lines represent the edges created by the insertion of* p_8 *and* p_9. (b) *If the edge* p_3p_6 *is shared by* $\mathcal{C}(p_8) = \{\triangle p_1p_2p_7, \triangle p_2p_3p_7, \triangle p_3p_6p_7\}$ *and* $\mathcal{C}(p_{10}) = \{\triangle p_3p_5p_6, \triangle p_3p_4p_5\}$, *then the new triangle* $\triangle p_3p_{10}p_6$ *can have point* p_8 *inside its circumdisk, thus violating the Delaunay property.*

LEMMA 1. $\mathcal{C}(p_i)$ $\mathcal{C}(p_j)$ p_i p_j . Since $\mathcal{C}(p_i)$ and $\mathcal{C}(p_j)$ do not intersect, the processors that are working independently on retriangulating $\mathcal{C}(p_i)$ and $\mathcal{C}(p_j)$ will not attempt to simultaneously delete the same triangles or create overlapping triangles, and thus the resulting mesh will be conformal.

To show that the mesh will also satisfy the Delaunay criterion, we need to recall the Delaunay lemma (see [13, 16]), which states that if the empty circumdisk criterion holds for every pair of adjacent triangles, the triangulation is globally Delaunay. The condition that $\mathcal{C}(p_i)$ and $\mathcal{C}(p_j)$ do not share any triangle edges implies that every

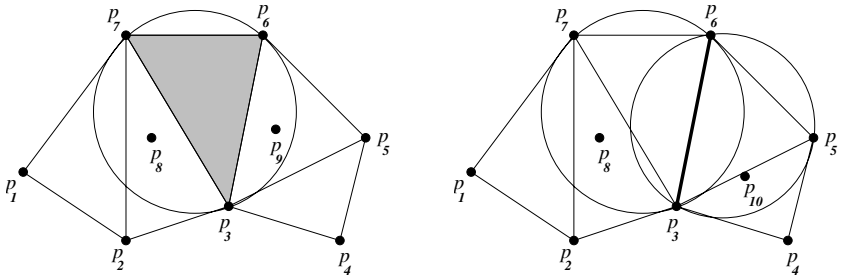


FIG. 3. (a) *If* $\triangle p_3p_6p_7 \in \mathcal{C}(p_8) \cap \mathcal{C}(p_9)$, *then* $\|p_8 - p_9\| < 2r(\triangle p_3p_6p_7) < 2\bar{r}$. (b) *If edge* p_3p_6 *is shared by* $\mathcal{C}(p_8) = \{\triangle p_1p_2p_7, \triangle p_2p_3p_7, \triangle p_3p_6p_7\}$ *and* $\mathcal{C}(p_{10}) = \{\triangle p_3p_5p_6, \triangle p_3p_4p_5\}$, *then* $\|p_8 - p_{10}\| < 2r(\triangle p_3p_6p_7) + 2r(\triangle p_3p_5p_6) < 4\bar{r}$.

external edge of $\mathcal{C}(p_i)$ and $\mathcal{C}(p_j)$ is incident upon some triangle which belongs neither to $\mathcal{C}(p_i)$ nor to $\mathcal{C}(p_j)$. These external triangles together with the ones created by the retriangulation of $\mathcal{C}(p_i)$ and $\mathcal{C}(p_j)$ will locally satisfy the Delaunay property; see [34]. Hence, the mesh will be globally Delaunay. ∎

In practice, though, this lemma may not be very useful, since it requires the construction and the comparison of the cavities for all candidate points. The next lemma achieves a more practical result.

LEMMA 2. $\bar{r}$ $\mathcal{M}$ $p_i, p_j \in \Omega$ Ω $\|p_i - p_j\| \geq 4\bar{r}$ $\mathcal{C}(p_i)$ $\mathcal{C}(p_j)$

We will prove this lemma by contradiction by considering two cases:

(i) Suppose $\|p_i - p_j\| \geq 4\bar{r}$ and $\mathcal{C}(p_i) \cap \mathcal{C}(p_j) \neq \emptyset$. Let $t \in \mathcal{C}(p_i) \cap \mathcal{C}(p_j)$; see Figure 3(a). By the definition of the cavity, $p_i \in \bigcirc(t)$ and $p_j \in \bigcirc(t)$, and hence $\|p_i - p_j\| < 2r(\triangle p_kp_lp_m) < 2\bar{r} < 4\bar{r}$, which contradicts our assumption.

(ii) Suppose $\|p_i - p_j\| \geq 4\bar{r}$ and there exists an edge p_kp_l which is shared by $\mathcal{C}(p_i)$ and $\mathcal{C}(p_j)$; see Figure 3(b). Let $\triangle p_kp_lp_m \in \mathcal{C}(p_i)$ and $\triangle p_kp_lp_n \in \mathcal{C}(p_j)$. Obviously, $\|p_k - p_i\| < 2r(\triangle p_kp_lp_m)$ and $\|p_k - p_j\| < 2r(\triangle p_kp_lp_n)$. From the triangle inequality it follows that $\|p_i - p_j\| < 2r(\triangle p_kp_lp_m) + 2r(\triangle p_kp_lp_n) < 4\bar{r}$, which again contradicts the assumption. ∎

Theorem 1 is the immediate consequence of Lemmas 1 and 2 and can be used for an efficient constant-time verification of whether two points are independent and, hence, can be inserted concurrently.

THEOREM 1. $\bar{r}$ $p_i, p_j \in \Omega$ $\|p_i - p_j\| \geq 4\bar{r}$ p_i p_j

2.3.2. The circumradius upper bound invariant. For Theorem 1 to be applicable throughout the run of the algorithm, we need to prove another invariant.

LOOP INVARIANT 2 (circumradius upper bound). $\bar{r}$

Next, we show that the execution of the B-W algorithm, either sequentially or in parallel, does not violate Loop Invariant 2.

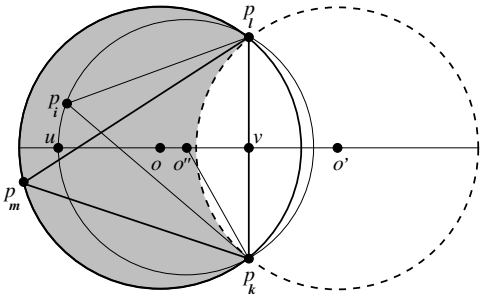


FIG. 4. *The solid circle corresponds to* $\bigcirc(\triangle p_k p_l p_m)$ *with the center in point* o*, and the dashed circle corresponds to* $\bigcirc'_{p_k p_l}(\triangle p_k p_l p_m)$ *with the center in point* o'*. Point* o'' *is the center of the variable-radius disk, whose circle passes through* p_k *and* p_l*. We prove that for any point* p_i *inside the shaded region,* $r(\triangle p_k p_l p_i) < r(\triangle p_k p_l p_m)$*.*

Let the [illegible] $\bigcirc(\triangle p_k p_l p_m)$ [illegible] $p_k p_l$ be the disk $\bigcirc'_{p_k p_l}(\triangle p_k p_l p_m)$ that has the same radius, whose circle passes through the points p_k and p_l, and whose center lies on the other side of edge $p_k p_l$ from point p_m. See Figure 4.

LEMMA 3. [illegible] p_i [illegible] $\bigcirc(\triangle p_k p_l p_m) \setminus \bigcirc'_{p_k p_l}(\triangle p_k p_l p_m)$ [illegible] 4. $r(\triangle p_k p_l p_i) < r(\triangle p_k p_l p_m)$

[illegible] Let o'' be the center of the disk $\bigcirc(\triangle p_k p_l p_i)$, where p_i is any point inside the shaded region $\bigcirc(\triangle p_k p_l p_m) \setminus \bigcirc'_{p_k p_l}(\triangle p_k p_l p_m)$. o'' has to lie on the straight line through o and o'. Let v be the point of intersection of this straight line with the edge $p_k p_l$ (i.e., the midpoint of $p_k p_l$), and let u be the point of intersection of the straight line with the circle of $\bigcirc(\triangle p_k p_l p_i)$ inside the shaded region. Let $x = \|u - v\|$. We will express the radius of $\bigcirc(\triangle p_k p_l p_i)$ as a function of x: $r(\triangle p_k p_l p_i) = f(x)$.

Let us consider the triangle $\triangle p_k v o''$ with the right angle at the vertex v. If we denote $\|p_k - p_l\| = a$, and $\|v - o\| = \|v - o'\| = b$, then by observing that $\|p_k - v\| = a/2$, $\|v - o''\| = x - f(x)$, and $\|o'' - p_k\| = f(x)$, we have

$$f^2(x) = \left(\frac{a}{2}\right)^2 + (x - f(x))^2 \quad \text{or} \quad f(x) = \frac{a^2}{8x} + \frac{x}{2}, \quad \text{where } 0 < x < \infty.$$

$f(x)$ is convex everywhere on $0 < x < \infty$ since its second derivative is positive:

$$f''(x) = \frac{a^2}{4x^3} > 0 \quad \text{for } 0 < x < \infty.$$

In addition,

$$\begin{aligned} f(x)|_{o''=o'} &= f(r(\triangle p_k p_l p_m) - b) = r(\triangle p_k p_l p_m), \\ f(x)|_{o''=o} &= f(r(\triangle p_k p_l p_m) + b) = r(\triangle p_k p_l p_m). \end{aligned}$$

Using Jensen's inequality (see [40]), for $0 \le \lambda \le 1$ and $r(\triangle p_k p_l p_m) - b < x < r(\triangle p_k p_l p_m) + b$, we can derive the following:

$$\begin{aligned} f(x) &< \lambda f(r(\triangle p_k p_l p_m) - b) + (1 - \lambda) f(r(\triangle p_k p_l p_m) + b) \\ &= \lambda r(\triangle p_k p_l p_m) + (1 - \lambda) r(\triangle p_k p_l p_m) \\ &= r(\triangle p_k p_l p_m). \qquad \square \end{aligned}$$

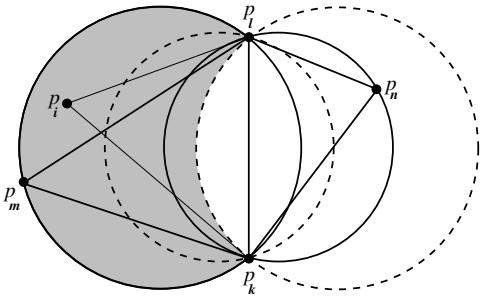


FIG. 5. $\triangle p_k p_l p_m \in \mathcal{C}(p_i)$, $\triangle p_k p_n p_l \notin \mathcal{C}(p_i)$, *and* $r(\triangle p_k p_l p_m) > r(\triangle p_k p_n p_l)$.

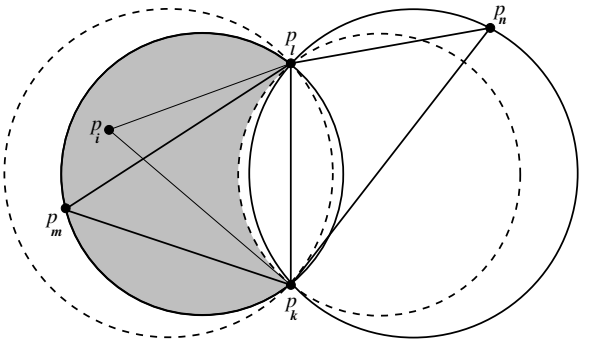


FIG. 6. $\triangle p_k p_l p_m \in \mathcal{C}(p_i)$, $\triangle p_k p_n p_l \notin \mathcal{C}(p_i)$, *and* $r(\triangle p_k p_l p_m) \leq r(\triangle p_k p_n p_l)$.

LEMMA 4. [illegible] $\triangle p_k p_l p_m \in \mathcal{C}(p_i)$ [illegible] $\triangle p_k p_n p_l \notin \mathcal{C}(p_i)$ [illegible] $r(\triangle p_k p_l p_i) < \max(r(\triangle p_k p_l p_m), r(\triangle p_k p_n p_l))$

[illegible] There are two cases:

(i) $r(\triangle p_k p_l p_m) > r(\triangle p_k p_n p_l)$; see Figure 5. In this case, p_i has to lie inside the region $\bigcirc(\triangle p_k p_l p_m) \setminus \bigcirc(\triangle p_k p_n p_l)$, which is a subset of the region $\bigcirc(\triangle p_k p_l p_m) \setminus \bigcirc'_{p_k p_l}(\triangle p_k p_l p_m)$, and, according to Lemma 3, $r(\triangle p_k p_l p_i) < r(\triangle p_k p_l p_m)$.

(ii) $r(\triangle p_k p_l p_m) \leq r(\triangle p_k p_n p_l)$; see Figure 6. Again, p_i has to lie in the region $\bigcirc(\triangle p_k p_l p_m) \setminus \bigcirc(\triangle p_k p_n p_l)$, which in this case is a subset of $\bigcirc'_{p_k p_l}(\triangle p_k p_n p_l) \setminus \bigcirc(\triangle p_k p_n p_l)$, and, by Lemma 3, $r(\triangle p_k p_l p_i) < r(\triangle p_k p_n p_l)$. □

Thus, we may state the following theorem.

THEOREM 2. [illegible] 2 [illegible]

[illegible] The B-W algorithm creates only triangles of the form $\triangle p_k p_l p_i$, where there exists $\triangle p_k p_l p_m \in \mathcal{M}$ such that $\triangle p_k p_l p_m \in \mathcal{C}(p_i)$. There are two cases:

(i) There exists $\triangle p_k p_n p_l \in \mathcal{M}$ such that $\triangle p_k p_n p_l \notin \mathcal{C}(p_i)$, in other words, $p_k p_l$ is not a boundary segment. In this case, by Lemma 4, $r(\triangle p_k p_l p_i) < \max(r(\triangle p_k p_l p_m), r(\triangle p_k p_n p_l))$.

(ii) $p_k p_l$ is a boundary segment. Then p_i must be outside the diametral circle of $p_k p_l$; otherwise, p_i wouldn't have been inserted. Consequently, p_i lies in the region $\bigcirc(\triangle p_k p_l p_m) \setminus \bigcirc'_{p_k p_l}(\triangle p_k p_l p_m)$, and, by Lemma 3, $r(\triangle p_i p_k p_l) < r(\triangle p_k p_l p_m)$. □

2.3.3. Adjusting the degree of concurrency. We have shown how with a simple and inexpensive test one can check whether two points (i.e., circumcenters) can be inserted independently. Now it remains to provide a way of finding enough independent circumcenters so that all processors can be kept busy inserting them. This task can be accomplished by decreasing all triangle circumradii below some value $\bar{r}$ with the purpose of increasing the number of pairwise independent circumcenters. Fortunately, this can be easily done by applying the sequential Delaunay refinement procedure as a preprocessing step. In this step, we use the initial parameter $\bar{\rho}$ and select parameter $\bar{\Delta}$ as

$$\bar{\Delta} = \frac{\bar{r}^2}{4\bar{\rho}^3}, \tag{2.2}$$

where $\bar{r}$ is computed based on the number of available processors and the size of the domain (see section 3).

Such preprocessing allows us to decrease the maximal circumradius of the triangles in a mesh automatically as a result of decreasing the maximal circumradius-to-shortest edge ratio and the maximal area. In the rest of this subsection, we prove (2.2).

We start by proving an auxiliary lemma, which relates the length of any side of a triangle to the sum of the cotangents of the adjacent angles and the triangle's area.

LEMMA 5. a b c A B C

$$a^2 = 2(\cot B + \cot C)\Delta, \tag{2.3}$$

Δ

. The cyclical application of the law of cosines,

$$\cos A = \frac{b^2 + c^2 - a^2}{2bc}, \tag{2.4}$$

coupled with the formula computing the area of the triangle,

$$\Delta = \frac{1}{2} bc \sin A, \tag{2.5}$$

allows us to obtain the following formula:

$$\cot A + \cot B + \cot C = \frac{a^2 + b^2 + c^2}{4\Delta}. \tag{2.6}$$

Rearranging (2.5) and substituting into (2.4) yields

$$\cot A = \frac{b^2 + c^2 - a^2}{4\Delta}. \tag{2.7}$$

Finally, by coupling (2.6) and (2.7), we obtain (2.3). ∎

Now, we show how an expression involving the smallest angle of a triangle bounds the sum of the cotangents of the other angles.

LEMMA 6. $A \le B \le C$

$$\cot B + \cot C \le \frac{1}{\sin A}.$$

Since A, B, C are positive summing themselves up to π, we can define the following function:

$$f(B) = \cot B + \cot C = \cot B + \cot(\pi - A - B) = \cot B - \cot(A + B),$$

where

$$0 < A \le B \le \frac{\pi - A}{2} \le C \le \pi - 2A, \quad \text{and} \quad A \le \frac{\pi}{3}. \tag{2.8}$$

$f(B)$ is continuous on $\left[A, \frac{\pi-A}{2}\right]$ and has no points of local maxima which satisfy (2.8); thus, it can reach its maxima only in the ends of the interval $\left[A, \frac{\pi-A}{2}\right]$. Direct substitution yields

$$f(A) = \frac{1}{2\sin A\cos A} \le \frac{1}{2\sin A \cdot \frac{1}{2}} = \frac{1}{\sin A}$$

and

$$f\left(\frac{\pi - A}{2}\right) = 2\frac{1 - \cos A}{\sin A} \le 2\frac{1 - \frac{1}{2}}{\sin A} = \frac{1}{\sin A}.$$

Hence,

$$f(B) \le \frac{1}{\sin A} \quad \forall B \in \left[A, \frac{\pi - A}{2}\right]. \qquad \square$$

Finally, assuming that the bounds $\bar{\rho}$ and $\bar{\Delta}$ on the triangle area and the circumradius-to-shortest edge ratio, respectively, hold for all triangles in the mesh, we find out the upper bound $\bar{r}$ on the circumradius of triangles.

LEMMA 7. [illegible] r [illegible] $a \le b \le c$ [illegible] ρ [illegible] $\bar{\rho}$ $\rho = r/a < \bar{\rho}$ [illegible] $\bar{\Delta}$ $\Delta < \bar{\Delta}$ [illegible]

$$r < 2(\bar{\rho})^{3/2}\sqrt{\bar{\Delta}}.$$

From $\rho = r/a$, or $r = \rho a$, using Lemma 5 it follows that

$$r = \rho a = \rho\sqrt{2(\cot B + \cot C)\Delta}$$

and Lemma 6 implies that

$$r \le \rho\sqrt{2\frac{1}{\sin A}\Delta} = \rho\sqrt{\frac{2\Delta}{\sin A}}. \tag{2.9}$$

By substituting $\sin A$ from (2.1) into (2.9), we have

$$r \le 2\rho^{3/2}\sqrt{\Delta} < 2(\bar{\rho})^{3/2}\sqrt{\bar{\Delta}}. \qquad \square$$

The following theorem directly follows from Lemma 7.

THEOREM 3. $\bar{\rho}$ $\bar{\Delta}$ [illegible] $\bar{r}$ [illegible]

$$\bar{r} = 2(\bar{\rho})^{3/2}\sqrt{\bar{\Delta}},$$

[illegible]

```
ParameterizedDelaunayRefinement(𝒳, ℳ, Δ̄, ρ̄, σ())
Input: 𝒳 is a PSLG which defines Ω
       ℳ is some Delaunay mesh over Ω
       Δ̄ and ρ̄ are desired upper bounds
       σ() is a boolean predicate which takes point coordinates as input
Output: a conforming Delaunay mesh ℳ such that
            ∀t ∈ ℳ  σ(⊙(t)) ⇒ (ρ(t) < ρ̄) ∧ (Δ(t) < Δ̄)
 1  Q ← {t ∈ ℳ | (ρ(t) ≥ ρ̄) ∨ (Δ(t) ≥ Δ̄)}
 2  for each t ∈ Q
 3      p ← ⊙(t)
 4      if σ(p)
 5          Find S, a set of segments encroached upon by p
 6          if S ≠ ∅
 7              SplitSegments(S)
 8          else
 9              ℳ ← ℳ \ 𝒞_ℳ(p) ∪ {△puv | (u, v) ∈ ℬ_ℳ(p)}
10          endif
11          Update Q
12      endif
13  endfor
14  return ℳ
```

FIG. 7. *An augmented sequential Delaunay refinement algorithm.*

3. Coarse grain partitioning. One possible way to break the meshing problem of the entire domain up into smaller independent subproblems is to partition the domain into subdomains in such a way that the circumcenters of the triangles in each subdomain can be inserted concurrently with the circumcenters in any other subdomain.

If we consider a selected spatial region as a subdomain, which is refined sequentially by one processor, then the layer of triangles whose circumcenters are within $2\bar{r}$ from any circumcenter in the region forms the buffer zone of the entire region. The partitioning and the decoupling of the subdomains can be achieved by creating a special buffer zone around every subdomain, so that the insertion of a point in one subdomain can modify the triangles only inside this subdomain and its buffer zone, but the changes will not propagate to other subdomains and their buffer zones. After refining the subdomains, the buffer zones can be refined in a similar way.

We have chosen a simple and efficient way to partition the domain, which consists in dividing its bounding box into squares. One of the main advantages of this approach is the efficiency of assigning triangles to subdomains based on the coordinates of their circumcenters. A small adjustment (see Figure 7) to the sequential Delaunay refinement algorithm (Figure 1) allows the insertion of only those circumcenters which satisfy some user-defined condition (i.e., being inside a given rectangle). The drawback is that the amount of work assigned to different processors can vary significantly. However, we developed a runtime system [2] for dynamic load balancing which solves this problem, and thus we do not have to address it here.

The selection of subdomain box sizes is closely related to the scheduling of the refinement phases. First, Theorem 1 requires that the regions simultaneously refined by different processors are at least $4\bar{r}$ away. Second, the refinement of a region can

ParallelDelaunayMeshing($\mathcal{X}$, $\bar{\Delta}$, $\bar{\rho}$, P, p)
Input: $\mathcal{X}$ is a PSLG which defines Ω
$\bar{\Delta}$ and $\bar{\rho}$ are desired upper bounds
P is the total number of processors (we assume $\sqrt{P}$ is integer)
p is the index of the current processor, $0 \le p \le P-1$
Output: a distributed conforming Delaunay mesh respecting $\bar{\Delta}$ and $\bar{\rho}$

1 Calculate $row(p)$ and $column(p)$ of the current processor
// $0 \le row(i), column(i) \le \sqrt{P}-1,\ 0 \le i \le P-1$
2 Let l be the longest dimension of the bounding box of Ω
3 $\bar{r} \leftarrow l/(4\sqrt{P})$
4 **if** $p = 0$
5 Construct $\mathcal{M}_1$, a constrained Delaunay mesh of Ω
6 $\bar{\Delta}_1 \leftarrow \bar{r}^2/(4\bar{\rho}^3)$
7 $\mathcal{M}_1 \leftarrow$ ParameterizedDelaunayRefinement($\mathcal{X}$, $\mathcal{M}_1$, $\bar{\Delta}_1$, $\bar{\rho}$, ...)
8 Assign triangles in $\mathcal{M}_1$ to cells $\{d_{ij} \mid 0 \le i,j \le 4\sqrt{P}-1\}$
of side length $4\bar{r}$ based on the coordinates of their circumcenters
9 **for each** $i \in \{0,\ldots,P-1\}$
10 Send cells $\{d_{m,n} \mid 4row(i)-1 \le m \le 4(row(i)+1)$,
$4column(i)-1 \le n \le 4(column(i)+1)\}$ to processor i
11 **endfor**
12 **endif**
13 Receive cells $\{d_{ij} \mid 1 \le i,j \le 4\}$ of local mesh $\mathcal{M}$
// Phase (0) is over
14 Send cells $\{d_{4,j} \mid 1 \le j \le 3\}$ to processor in $(row(p)+1,\ column(p))$
15 Send cells $\{d_{i,4} \mid 1 \le i \le 3\}$ to processor in $(row(p),\ column(p)+1)$
16 Send cell $d_{4,4}$ to processor in $(row(p)+1,\ column(p)+1)$
17 Receive cells $\{d_{0,j} \mid 1 \le j \le 3\}$ from processor in $(row(p)-1,\ column(p))$
18 Receive cells $\{d_{i,0} \mid 1 \le i \le 3\}$ from processor in $(row(p),\ column(p)-1)$
19 Receive cell $d_{0,0}$ from processor in $(row(p)-1,\ column(p)-1)$
// Phase (1) is over
20 Let (x_0, y_0) be the upper left coordinate of local cell $d_{0,0}$
21 $\sigma(q) \leftarrow (q \in [x_0+2\bar{r}, y_0+2\bar{r}] \times [x_0+14\bar{r}, y_0+14\bar{r}])$
22 $\mathcal{M} \leftarrow$ ParameterizedDelaunayRefinement($\mathcal{X}$, $\mathcal{M}$, $\bar{\Delta}$, $\bar{\rho}$, $\sigma()$)
// Phase (2) is over
// Phases (3)–(10) are performed by analogy (Figure 9)
23 **return** $\mathcal{M}$

FIG. 8. *A parallel Delaunay meshing algorithm. In order to simplify the presentation of the algorithm, we do not address the index out-of-range problems (e.g., d_{-1}), which we handle in the implementation.*

create poor quality triangles near the boundary in a neighboring region, which has been refined in a previous iteration (see Figure 8). To avoid performing multiple refinement iterations of the same region, some overlapping among the regions needs to be introduced. The degree of overlapping depends on the partitioning of Ω. We partition the domain Ω into a two-dimensional lattice of squares, arrange the processors into logical rows and columns, and assign to each processor a fraction (N/P) of the lattice, where N is the number of cells and P is the number of processors, which it is responsible to mesh. We ensure global mesh consistency by forcing each processor to

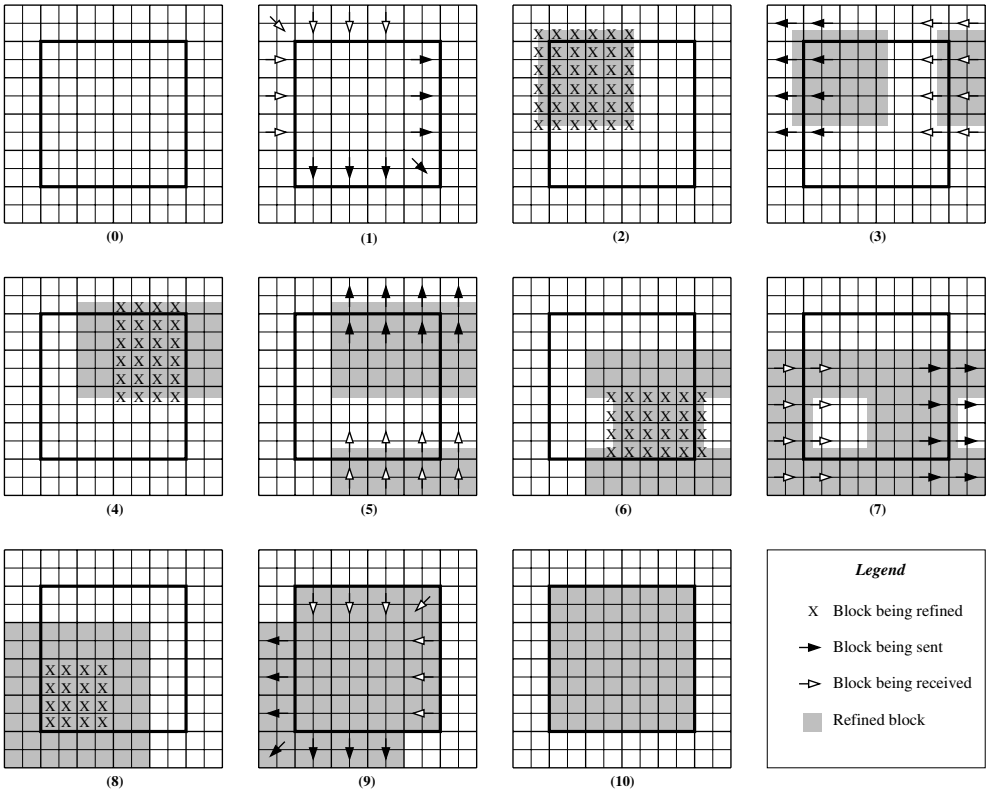


FIG. 9. *A schedule of the parallel Delaunay refinement and communication on distributed memory. Each phase* (0)–(10) *depicts the actions performed by a single processor. The smallest cells have side length* $2\bar{r}$*; they are the atomic units of refinement. Since the refinement may leave fractions of big triangles, whose circumcenters are outside the small cell, the refined cells sometimes are not completely shaded. The bigger cells, which consist of* 4 *small cells, are the atomic units of data migration. The thick lines outline the subdomain assigned to the given processor. A processor also holds parts of its neighbors' meshes. Arrows represent the direction of data migration.*

hold and exchange $(4\bar{r})$-wide layers of triangles near the boundaries of their assigned subdomains. Each processor in position (i, j) communicates only with its neighbors in positions (s, t), such that $|i - s| \leq 1$ and $|j - t| \leq 1$.

The entire parallel algorithm is summarized in Figure 8. The scheduling of the refinement and the communication phases is also schematically presented in Figure 9. A sequence of meshes created by four processors for the pipe cross-section model (Figure 10) is shown in Figure 11.

4. Parallel implementation. For the experimental evaluation of our method, we triangulated (i) 1000 random points uniformly distributed in a unit square and (ii) the pipe cross-section model (Figure 10). We used the "Triangle" [33] as a sequential mesher on each processor. The Triangle facilitates the use of a user-defined function for deciding which triangles should be considered "big" and queued for refinement. However, apparently, it doesn't provide a mechanism for the user to decide at run-time which triangles are "bad" in terms of the circumradius-to-shortest edge ratio or minimal angle. This fact caused us to insert a test for the triangle circumcenter position inside the Triangle code and recompile it. We believe that in future releases

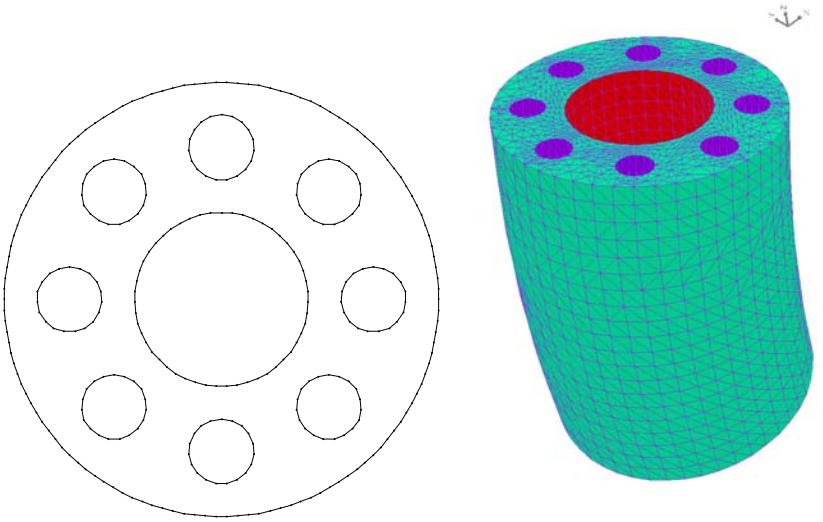


FIG. 10. *A cross section and a cartoon pipe of a regenerative cooled pipe geometry which came from the testing of a rocket engine at NASA Marshall. NASA had to slightly bend one of the regenerative cooled "pipes" that contained combustion gases in order to fit a rocket engine onto a test stand. In testing, this curved pipe failed due to an unanticipated increase of heating. Since this was a one-off design, NASA couldn't afford to do the extensive testing. The failure caused them to scrap that particular project since it destroyed the engine under test. This is an example of why simulation is useful in reducing the cost.*

of sequential mesh generation codes, the facility for defining user-supplied functions, which test the suitability of both types of triangles, can be easily provided.

Another technical detail that we ran into while running our tests is the substantial difference in time required by the Triangle to refine an existing mesh and to construct a fine mesh without taking advantage of a previously constructed coarser mesh. Contrary to our expectations, building a fine mesh anew turned out to be much faster. For example, in the case of the contour of the letter "A," which is included in the Triangle distribution, the Triangle is about six times faster in retriangulating a given set of points than in reconstructing the mesh data structure from a list of triangles. That is why, when moving a part of the mesh from one processor to another, we decided to migrate only the set of points and boundary edges and retriangulate them as necessary. Since the Delaunay triangulation of a given set of points is unique, provided that there are no four cocircular points [34], this process will not destroy the boundary conformity. In the presence of cocircular points near the boundary, the enforcement of boundary segments causes the required edge flips. The additional advantage of this optimization is the decrease of the network traffic.

We have conducted our experiments on the Sciclone cluster computing system at the College of William and Mary using its two tightly coupled subclusters: "whirlwind"—64 single-cpu Sun Fire V120 servers (650 MHz, 1 GB RAM)—and "twister"—32 dual-cpu Sun Fire 280R servers (900 MHz, 2 GB RAM). The scaled and the fixed work-load evaluation for a mesh of a unit square is shown in Table 1, with the breakdown of the total time in Table 2, and for the pipe model in Table 3. We can see that in the pipe case the algorithm performs slightly worse, which happens because of the load imbalance due to the empty space inside pipe holes. Table 4 shows a comparison of a distributed- and a shared-memory implementation on the

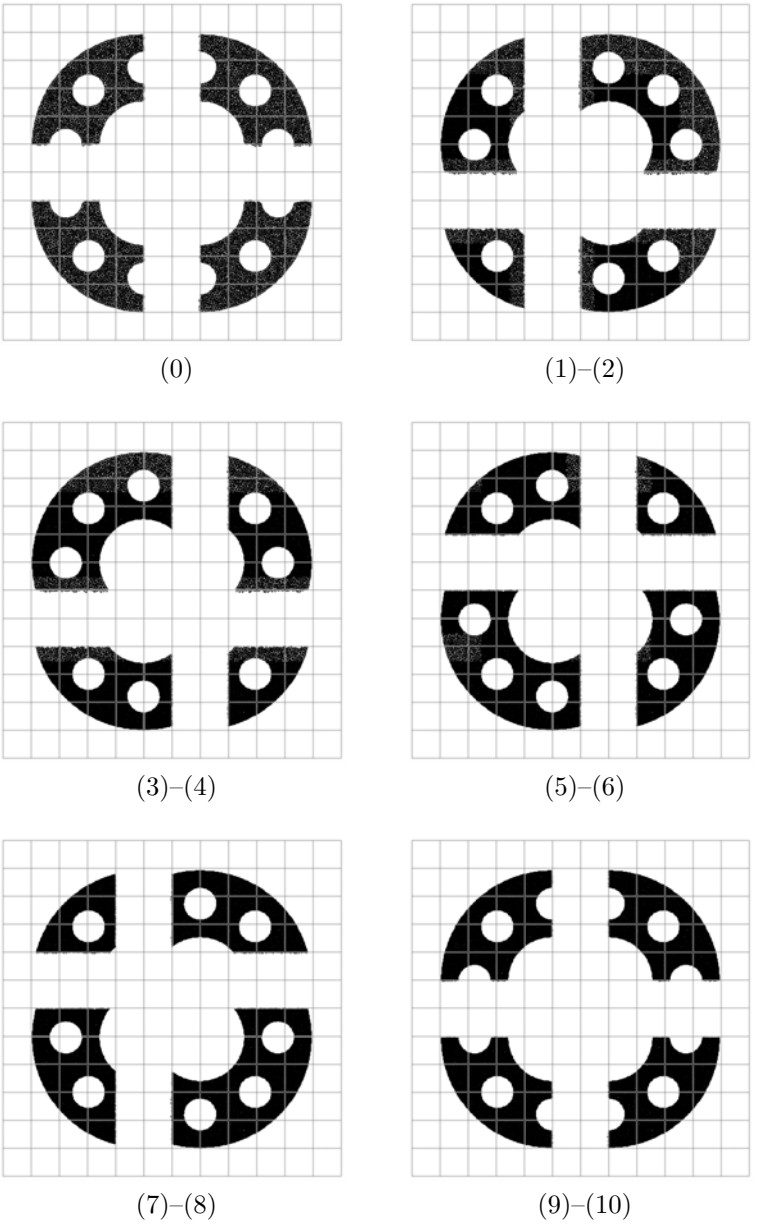


FIG. 11. *Snapshots of a mesh distributed among four processors. Each picture corresponds to the specified phases schematically shown in Figure* 9*. Dark areas are the parts of the mesh that have already been refined. Gray areas are those that have not been refined. Each processor starts with an initial coarse mesh and ends up with a refined mesh covering the same area.*

"hurricane" subcluster.

Finally, we didn't have to write any mesh generation code, since for both the preprocessing and the refinement it was sufficient to call the Triangle as a library.

5. Conclusions. We introduced a parallel coarse-grained mesh generation algorithm based on a block domain decomposition, which can be used in a variety of

TABLE 1
Parallel Delaunay refinement for a mesh of a unit square. For all tests, $\bar{\rho} = 1.41$.

P	Scaled work load			Fixed work load, 23.8M elements, $\bar{\Delta} = 5 \times 10^{-8}$, total time, sec.
	Total time, sec.	Number of elements, millions	Area bound $\bar{\Delta}$ $\times 10^{-8}$	
4	293.7	23.8	5.00	293.7
9	294.7	58.8	2.22	122.3
16	295.4	109.3	1.25	67.2
25	296.8	175.4	0.80	43.3
36	293.4	255.0	0.56	30.2
49	294.5	352.6	0.41	23.3
64	300.1	470.7	0.31	19.4
81	296.2	587.8	0.25	17.0
100	300.3	738.9	0.20	15.6
121	293.7	873.5	0.17	15.1

TABLE 2
A breakdown of the execution time for a single processor in the case of the unit square, in seconds. The communication and synchronization column shows the time to send the data to the neighboring processors and receive the data from them, which includes waiting for their completion of the current refinement task.

P	Sequential prerefinement	Parallel refinement	Communication and synchronization	Filling in and merging cells
4	0.24	218.7	32.7	39.9
9	0.44	220.0	35.4	40.3
16	0.71	218.8	37.1	40.0
25	1.05	219.1	39.1	40.8
36	1.43	218.6	36.1	40.0
49	1.88	217.7	37.5	40.7
64	2.49	221.0	40.7	41.5
81	3.04	216.6	46.2	41.1
100	3.69	218.4	46.5	41.5
121	4.42	210.3	53.6	40.9

TABLE 3
Parallel Delaunay refinement for a mesh of the pipe model. $\bar{\rho} = 1.41$.

P	Scaled work load			Fixed work load, 14.6M elements, $\bar{\Delta} = 2 \times 10^{-2}$, total time, sec.
	Total time, sec.	Number of elements, millions	Area bound $\bar{\Delta}$, $\times 10^{-2}$	
4	179.1	14.6	2.00	179.1
9	204.8	32.8	0.88	105.6
16	225.7	58.3	0.50	80.5
25	239.8	91.1	0.32	54.2
36	255.3	131.2	0.22	40.1
49	253.7	178.6	0.16	30.2
64	273.6	233.3	0.13	21.9
81	258.6	295.3	0.10	17.1
100	232.2	364.6	0.08	15.0
121	212.7	441.1	0.06	14.8

contexts. First, as we showed, the parallel algorithm can be implemented efficiently on distributed- and shared-memory systems by exploiting the fact that only local communication between processors is necessary, the data can be sent in large blocks at the

TABLE 4
Pipe cross-section, distributed- (MPI), and shared- (OpenMP) memory implementations.

Number of processors	Time, sec. MPI	Time, sec. OpenMP	Number of elements, millions
4	220.3	214.1	14.6

beginning of each computation phase, and the number of the communication phases is a small constant (five). In this paper, we addressed the code reuse issue from the viewpoint of concurrency; the design of the generic application programming interface is out of the scope; see, e.g., [3] for a discussion. Second, we designed an out-of-core algorithm [24] which utilizes the absence of communication during subdomain refinement phases. Third, a shared-memory implementation can use the sufficient condition of the circumcenter independence to guide the fine-grained parallelism. However, our experimental evaluation [1] of fine-grain parallelism in a B-W kernel showed that current simultaneous multithreaded processors (SMTs) are not capable of achieving performance improvement, while the experiments on a simulated SMT indicate that with modest hardware support it is possible to exploit fine-grain parallelism opportunities. Fourth, our future work includes the extension of the framework to the three-dimensional domains. We also plan to improve the performance of our parallel implementation, for more irregular geometries, by addressing the work-load balancing problem using our runtime system [2] and improving the memory management for a single processor. Finally, we are currently working on the extension of the theoretical framework for the construction of nonuniform (graded) meshes [7].

Acknowledgments. We thank Andriy Kot for the shared-memory implementation of the algorithm. We appreciate the very detailed comments from the anonymous reviewers which helped us improve the manuscript.

REFERENCES

[1] C. D. ANTONOPOULOS, X. DING, A. N. CHERNIKOV, F. BLAGOJEVIC, D. S. NIKOLOPOULOS, AND N. P. CHRISOCHOIDES, *Multigrain parallel Delaunay mesh generation: Challenges and opportunities for multithreaded architectures*, in Proceedings of the 19th Annual International Conference on Supercomputing, ACM Press, New York, 2005, pp. 367–376.

[2] K. BARKER, A. CHERNIKOV, N. CHRISOCHOIDES, AND K. PINGALI, *A load balancing framework for adaptive and asynchronous applications*, IEEE Trans. Parallel Distrib. Systems, 15 (2004), pp. 183–192.

[3] G. BERTI, *Gral—the grid algorithms library*, in International Conference on Computational Science (3), P. M. A. Sloot, C. J. K. Tan, J. Dongarra, and A. G. Hoekstra, eds., Lecture Notes in Comput. Sci. 2331, Springer, Berlin, 2002, pp. 745–754.

[4] G. E. BLELLOCH, J. C. HARDWICK, G. L. MILLER, AND D. TALMOR, *Design and implementation of a practical parallel Delaunay algorithm*, Algorithmica, 24 (1999), pp. 243–269.

[5] A. BOWYER, *Computing Dirichlet tesselations*, Comput. J., 24 (1981), pp. 162–166.

[6] A. N. CHERNIKOV AND N. P. CHRISOCHOIDES, *Practical and efficient point insertion scheduling method for parallel guaranteed quality Delaunay refinement*, in Proceedings of the 18th Annual International Conference on Supercomputing, ACM Press, New York, 2004, pp. 48–57.

[7] A. N. CHERNIKOV AND N. P. CHRISOCHOIDES, *Parallel 2D graded guaranteed quality Delaunay mesh refinement*, in Proceedings of the 14th International Meshing Roundtable, Springer, New York, 2005, pp. 505–517.

[8] L. P. CHEW, N. CHRISOCHOIDES, AND F. SUKUP, *Parallel constrained Delaunay meshing*, in ASME/ASCE/SES Summer Meeting, Special Symposium on Trends in Unstructured Mesh Generation, Northwestern University, Evanston, IL, 1997, pp. 89–96.

[9] L. P. CHEW, N. CHRISOCHOIDES, K. PINGALI, AND S. STODGHIL, *Meshing Using Independent Sets*, manuscript, 1999.

[10] N. Chrisochoides and D. Nave, *Parallel Delaunay mesh generation kernel*, Internat. J. Numer. Methods Engrg., 58 (2003), pp. 161–176.

[11] N. P. Chrisochoides, *A Survey of Parallel Mesh Generation Methods*, Tech. Report BrownSC-2005-09, Brown University, 2005. Also appears as a chapter in Numerical Solution of Partial Differential Equations on Parallel Computers, M. Bruaset, P. Bjorstad, and A. Tveito, eds., Springer, New York, pp. 237–259.

[12] H. L. de Cougny, M. S. Shephard, and C. Ozturan, *Parallel three-dimensional mesh generation*, Comput. Systems Engrg., 5 (1994), pp. 311–323.

[13] B. N. Delaunay, *Sur la sphere vide*, Izvestia Akademia Nauk SSSR, VII Seria, Otdelenie Mataematicheskii i Estestvennyka Nauk, 7 (1934), pp. 793–800.

[14] S. Dong, D. Lucor, and G. Em Karniadakis, *Flow past a stationary and moving cylinder: DNS at Re* = 10,000, in 2004 Users Group Conference (DOD_UGC'04), IEEE, Williamsburg, VA, 2004, pp. 88–95.

[15] H. Edelsbrunner and D. Guoy, *Sink-insertion for mesh improvement*, in Proceedings of the 7th Annual Symposium on Computational Geometry, ACM Press, New York, 2001, pp. 115–123.

[16] P.-L. George and H. Borouchaki, *Delaunay Triangulation and Meshing. Application to Finite Elements*, Hermès, Paris, 1998.

[17] G. Globisch, *On an automatically parallel generation technique for tetrahedral meshes*, Parallel Comput., 21 (1995), pp. 1979–1995.

[18] G. Globisch, *Parmesh—A parallel mesh generator*, Parallel Comput., 21 (1995), pp. 509–524.

[19] Y. Ito, A. M. Shih, A. K. Erukala, B. K. Soni, A. N. Chernikov, N. P. Chrisochoides, and K. Nakahashi, *Generation of unstructured meshes in parallel using an advancing front method*, in Proceedings of the 9th International Conference on Numerical Grid Generation in Computational Field Simulations, International Society of Grid Generation, 2005.

[20] D. A. Jefferson, *Virtual time*, ACM Trans. Program. Languages Systems, 7 (1985), pp. 404–425.

[21] C. Kadow, *Adaptive dynamic projection-based partitioning for parallel Delaunay mesh generation algorithms*, in SIAM Workshop on Combinatorial Scientific Computing, 2004.

[22] C. Kadow and N. Walkington, *Design of a projection-based parallel Delaunay mesh generation and refinement algorithm*, in Proceedings of the 4th Symposium on Trends in Unstructured Mesh Generation, 2003; http://www.andrew.cmu.edu/user/sowen/usnccm03/agenda.html.

[23] G. E. Karniadakis and S. A. Orszag, *Nodes, modes, and flow codes*, Phys. Today, 46 (1993), pp. 34–42.

[24] A. Kot, A. N. Chernikov, and N. P. Chrisochoides, *Out-of-core parallel Delaunay mesh generation*, in Proceedings of the 17th IMACS World Congress Scientific Computation, Applied Mathematics and Simulation, Paper T1-R-00-0710, Institute of Computational Mathematics and Mathematical Geophysics of the Russian Academy of Sciences, 2005.

[25] V. Kumar, A. Grama, A. Gupta, and G. Karypis, *Introduction to Parallel Computing: Design and Analysis of Parallel Algorithms*, Benjamin/Cummings, Redwood City, CA, 1994.

[26] C. L. Lawson, *Software for* C^1 *surface interpolation*, in Mathematical Software, III, Academic Press, New York, 1977, pp. 161–194.

[27] L. Linardakis and N. Chrisochoides, *Delaunay decoupling method for parallel guaranteed quality planar mesh refinement*, SIAM J. Sci. Comput., 27 (2006), pp. 1394–1423.

[28] R. Löhner and J. R. Cebral, *Parallel advancing front grid generation*, in Proceedings of the 8th International Meshing Roundtable, Sandia National Laboratories, Albuquerque, NM, 1999, pp. 67–74.

[29] G. L. Miller, D. Talmor, S.-H. Teng, and N. Walkington, *A Delaunay based numerical method for three dimensions: Generation, formulation, and partition*, in Proceedings of the 27th Annual ACM Symposium on Theory of Computing, ACM Press, New York, 1995, pp. 683–692.

[30] D. Nave, N. Chrisochoides, and L. P. Chew, *Guaranteed-quality parallel Delaunay refinement for restricted polyhedral domains*, Comput. Geom., 28 (2004), pp. 191–215.

[31] J. Ruppert, *A Delaunay refinement algorithm for quality* 2*-dimensional mesh generation*, J. Algorithms, 18 (1995), pp. 548–585.

[32] J. R. Shewchuk, *Lecture notes on Delaunay mesh generation*, http://www.cs.berkeley.edu/~jrs/meshpapers/delnotes.ps.gz (accessed November 2005).

[33] J. R. Shewchuk, *Triangle: Engineering a* 2*D Quality Mesh Generator and Delaunay Triangulator*, in Applied Computational Geometry: Towards Geometric Engineering, M. C. Lin and D. Manocha, eds., Lecture Notes in Comput. Sci. 1148, Springer, Berlin, 1996, pp. 203–222.

[34] J. R. Shewchuk, *Delaunay Refinement Mesh Generation*, Ph.D. thesis, Carnegie Mellon University, Pittsburgh, PA, 1997.
[35] J. R. Shewchuk, *Delaunay refinement algorithms for triangular mesh generation*, Comput. Geom., 22 (2002), pp. 21–74.
[36] D. A. Spielman, S.-H. Teng, and A. Üngör, *Parallel Delaunay refinement: Algorithms and analyses*, in Proceedings of the 11th International Meshing Roundtable, Sandia National Laboratories, Albuquerque, NM, 2001, pp. 205–217.
[37] J. R. Shewchuk, *Time complexity of practical parallel Steiner point insertion algorithms*, in Proceedings of the 16th Annual ACM Symposium on Parallelism in Algorithms and Architectures, ACM Press, New York, 2004, pp. 267–268.
[38] S.-H. Teng, *Personal communication*, 2004.
[39] D. F. Watson, *Computing the n-dimensional Delaunay tesselation with application to Voronoi polytopes*, Comput. J., 24 (1981), pp. 167–172.
[40] R. Webster, *Convexity*, Oxford Science, New York, 1994, p. 200.

SIAM J. SCI. COMPUT.
Vol. 28, No. 5, pp. 1927–1956
© 2006 Society for Industrial and Applied Mathematics

BEHAVIOR OF FINITE VOLUME SCHEMES FOR HYPERBOLIC CONSERVATION LAWS ON ADAPTIVE REDISTRIBUTED SPATIAL GRIDS*

CH. ARVANITIS† AND A. I. DELIS‡

Abstract. In this work we consider finite volume schemes combined with dynamic spatial mesh redistribution. We study whether appropriate mesh redistribution is a satisfactory mechanism for increasing the resolution of numerical solutions for problems of scalar and systems of conservation laws (CL) in one space dimension, while being at the same time a stabilization mechanism for selecting the appropriate entropy solution. In order to increase the resolution around shock areas and keep the computational cost low, our redistribution policy is to reconstruct spatially the numerical solution on a new mesh, where the solution's curvature is almost uniformly distributed, while the node's cardinality is kept constant. We examine the stabilization properties of that redistribution process by adding it as a substep on the time evolution step of some classical schemes with known (unstable) characteristics. Testing the resulting method for several such schemes and on a large number of CL problems that have solutions with special characteristics (shocks, rarefaction areas, steady states) and comparing the results with those produced by schemes with extra stabilization mechanisms (like slope/flux limiters, entropy corrections), we conclude that indeed the proposed redistribution adds such stabilization properties while at the same time increasing the resolution.

Key words. finite volume methods, adaptive grid redistribution, hyperbolic conservation laws

AMS subject classifications. 35L65, 76M12, 65M12

DOI. 10.1137/050632853

1. Introduction.

The application of finite volume schemes is a very popular choice for computing solutions of systems of conservation laws (CL) in the following context: $u : \mathbb{R}^d \times [0,T] \to \mathbb{R}^M$

$$\begin{aligned} u(\cdot,0) &= u_0 \\ \partial_t u + \sum_{i=1}^{d} \partial_{x_i} F_i(u) &= 0. \end{aligned} \tag{1.1}$$

Some classical schemes, of first or second order in space, when applied directly to this system will result in computational solutions with diffusive or oscillatory behavior, especially close to shocks. To overcome this difficulty several modifications of such schemes have been proposed in the literature where the necessary stability and viscosity mechanisms are imposed "by hand." Mesh adaptation is a main current stream to efficiently compute numerical solutions of complex systems by increasing the resolution of the essential solution. Several redistribution techniques have been introduced recently for solving the problem of proper mesh selection, starting with the self-adjusting method of Harten and Hyman [14] to the moving mesh methods of Azarenok et al., Fazio and LeVeque, Stockie et al., Tao Tang, Huazhong Tang, and

*Received by the editors June 1, 2005; accepted for publication (in revised form) May 15, 2006; published electronically November 16, 2006.
http://www.siam.org/journals/sisc/28-5/63285.html

†Department of Applied Mathematics, University of Crete, Heraklion 71409, Crete, Greece (arvas@tem.uoc.gr). The work of this author was partially supported by the European Union RTN-network HYKE, HPRN-CT-2002-00282, and the program Pythagoras of EPEAEK II.

‡Department of Sciences, Division of Mathematics, Technical University of Crete, University Campus, Chania 73100, Crete, Greece (adelis@science.tuc.gr).

others (see [4], [5], [6], [8], [12], [17], [19], [20], [21], [23], [27], [28], [29], [30]). These methods calculate the spatial positions of the nodes of the new mesh, some of them by solving an Euler–Lagrange equation, others by optimizing proper energy metrics. All of them have as a common factor that they can be combined with any numerical scheme (making appropriate modifications) for increasing its resolution.

In this work the evolving mesh is constructed such that its spatial resolution is controlled via selective characteristics of the computed solution. Our main aim here is to experimentally study the behavior of finite difference-volume schemes, with or without stabilization mechanisms, but under the regime of this adaptively evolving mesh. Our choice of classical schemes includes, for example, the first order Roe scheme, the second order Lax–Wendroff and MacCormack schemes, and also some TVD schemes. The adaptive procedure studied in this work is based on a mesh redistribution policy that evolves within every computational time step. The basic principle of the suggested mesh redistribution is to (re)distribute the nodes of the generated partition with respect to geometrical characteristics of the solution. These characteristics are defined through a positive functional of the solution, the so-called [illegible] [1], [2], [3]. Among other estimator functions for evolution PDEs, like the arc-length and variance, we choose the curvature of the solution as such a function, for its diffuseless behavior. Experiments in [3] with Galerkin finite element schemes, which approximate CL solutions like central finite difference schemes, have shown that this redistribution procedure has stabilization properties of its own. For example, in this work we show that schemes like Lax–Wendroff combined with our redistribution policy provide surprisingly stable solutions free of oscillations. From the numerical experiments produced in the course of this work we concluded the following advantages of the use of this particular adaptive grid redistribution (AGR) method.

- The method when applied to classical second order schemes, which produce oscillating solutions, suppresses the oscillations, producing TVD-like approximations. Classical schemes like the Lax–Wendroff or the MacCormack scheme become stable and produce reliable solutions.
- When applied to numerical schemes that do not satisfy entropy conditions, for example, the original Roe scheme, the method approximates the entropy satisfying solution.
- The method works well for hyperbolic problems with source terms and produces stable solutions for first and second order balanced schemes and can converge to correct steady states.
- The method can automatically detect, resolve, and track steep wave fronts and discontinuities, without having to resort to finer grids.
- The AGR method is of linear complexity, as we will see in the next section (cf. Remark 3), and consequently its computational cost is in favor when compared, for example, to the stabilization mechanisms for high resolution TVD schemes, where we have to solve a number of Riemann problems in order to compute the corresponding stabilization limiters.
- The mechanism has been proved robust for all the applications presented and schemes used.

Following from the above, our main conclusion from the present work is that numerical schemes, when combined with the proposed mesh adaptation, yield stable solutions free of oscillations.

The paper is organized as follows. In the next section we define the AGR algorithm. In section 3 we present finite volume schemes for scalar CL, adjusted to nonuniform grids, which will be tested in the regime of the adaptive redistribution al-

gorithm. Section 4 is devoted to numerical results. In section 5 numerical schemes for systems of CL, with or without source terms, are detailed, and in section 6 numerical results for the shallow water system of equations are presented.

2. The AGR method in one dimension. Let X be a partition of a domain $[a,b]$ and $\{x_i\}_{i=0}^{N}$ the corresponding nodes in increasing order, i.e., $a = x_0 < x_1 < \cdots < x_{N-1} < x_N = b$. Then we introduce the following notation:

- With $\sigma(X)$ we denote the σ-algebra of X, which for finite partitions coincides with the superset of unions of sets from X.
- With **resolution** of X, we denote the measure which for any measurable set A is defined by

$$\mathbf{resolution}(A) = \operatorname{card}\{K \in X : K \subset A\},$$

  i.e., the resolution of the partition X over some set A is the number of partition elements that A contains.
- If $U = \{u_i\}_{i=0}^{N}$ is a vector of values defined on elements of the partition X, then it shall be notated as $U(X)$. For compatibility with approximations given from finite volume schemes, in the following we shall identify any vector of function values $\{u_i\}_{i=0}^{N}$ evaluated on partition nodes $\{x_i\}_{i=0}^{N}$, i.e., $U = U(X)$, with the locally constant function U, defined on $[a,b]$ by

$$U = \sum_{i=0}^{N} u_i \ \chi_{[x_{i-1}^+, x_i^+)} + u_N \ \chi_{\{x_N^+\}}, \tag{2.1}$$

  with χ_A denoting the characteristic function of the set A, and $\{x_i^+\}_{i=-1}^{N}$ denoting the points

$$x_{-1}^+ = x_0, \quad x_i^+ = x_{i+\frac{1}{2}}, \quad i = 0, 1, \ldots, N-1, \quad x_N^+ = x_N.$$

In the case of evolution PDEs, where the numerical solution is constructed like a sequence of spatial approximations of solutions instances, the redistribution process could be applied before every evolution step. Let **Solver** denote a (general) finite volume scheme for solving some evolution PDE. For $n = 1, 2, \ldots$ **Solver** gives sequentially the approximations U^n of the solution instances $u(t^n, \cdot)$ at given time moments $t^0 < t^1 < \cdots < t^n \in \mathbb{R}^+$, starting at $t^0 = 0$. The given initial data U^0 is defined on the uniform partition X^0 of the domain. In the case of uniform partition, the evolution step can be represented for $n = 1, 2, \ldots$ as an equation on the vector space $\mathbb{R}^{2(N+1)}$:

$$(X^n,\ U^n(X^n)) = (X^{n-1},\ \mathbf{Solver}(X^{n-1},\ U^{n-1}(X^{n-1}))).$$

On meshes generated by the AGR process, the evolution step can be represented by the system of vector equations:

$$\begin{aligned} (\tilde{X},\ \tilde{U}(\tilde{X})) &= \mathbf{AGR}(X^{n-1},\ U^{n-1}(X^{n-1})), \\ (X^n,\ U^n(X^n)) &= (\tilde{X},\ \mathbf{Solver}(\tilde{X},\ \tilde{U}(\tilde{X}))), \end{aligned} \tag{2.2}$$

where $\tilde{X}, \tilde{U}$ are temporal vectors. Observe that in the case where the AGR process returns the same vectors, i.e., the AGR coincides with the identity linear transformation, then (2.2) reduces to the uniform evolution step. The AGR procedure, for a

given vector of values $U = \{u_i\}_{i=0}^{N}$ defined on the nodes $\{x_i\}_{i=0}^{N}$ of the partition X, i.e., $U = U(X)$, is described by the following two sequential steps,

$$\begin{aligned} \tilde{X} &= \mathbf{GMesh}(X,\ U(X)), \\ \tilde{U}(\tilde{X}) &= \mathbf{Rec}(X,\ U(X),\ \tilde{X}). \end{aligned} \tag{2.3}$$

In what follows we describe in detail the two steps **GMesh**, **Rec** of our redistribution algorithm AGR.

2.1. The GMesh step. At the **GMesh** step of the AGR procedure (2.3), a new partition $\tilde{X}$ of spatial nodes $\{\tilde{x}_i\}_{i=0}^{N}$ is formed, with resolution controlled by selected characteristics of the numerical solution U. The step is accomplished in two phases.

At the [illegible] the selected characteristics are defined on the domain $[a, b]$ through a strictly positive functional g of the approximate solution, the [illegible]. Since we are mainly interested in increasing the resolution over areas with discontinuities, and taking into account the results in [3], in this work we will also use some power p of proper approximation of the solution's curvature defined by

$$(\mathrm{curv} \circ u)(x) = \frac{|u''(x)|}{(1 + u'(x)^2)^{\frac{3}{2}}}$$

as the estimator function for selecting the resolution's density of the new partition. For a given partition $\{x_i\}_{i=0}^{N}$, the solution's curvature at the node x_i can be approximated by

$$(\mathrm{curv}_h \circ u)(x_i) = \frac{\frac{2}{x_{i+1}-x_{i-1}} \left| \frac{u(x_i)-u(x_{i-1})}{x_i - x_{i-1}} - \frac{u(x_{i+1})-u(x_i)}{x_{i+1}-x_i} \right|}{\sqrt{1+\left(\frac{u(x_i)-u(x_{i-1})}{x_i-x_{i-1}}\right)^2}\sqrt{1+\left(\frac{u(x_{i+1})-u(x_i)}{x_{i+1}-x_i}\right)^2}\sqrt{1+\left(\frac{u(x_{i+1})-u(x_{i-1})}{x_{i+1}-x_{i-1}}\right)^2}},$$

which is the inverse outer radius of the plane points $A_j = (x_j, u(x_j))$, $j = i-1, i, i+1$, i.e.,

$$(\mathrm{curv}_h \circ u)(x_i) = 2\frac{\|(A_{i+1} - A_i) \times (A_i - A_{i-1})\|}{\|A_{i+1} - A_i\|\ \|A_{i+1} - A_{i-1}\|\ \|A_i - A_{i-1}\|},$$

where $\|\cdot\|$ denotes the Euclidean norm. Thus, at the discrete level, the values g_i of our estimator function on the node points $\{x_i\}_{i=0}^{N}$, are given by $g_0 = g_N = \delta^p$ and for $i = 1, \ldots, N-1$,

$$g_i = \left(\max\left\{ \delta,\ 2\frac{\|(A_{i+1} - A_i) \times (A_i - A_{i-1})\|}{\|A_{i+1} - A_i\|\ \|A_{i+1} - A_{i-1}\|\ \|A_i - A_{i-1}\|} \right\} \right)^p, \tag{2.4}$$

where the power p is left as a free parameter taking values in the range $[0, 1]$ and δ is a very small positive number which ensures that g_i is strictly positive. The constant δ must be chosen such that, at a computational level, g_i behaves monotonically as a function of p even for negligible values. In our experiments, where the calculations were performed with ANSI-C computational libraries of double precision, δ was fixed to the value 10^{-32}. In the case of vector solution $u = (u_1, \ldots, u_M)^{\mathrm{T}}$, we intend to use the average of the coordinate estimators as the uniform estimator function for all the coordinates of the solution. Therefore, in general, instead of the values g_i we keep the normalized version $g_i / \sum_{j=0}^{N} g_j$, $i = 0, 1, \ldots, N$.

At the [illegible], the partition $\tilde{X}$ is actually produced through $g_h \circ U$, the corresponding function of type (2.1) to values (2.4). Since we expect a new mesh $\tilde{X}$ with resolution that locally follows the function $g_h \circ U$, we conclude that the resolution measure must be distributed in space like the measure $G_U(A) = \int_A (g_h \circ U)\, d\mu$ introduced by the positive function $g_h \circ U$, at least for sets from $\sigma(\tilde{X})$. Therefore the new partition $\tilde{X}$ is defined such that

$$\forall A \in \sigma(\tilde{X}) \quad G_U(A) = C\ \mathbf{resolution}(A),\ \text{with } C = \frac{G_U([a,b])}{N}. \tag{2.5}$$

Applying sequentially (2.5) with the sets $A_i = [a, \tilde{x}_i]$, $i = 1, 2, \ldots, N$, where their indices declare also the resolution of the partition $\tilde{X}$ over them, we induce, as it is called in the literature of moving meshes, the [illegible]:

(2.6)

$$\tilde{x}_0 = a \text{ and for } i = 1, \ldots, N,\ \tilde{x}_i \text{ is such that } \int_a^{\tilde{x}_i} g_h(U(x))\, dx = \frac{i}{N} \int_a^b g_h(U(x))\, dx.$$

In the present work, the new mesh $\tilde{X}$ is defined through (2.6), but we note that other inverting algorithms could also be constructed through a more elegant treatment of the generic equidistribution principal (2.5). Let $G(x) = G_U([a,x])$ be the distribution function, which is an increasing local linear function defined on the grid X, since $g_h \circ U$ is a positive function of type (2.1). Thus, G is completely defined from the values $G_i = G(x_i)$ of the G_U measure of the intervals $[a, x_i]$, that is, $G_0 = 0$, and for $i = 1, \ldots, N$,

$$\begin{aligned} G_i = G_U([a,x_i]) &= \int_{[a,x_i]} (g_h \circ U)\, d\mu = G_{i-1} + \int_{x_{i-1}}^{x_i} g_h(U(x))\, dx \\ &\overset{(2.1)}{=} G_{i-1} + (g_{i-1} + g_i)(x_i - x_{i-1})/2. \end{aligned} \tag{2.7}$$

From (2.6) it follows that the $\tilde{x}_i$ node is given by inverting the equation $G(\tilde{x}_i) = \frac{i}{N} G(\tilde{x}_N)$, so the nodes of the new mesh $\tilde{X}$ are $\tilde{x}_0 = a$, $\tilde{x}_N = b$, and, for $i = 1, \ldots, N-1$, $k_0 := 0$,

(2.8)

$$\tilde{G}_i = \frac{i}{N} G_N, \quad k_i = \max_{k_{i-1} \le \ell \le N} \{\ell : G_\ell \le \tilde{G}_i\}, \quad \tilde{x}_i = x_{k_i} + \frac{x_{k_i+1} - x_{k_i}}{G_{k_i+1} - G_{k_i}} (\tilde{G}_i - G_{k_i}).$$

2.2. The Rec step. At the **Rec** step of the AGR procedure (2.3), a new numerical solution $\tilde{U} = \{\tilde{u}_i\}_{i=0}^N$ of type (2.1) is defined on the new grid $\tilde{X}$ by reconstructing U from the old grid X to the new one. Since the schemes used in this work produce conservative solutions, the reconstruction step is done so that it is locally conservative on each interval $[\tilde{x}_{i-1}^+, \tilde{x}_i^+]$ of the new mesh. That is, for $i = 0, \ldots, N$, we require

$$\int_{\tilde{x}_{i-1}^+}^{\tilde{x}_i^+} \tilde{U}(x)\, dx = \int_{\tilde{x}_{i-1}^+}^{\tilde{x}_i^+} U(x)\, dx.$$

Since the functions $U, \tilde{U}$ are of type (2.1) defined on the partitions $X, \tilde{X}$, respectively,

$$\tilde{u}_i(\tilde{x}_i^+ - \tilde{x}_{i-1}^+) = \int_{\tilde{x}_{i-1}^+}^{\tilde{x}_i^+} U(x)\, dx = \int_{x_{k_i}^+}^{\tilde{x}_i^+} U(x)\, dx + \int_{x_{k_{i-1}}^+}^{x_{k_i}^+} U(x)\, dx - \int_{x_{k_{i-1}}^+}^{\tilde{x}_{i-1}^+} U(x)\, dx,$$

where, using the real line ordering of the domain $[a,b]$, $x^+_{k_i}$ denotes the larger node from the old partition not exceeding the node $\tilde{x}^+_i$ of the new partition. Consequently, the new values $\tilde{U}$ are $\tilde{u}_0 = u_0$ and, for $i = 1, \ldots, N$, $k_0 := 0$,

(2.9)
$$k_i = \max_{k_{i-1} \le \ell \le N} \{\ell : x^+_\ell \le \tilde{x}^+_i\},$$

$$\tilde{u}_i = \left\{ (\tilde{x}^+_i - x^+_{k_i}) u_{k_i+1} + \sum_{\ell=k_{i-1}+1}^{k_i} (x^+_\ell - x^+_{\ell-1}) u_\ell - (\tilde{x}^+_{i-1} - x^+_{k_{i-1}}) u_{k_{i-1}+1} \right\} / \big(\tilde{x}^+_i - \tilde{x}^+_{i-1}\big).$$

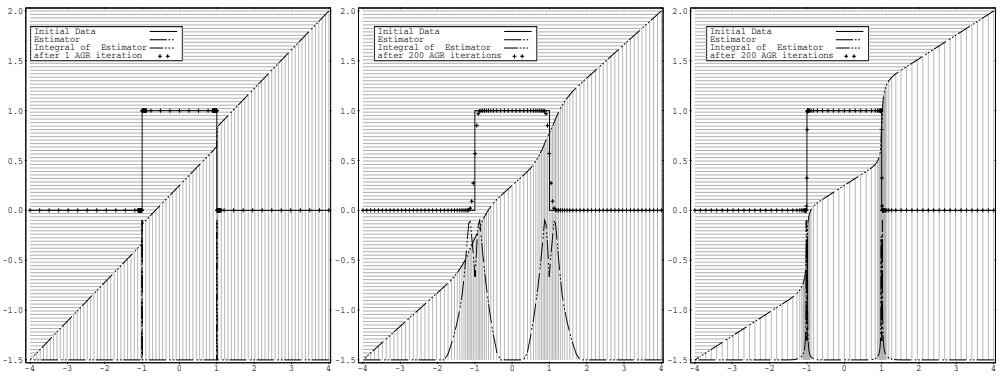


FIG. 1. *Iteration of the AGR procedure applied on Riemann data. After* 1 *iteration (left) and after* 200 *iterations for* $p = 0.023$ *(middle) and* $p = 0.1$ *(right).*

The **Gmesh** step of the AGR process can be graphically represented as in Figure 1. Starting with some initial data (solid line) we calculate the estimator function (dotted line) through (2.4) and then the corresponding distribution function (increasing dotted line) using (2.7). The proposed mesh (x position of the vertical lines, under the distribution function) is given by inverting a uniform mesh (y position of the horizontal lines, over the distribution function) through the distribution function using (2.8). Notice that y-coordinates are valid only for the data while either the estimator or its distribution function was scaled vertically to fit on the viewing graph area.

. . . . 1 (the action of the parameter p). Parameter p controls essentially the maximum density of the generated mesh $\tilde{X}$. Indeed, from relations (2.4), (2.6), we conclude that the corresponding nodes $\tilde{x}_i$, $i = 1, \ldots, N$, satisfy the equation

$$\frac{1}{N} \int_a^b \mathrm{curv}_h^p(U(x))\, dx = \int_{\tilde{x}_{i-1}}^{\tilde{x}_i} \mathrm{curv}_h^p(U(x))\, dx \le \max_{x \in [\tilde{x}_{i-1}, \tilde{x}_i]} \{\mathrm{curv}_h^p(U(x))\}\ (\tilde{x}_i - \tilde{x}_{i-1}),$$

and therefore

$$\frac{1}{N} \left(\frac{||\mathrm{curv}_h \circ U||_{L^p[a,b]}}{||\mathrm{curv}_h \circ U||_{L^\infty[a,b]}} \right)^p \le \min_i \{\tilde{x}_i - \tilde{x}_{i-1}\}.$$

Thus, choosing p appropriately, the resolution of the $\tilde{X}$ mesh respects an upper given bound. The action of the parameter p in the AGR procedure can also be seen in

Figure 1 (middle, right). For $p = 0$ observe from (2.7) that $G_i = x_i - x_0$, so the proposed partition from (2.8) coincides with the uniform partition of $[a, b]$. As the value of p increases, the proposed partition becomes more dense on the discontinuity areas, but over some value of p the process becomes unreliable because the partition lacks nodes over smooth areas. In our experiments with finite volume schemes an appropriate range for the parameter p was $[0.02, 0.10]$ for first order and $[0.04, 0.14]$ for second order schemes.

. . . . 2 (smoothing effect of the AGR process). Results in [3] show that the reconstructed function $\tilde{U}$ of the AGR procedure is more diffusive than the initial function U. Indeed this diffusive behavior can be observed by applying the AGR procedure iteratively on some initial data, i.e., starting with some initial data U^0 on the uniform partition X^0 of $[a, b]$, we define the sequence $(X^n, U^n(X^n)) = \mathbf{AGR}(X^{n-1}, U^{n-1}(X^{n-1}))$, $n = 1, 2, \ldots$. Experiments with Riemann initial data show that the above sequence attains a limit pair $(X, U(X))$. In Figure 1 (middle, right) we present the results after 200 iterations for $p = 0.023$ (middle) and for $p = 0.1$ (right). In Figure 2 (middle, right) we present the trajectories of the nodes for the above iterative process. Notice that after about 20 iterations for $p = 0.023$ and about 80 for $p = 0.1$ the position of all the nodes X stabilizes. But from (2.9) one can observe that reconstructing any function on the same grid leads to the same function, i.e., $U(X) = \mathbf{Rec}(X, U(X), X)$, so we conclude that also the reconstructed values $U(X)$ after the above iterations became constant. Figure 2 (left) shows the smoothing effect of the AGR process on the initial data after 200 iterations and for various values of p.

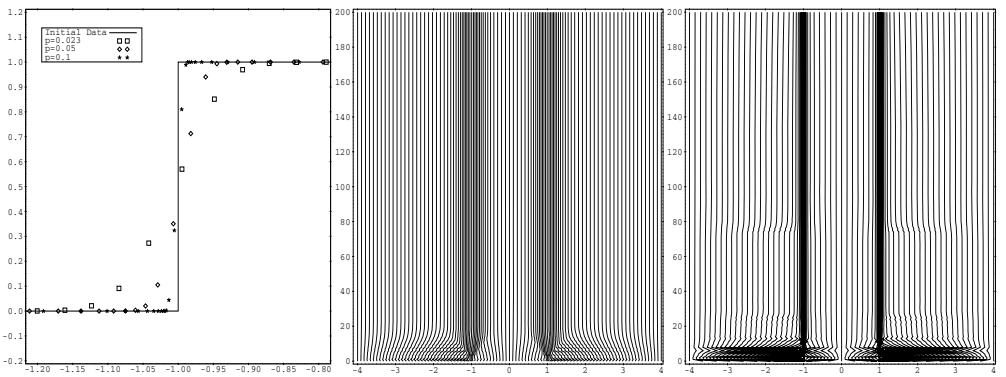


FIG. 2. *Focus in the shock area (left) after* 200 *iterations of the AGR procedure and for various values of parameter p. Trajectories of mesh nodes for* $p = 0.023$ *(middle) and for* $p = 0.1$ *(right).*

. . . . 3 (complexity of the AGR process). Observe that both the **Gmesh** step (2.7), (2.8) and the **Rec** step (2.9) of the AGR process are of linear complexity.

2.3. Redistribution in the steady state regime. The diffusive behavior of the AGR process might be one of the reasons that second order schemes, when combined with the redistribution step, produce nonoscillatory approximations; however, in the case where the current and the new mesh are almost the same the redistribution step should be avoided. In addition, in the case of very sensitive CL problems with steady states we noticed that the repeated application of the AGR process prevents the conservation of the steady states. Taking into account these observations, in the numerical experiments of this work we adopt a new version of the AGR process:

$$\begin{aligned} &\tilde{X} = \mathbf{GMesh}(X, U), \\ &\text{if } (|X - \tilde{X}| > D), \quad \tilde{U} = \mathbf{Rec}(X, U, \tilde{X}), \\ &\text{else} \qquad\qquad\quad \tilde{X} = X,\ \tilde{U} = U, \end{aligned}$$

where D is a "cutoff" level for the relative mean displacement $|X - \tilde{X}|$ between the current and the proposed meshes, under which the **Rec** step is avoided. The relative mean displacement is measured in the ℓ^1 vector norm, that is, $|X - \tilde{X}| = \frac{\frac{1}{N+1}\sum_{i=0}^{N}|x_i - \tilde{x}_i|}{b-a}$. Experiments show that, for first order schemes or those with extra stabilization mechanisms (like using slope/flux limiters), the AGR process produces results with higher resolution when the cutoff parameter D is of order 10^{-2}, whereas, due to the oscillatory behavior of second order schemes like the classical Lax–Wendroff, the cutoff level can be of order 10^{-3} or even less. In all cases, when the problem imposes steady states a nonzero value of D is crucial for the solving process in order to reach the time invariant solution for all schemes. The above observations become more pronounced for systems of conservation laws.

3. Numerical schemes on nonuniform grids (one-dimensional scalar equations). In this section we review some well-known classical numerical schemes, implemented at the **Solver** step, for the discretization of a scalar conservation law,

$$u_t + f(u)_x = 0, \tag{3.1}$$

and present them in their nonuniform grid formulation. We will also state some well-known disadvantages that these schemes exhibit.

We will approximate the solution $u(t, x)$, $x \in \mathbb{R}$, $t \geq 0$, of (3.1) by the discrete values u_i^n, $i \in \mathbb{Z}$, $n \in \mathbb{N}$, and in order to do so we consider a grid of points $x_{i+\frac{1}{2}}$ and define the computational cells and their lengths as

$$C_i = [x_{i-\frac{1}{2}}, x_{i+\frac{1}{2}}], \qquad \Delta x_i = x_{i+\frac{1}{2}} - x_{i-\frac{1}{2}} > 0.$$

We also denote by $x_i = (x_{i-\frac{1}{2}} + x_{i+\frac{1}{2}})/2$ the centers of the partition cells. The values u_i^n will be approximations of the averages of the exact solution over the cell

$$u_i^n \approx \frac{1}{\Delta x_i} \int_{C_i} u(t^n, x) dx,$$

with t^n the discrete time levels.

For consistency, we present the schemes in the usual conservative formulation with explicit time stepping,

$$u_i^{n+1} = u_i^n - \frac{\Delta t}{\Delta x_i} \left(f_{i+\frac{1}{2}}^n - f_{i-\frac{1}{2}}^n \right), \tag{3.2}$$

where $f_{i+\frac{1}{2}}^n$, $i = 1, 2, \ldots, N-1$, is the numerical flux function that determines the scheme.

. The numerical flux of the well-known first order upwind Roe scheme is given by (see [15])

$$f_{i+\frac{1}{2}}^{\mathrm{ROE}} = \frac{1}{2} \left(f_{i+1} + f_i - |a_{i+\frac{1}{2}}| \Delta u_{i+\frac{1}{2}} \right), \tag{3.3}$$

with $f_i = f(u_i)$, $\Delta u_{i+\frac{1}{2}} = u_{i+1} - u_i$ and $|a_{i+\frac{1}{2}}|$ the characteristic speed defined by

$$a_{i+\frac{1}{2}} = \begin{cases} \dfrac{f_{i+1} - f_i}{u_{i+1} - u_i}, & u_{i+1} \neq u_i, \\ a_i = \dfrac{\partial f}{\partial u}, & u_{i+1} = u_i. \end{cases} \tag{3.4}$$

A well-known drawback of Roe's scheme is that it may resolve nonphysical solutions by admitting stationary entropy-violating expansion shocks. Various [illegible] have been proposed; see [18]. For example, we can replace the moduli of a in (3.3) by

$$\psi(a) = \max(\delta, |a|)$$

or by the smoother form

$$\psi(a) = \begin{cases} |a|, & |a| \geq \delta, \\ (a^2 + \delta^2)/2\delta, & |a| < \delta, \end{cases}$$

with

$$\delta_{i+\frac{1}{2}} = \max\{0, a_{i+\frac{1}{2}} - a_i, a_{i+1} - a_{i+\frac{1}{2}}\}, \quad \delta_{i-\frac{1}{2}} = \max\{0, a_{i-\frac{1}{2}} - a_{i-1}, a_i - a_{i-\frac{1}{2}}\}.$$

The Roe numerical flux is then defined as

$$f^{\mathrm{ROE}}_{i+\frac{1}{2}} = \frac{1}{2}\left(f_{i+1} + f_i - \psi(a_{i+\frac{1}{2}})\Delta u_{i+\frac{1}{2}}\right). \tag{3.5}$$

In our numerical examples to follow we will see that when the adaptive mechanism is applied, the above entropy corrections are not necessary for the scheme to produce approximations of entropy solutions. This is a first order diffusive scheme, but it serves as a key ingredient in developing higher order methods.

[illegible]. A well-known first order scheme that produces approximations of entropy solutions is the local Lax–Friedrichs (LF) scheme. Its numerical flux function (see [18]) is given by

$$f^{\mathrm{LF}}_{i+\frac{1}{2}} = \frac{1}{2}\left(f_{i+1} + f_i - a_{max}\Delta u_{i+\frac{1}{2}}\right), \tag{3.6}$$

where $a_{max} = \max\{|f'(u_i)|, f'(u_{i+1})|\}$. This scheme produces diffusive numerical solutions and smears discontinuities and was used for comparison reasons only.

[illegible]. A natural generalization of the classical second order Lax–Wendroff (LW) scheme for nonuniform grids can be given as a combination of the first order Roe flux plus a correction flux (see [18]):

$$f^{\mathrm{LW}}_{i+\frac{1}{2}} = f^{\mathrm{ROE}}_{i+\frac{1}{2}} + \frac{1}{2}\left(\frac{\Delta x_i}{0.5\,(\Delta x_{i+1} + \Delta x_i)} - \frac{\Delta t}{0.5\,(\Delta x_{i+1} + \Delta x_i)}|a_{i+\frac{1}{2}}|\right)|a_{i+\frac{1}{2}}|\Delta u_{i+\frac{1}{2}}, \tag{3.7}$$

with $a_{i+\frac{1}{2}}$ as defined in (3.4). It is easy to see that for a uniform grid numerical flux (3.7) is reduced to the classical LW numerical flux. As for uniform grids this scheme has highly oscillatory behavior near discontinuities and without entropy correction cannot produce entropy solutions.

. One of the most classical second order schemes (see [15]) is the well-known two-step MacCormack scheme:

$$u_i^{(1)} = u_i^n - \frac{\Delta t}{\Delta x_i}\left(f_{i+1}^n - f_i^n\right), \tag{3.8}$$

$$u_i^{(2)} = u_i^n - \frac{\Delta t}{\Delta x_i}\left(f_{i+1}^{(1)} - f_i^{(1)}\right), \tag{3.9}$$

$$u_i^{n+1} = \frac{1}{2}\left(u_i^{(1)} + u_i^{(2)}\right). \tag{3.10}$$

This simple scheme also exhibits oscillatory behavior near discontinuities and cannot produce the correct entropy solutions.

. Here we present a second order slope-limiting scheme based on the MUSCL-TVD interpolation formula on nonuniform grids; see, for example, [22]. We define

$$u_{i+\frac{1}{2}}^{\mathrm{L}} = u_i + \frac{h_{i+1}}{2}\left(\frac{u_i - u_{i-1}}{h_i}\right)\Phi\left(\theta_i\right),$$

$$u_{i+\frac{1}{2}}^{\mathrm{R}} = u_{i+1} - \frac{h_{i+1}}{2}\left(\frac{u_{i+2} - u_{i+1}}{h_{i+2}}\right)\Phi\left(\frac{1}{\theta_{i+1}}\right),$$

with $h_i = x_i - x_{i-1}$ and θ_i given by

$$\theta_i = \frac{\left(u_{i+1} - u_i\right)/h_{i+1}}{\left(u_i - u_{i-1}\right)/h_i}.$$

The Φ is a limiter function, and there are several options from which to choose Φ; see, for example, [25]. Some of the most popular limiters are the MinMod (MM) limiter,

$$\Phi(\theta) = \max(0, \min(1, \theta)),$$

the Superbee (SB) limiter

$$\Phi(\theta) = \max(0, \min(2\theta, 1), \min(\theta, 2)),$$

the VanLeer (VL) limiter

$$\Phi(\theta) = \frac{|\theta| + \theta}{1 + |\theta|},$$

and the monotonized central (MC) limiter

$$\Phi(\theta) = \max(0, \min((1+\theta)/2, 2, 2\theta)).$$

The numerical flux for the MUSCL scheme can then be expressed, based on the Roe flux (without an entropy fix), as

$$f_{i+\frac{1}{2}}^{\mathrm{MUSCL}} = \frac{1}{2}\left(f(u_{i+\frac{1}{2}}^{\mathrm{R}}) + f(u_{i+\frac{1}{2}}^{\mathrm{L}}) - |a_{i+\frac{1}{2}}|(u_{i+\frac{1}{2}}^{\mathrm{R}} - u_{i+\frac{1}{2}}^{\mathrm{L}})\right), \tag{3.11}$$

where $a_{i+\frac{1}{2}}$ is evaluated as before, with u_{i+1} and u_i replaced by $u_{i+\frac{1}{2}}^R$ and $u_{i+\frac{1}{2}}^L$. In a similar way one can define a MUSCL flux based on the LF flux.

Second order TVD schemes reduce to first order at extrema, and there are also differences in the behavior for each limiter; for example, the last three limiters have been shown to exhibit sharper resolution of discontinuities, since they do not reduce the slope as severely as MM near a discontinuity.

4. Numerical examples for scalar conservation laws. Because all the schemes presented above are explicit in time, stability requires the time step Δt to satisfy the CFL condition, as to ensure that the time step is small enough that waves from neighboring cells do not interact, with CFL ≤ 1. Based on that we use a variable time step calculated from

$$\text{CFL} = \Delta t \max_i \left\{ \left| \frac{a^+_{i-\frac{1}{2}}}{\Delta x_i} \right|, \left| \frac{a^-_{i+\frac{1}{2}}}{\Delta x_i} \right| \right\}, \tag{4.1}$$

with $a^+ = \max(0, a)$ and $a^- = \min(0, a)$.

4.1. Burgers equation. We first consider numerical experiments for the Burgers equation in the inviscid limit,

$$u_t + \left(\frac{u^2}{2} \right)_x = 0. \tag{4.2}$$

The analytical solutions to this problem can be found, for example, in [15].

The first problem is an academic test case with initial conditions, for $x \in [-5, 5]$,

$$u(x,0) = \begin{cases} -1.0, & x < 0, \\ 1.0, & x \geq 0, \end{cases} \tag{4.3}$$

and has the following exact solution, also known as a [illegible]:

$$u(x,0) = \begin{cases} -1.0, & x < -t, \\ x/t, & -t \leq x \leq t, \\ 1.0, & x > t. \end{cases} \tag{4.4}$$

As pointed out in [18], dealing with transonic rarefactions properly is an important component in the development of successful methods. Results for this problem are presented in Figures 3 and 4 at $t = 2$s. A grid of 61 points was used for all schemes with CFL number equal to 0.9. All the schemes presented in the previous section, with the exception of the LF scheme, cannot produce the correct entropy solution. By the application of the adaptive mechanism all the schemes calculate the correct solution. The most accurate results were produced with the MUSCL adaptive scheme using the SB limiter. We note here that when the adaptive mechanism was applied to the MUSCL scheme, we were able to produce similar results for all limiters.

The second problem was presented in [22] and has initial conditions given by

$$u(x,0) = \begin{cases} 1.0, & x \in [0.2, 2.0], \\ -0.5, & x \in (2.0, 3.0], \\ -1.0, & x \in (3.0, 4.8], \\ 0.0 & \text{otherwise.} \end{cases} \tag{4.5}$$

The solution domain is for $x \in [0, 5]$ with homogeneous Dirichlet conditions. This problem includes two shocks initially at $x = 2$ and $x = 3$, moving to the right and left, respectively, and two expansion discontinuities at $x = 0.2$ and $x = 4.8$ also expanding to the right and left, respectively. The two shocks collide at time $t = 1$s and form a single shock moving to the left. The numerical results are presented in Figures 5 and 6 for $t = 2$s when the shocks have combined into a single one. A grid of 121 points was used for all schemes with CFL number equal to 0.9. All the schemes

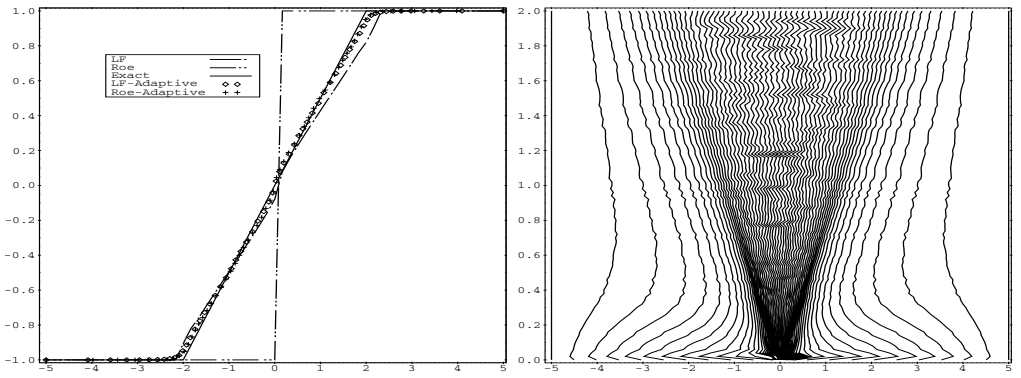


FIG. 3. *Transonic rarefaction problem: Numerical solution (left) and grid point trajectories (right) for the adaptive Roe scheme* ($p = 0.09$, $D = 0$).

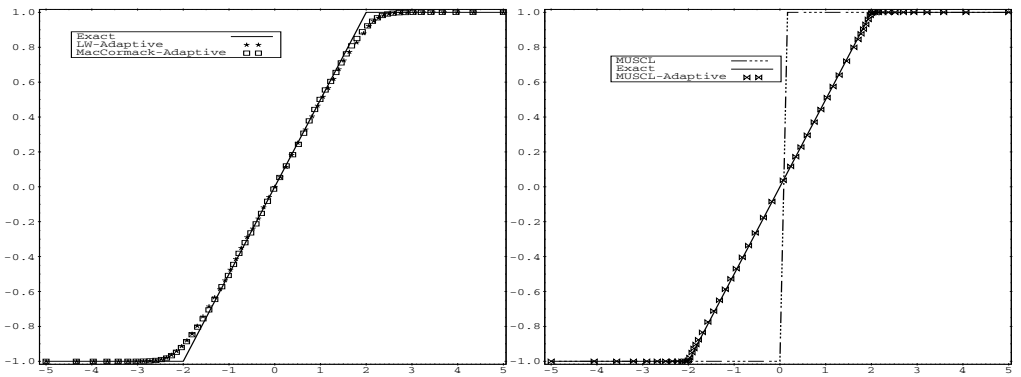


FIG. 4. *Transonic rarefaction problem: Numerical solution (left) for the LW and MacCormack schemes* ($p = 0.062$, $D = 0$) *and (right) for the MUSCL scheme* ($p = 0.10$, $D = 0$).

produce improved results when the adaptive mechanism is imposed, when compared to those produced in a uniform mesh. It is impressive that even the second order oscillatory LW and MacCormack schemes are now able to produce accurate solutions. The performance of the MUSCL using the MC limiter scheme is also greatly improved, since in the nonadaptive case the scheme fails to produce the entropy correct solution at expansions. Similar observations were made for the other limiters as well. In Figure 6 one can also see the ability of the mesh to capture and follow the evolution of the solution as demonstrated by the grid point trajectories.

The third problem has an initial profile of a smooth wave given by

$$u(x, 0) = 0.5 \sin(\pi x) + \sin(2\pi x), \qquad x \in [0, 1],$$

with periodic boundary conditions imposed. The solution propagates to the right, steepening until the singularity time $t_c = 64/(129\pi)$. Results are presented in Figures 7 and 8 for the second order schemes for a 61-point grid and with a CFL number of 0.6 at $t = 1$s, and verify the observations made for the second example. Considerably better results are obtained for the adaptive methods, compared to the ones in uniform grid, especially in shock resolution because of the clustering of grid points near the shock. Again the grid trajectories show the formation and following of the shock layer.

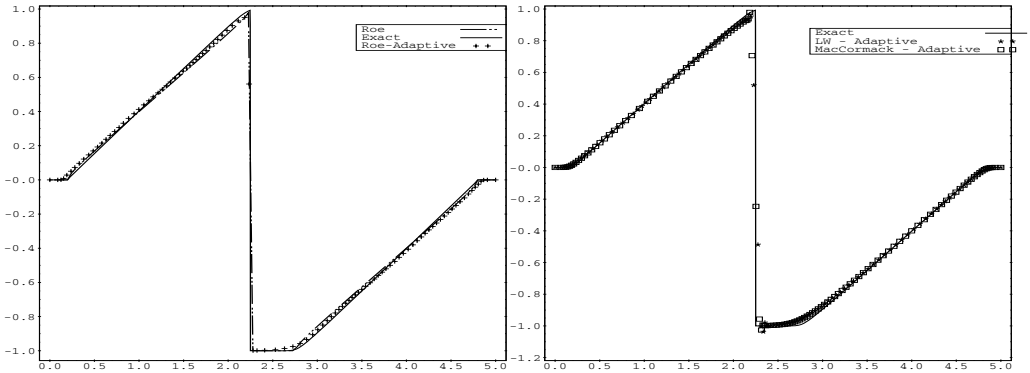


FIG. 5. *The second Burgers problem: Numerical solution Roe's scheme (left)* ($p = 0.079$, $D = 0.0224$) *and the LW and MacCormack schemes (right)* ($p = 0.10$, $D = 0$).

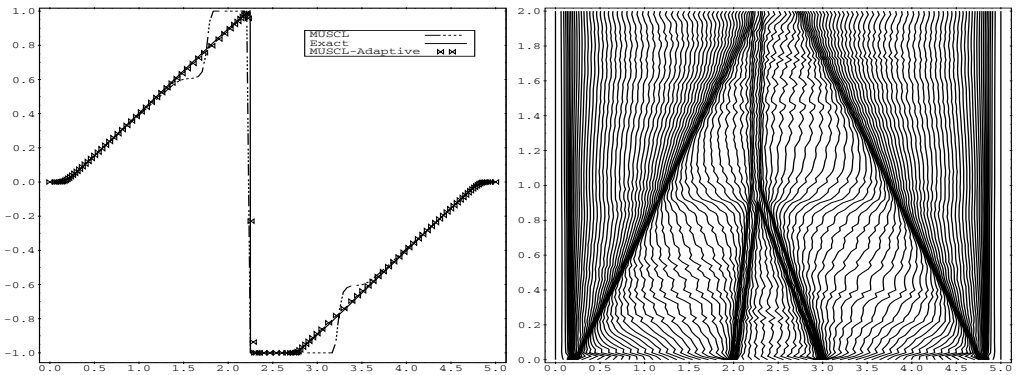


FIG. 6. *The second Burgers problem: Numerical solution (left) and grid point trajectories (right) for the adaptive MUSCL scheme* ($p = 0.09$, $D = 0$).

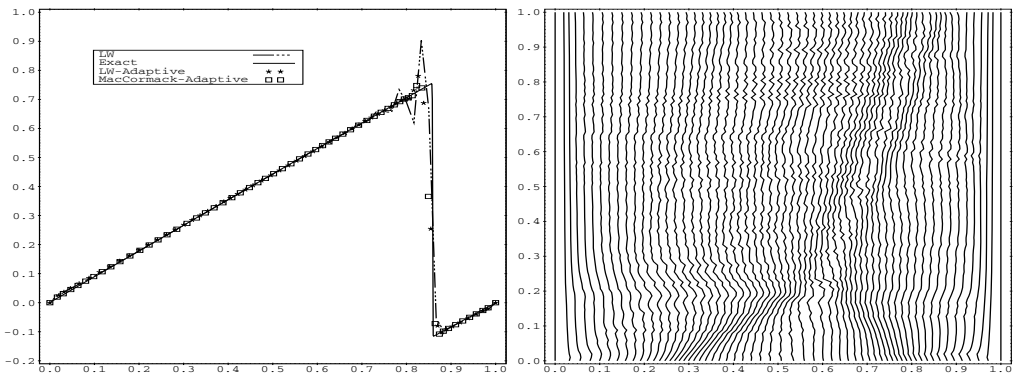


FIG. 7. *The third Burgers problem: Numerical solution for the LW and MacCormack schemes (left) and grid point trajectories (right) for the LW scheme* ($p = 0.06$, $D = 0$).

The fourth problem has initial condition

$$u(x, 0) = 1 - 0.02x, \qquad x \in [0, 100],$$

and has been selected to illustrate convergence to a discontinuous steady state. The

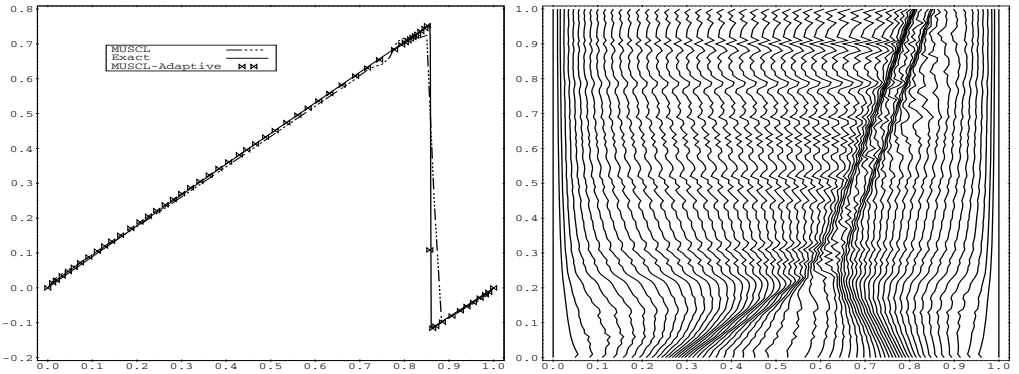


FIG. 8. *The third Burgers problem: Numerical solution (left) and grid point trajectories (right) for the MUSCL scheme* ($p = 0.118$, $D = 0$).

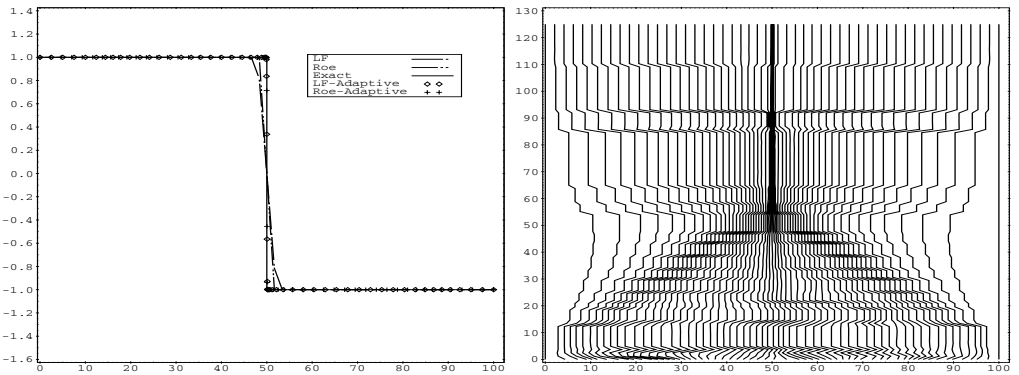


FIG. 9. *The fourth Burgers problem: Numerical solutions for the Roe and LF schemes (left) and grid point trajectories (right) for the adaptive Roe scheme* ($p = 0.10$, $D = 0.013$).

parameters for the calculations were 61 grid points and a CFL number equal to 0.9. The stabilizing tendency of the adaptive method can be clearly seen since it does not produce oscillatory results, as shown in Figures 9 and 10 for $t = 125$s. One can also observe the improvement in all the calculations when the adaptive mechanism is imposed. The effect in the grid movement of a zero and a nonzero value for the "cutoff" parameter D, for Roe's first order scheme, can be seen in Figure 11, following the remarks made in section 2.3.

4.2. Nonconvex conservation law and the Buckley–Leverett equation. In this section we first apply the adaptive grid method to the scalar conservation law with nonlinear nonconvex flux,

$$f(u) = (u^2 - 1)(u^2 - 4)/4 \quad \text{for} \quad x \in [0, 1]. \tag{4.6}$$

The initial profile is $u(x, 0) = -2\text{sign}(x)$. As pointed out in [29] solving this Riemann problem for a scalar conservation law with nonconvex flux leads to difficulties with some numerical schemes. Numerical solutions for all schemes are shown in Figure 12 at $t = 1.2$s, in a 101-point grid and for CFL number equal to 0.8. Again by imposing the adaptive mechanism we are able to produce quality results for all schemes.

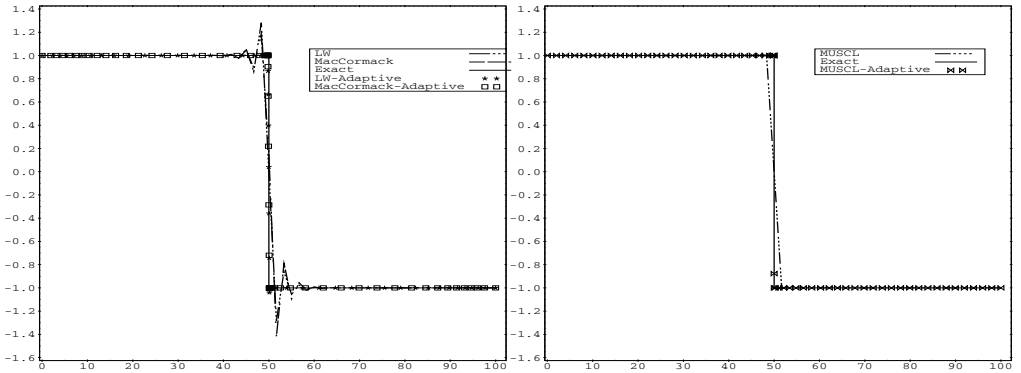


FIG. 10. *The fourth Burgers problem: Numerical solutions for the LW and MacCormack schemes (left) and MUSCL scheme (right)* ($p = 0.13$, $D = 0$).

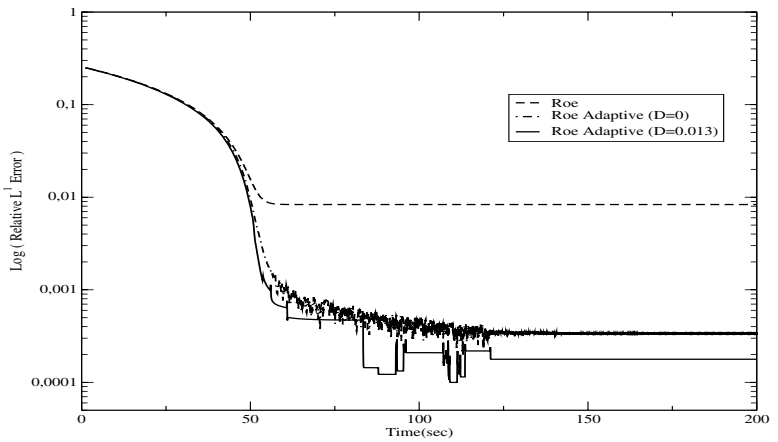


FIG. 11. *The fourth Burgers problem: convergence histories for the Roe scheme.*

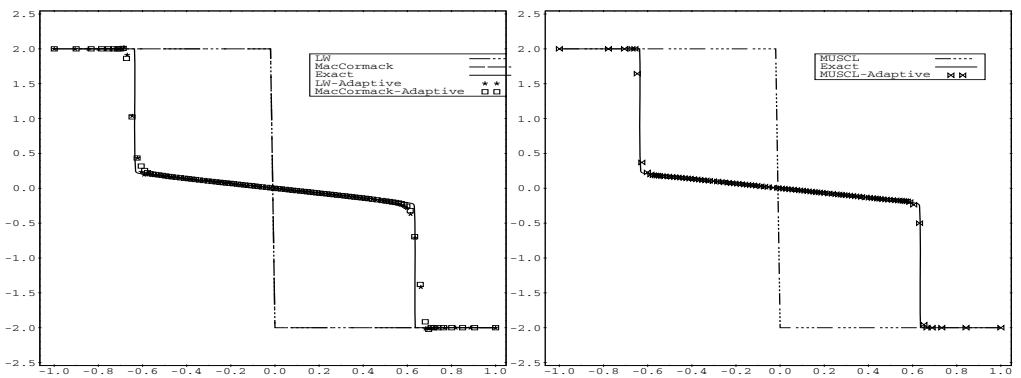


FIG. 12. *Nonconvex flux problem: Numerical solutions for the LW and MacCormack schemes (left) and MUSCL scheme (right)* ($p = 0.069$, $D = 0$).

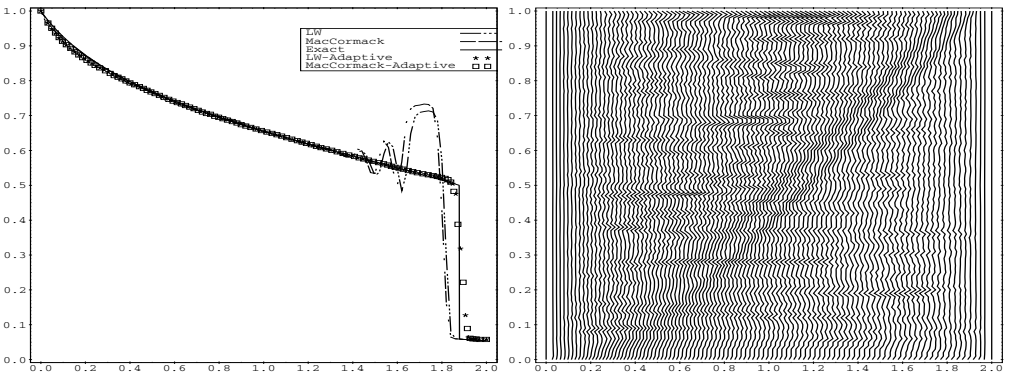


FIG. 13. *Buckley–Leverett problem: Numerical solutions for the LW and MacCormack schemes (left) and grid point trajectories for the adaptive LW scheme* ($p = 0.076$, $D = 0$).

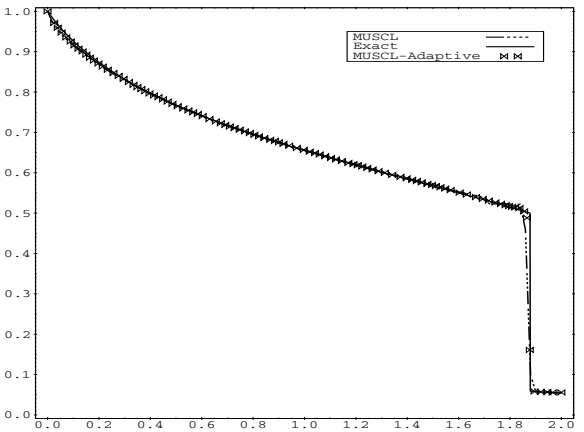


FIG. 14. *Buckley–Leverett problem: Numerical solutions for the MUSCL scheme* ($p = 0.10$, $D = 0.0088$).

Next we consider the scalar Buckley–Leverett equation with the flux function

$$f(u) = \frac{u^2}{u^2 + 0.5(1-u)^2},$$

with initial and boundary conditions for $x \in [0, 2]$,

$$u(x,0) = \frac{1}{1+10x}, \qquad u(0,t) = 1, \qquad u(1,t) = \frac{1}{21}.$$

The solution for the adaptive and uniform LW and MacCormack schemes is shown in Figure 13 in a 101-point grid. The highly oscillatory behavior of these schemes is totally suppressed by the adaptive mechanism resolving and following the shock layer much more accurately. Improvement in the shock resolution can also be observed for the MUSCL scheme in Figure 14.

. . . . 4 (a result on the smoothing effect of the AGR). As a final result for scalar CL we present a simple advection test (see [26]) of a contact discontinuity in order to measure the smoothing effect of the adaptive mechanism. We compute an

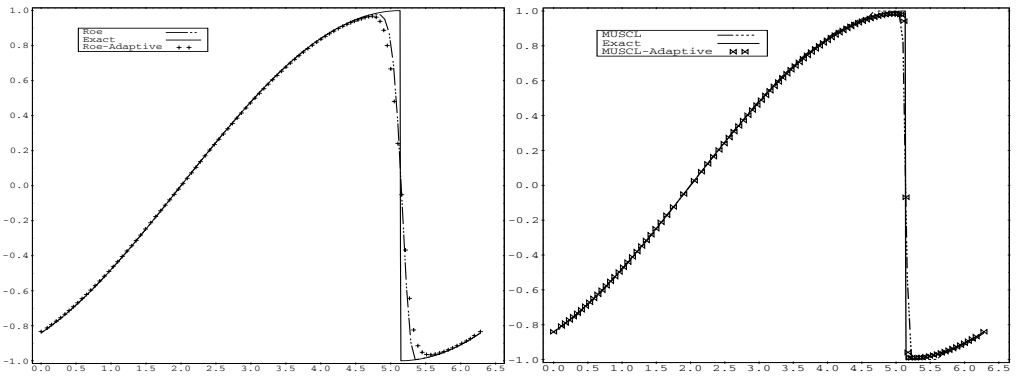


FIG. 15. *Linear advection of a contact discontinuity for the Roe scheme* ($p = 0.03$, $D = 0.004$) *and MUSCL scheme* ($p = 0.12$, $D = 0$).

approximate solution to the linear advection equation, $u_t + u_x = 0$, with periodic boundary conditions and initial condition

$$u(x,0) = \begin{cases} \sin(x/2), & x \in [0,\pi), \\ -\sin(x/2), & x \in [\pi, 2\pi), \end{cases}$$

that has a discontinuity at $x = \pi$ which at t = 2s is advected to $x = \pi + 2$. Results on a 101-point grid for the Roe and MUSCL schemes, with CFL = 0.95, are shown in Figure 15. The smoothing effect of the adaptive mechanism can be seen for the first order Roe scheme, but for the second order MUSCL scheme the results have been improved.

5. Numerical schemes for nonuniform grids (one-dimensional systems of equations). We consider the general system of conservation laws with a source term added, i.e.,

$$\mathbf{U}_t + \mathbf{F}(\mathbf{U})_x = \mathbf{G}(\mathbf{U}), \quad \mathbf{U} \in \mathbb{R}^M; \tag{5.1}$$

here $\mathbf{F}(\mathbf{U})$ is the flux function and $\mathbf{G}$ is the source term. We can numerically approximate system (5.1) by using the explicit conservative numerical scheme with a source term approximation within the finite difference-volume frame, as

$$\mathbf{U}_i^{n+1} = \mathbf{U}_i^n - \frac{\Delta t}{\Delta x_i}\left[\mathbf{F}^n_{i+\frac{1}{2}} - \mathbf{F}^n_{i-\frac{1}{2}}\right] + \frac{\Delta t}{\Delta x_i}\widetilde{\mathbf{G}}_i^n, \tag{5.2}$$

where again $\mathbf{F}^n_{i+\frac{1}{2}}$, $i = 1, 2, \ldots, N-1$, is the numerical flux function, and the approximation of the source term $\widetilde{\mathbf{G}}_i^n$ is considered as the cell average value (numerical source integral over the computational cell) satisfying the relationship

$$\widetilde{\mathbf{G}}_i^n = \frac{1}{\Delta x_i}\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \mathbf{G}(x, \mathbf{U}^n)dx, \tag{5.3}$$

and after choosing the numerical flux function it remains to choose an appropriate approximation (depending on the scheme used) for the numerical source integral $\widetilde{\mathbf{G}}_i^n$.

For the stationary case, $\mathbf{U}_t = 0$, the flux function and source term balance:

$$\mathbf{F}(\mathbf{U})_x = \mathbf{G}(\mathbf{U}).$$

Therefore, an accurate numerical scheme should also balance the numerical flux with the source term approximation,

$$\mathbf{F}^n_{i+\frac{1}{2}} - \mathbf{F}^n_{i-\frac{1}{2}} = \widetilde{\mathbf{G}}^n_i. \tag{5.4}$$

The numerical schemes presented next incorporate the numerical source integral in such a way that (5.4) is satisfied.

5.1. Roe's scheme for systems of conservation laws with source terms. The numerical flux for the Roe upwind scheme can be written in the form

$$\mathbf{F}^{\mathrm{ROE}}_{i+\frac{1}{2}} = \frac{1}{2}\left[\mathbf{F}_{i+1} + \mathbf{F}_i - \mathbf{J}_{i+\frac{1}{2}}(\mathbf{U}_{i+1} - \mathbf{U}_i)\right]. \tag{5.5}$$

The matrix $\mathbf{J}_{i+\frac{1}{2}}$ is the Roe-type linearization and satisfies

$$\mathbf{F}_{i+1} - \mathbf{F}_i = \mathbf{J}_{i+\frac{1}{2}}(\mathbf{U}_{i+1} - \mathbf{U}_i) \tag{5.6}$$

and has real eigenvalues $(\tilde{a}^k_{i+\frac{1}{2}})$, $k = 1, \ldots, M$, and a complete set of eigenvectors $(\tilde{\mathbf{e}}^k_{i+\frac{1}{2}})$. Representing now $\mathbf{U}_{i+1} - \mathbf{U}_i$ in terms of the eigenvectors of the Roe linearization, i.e.,

$$\mathbf{U}_{i+1} - \mathbf{U}_i = \sum_{k=1}^{M} \alpha^k_{i+\frac{1}{2}} \tilde{\mathbf{e}}^k_{i+\frac{1}{2}}, \tag{5.7}$$

we obtain from (5.6)

$$\mathbf{J}_{i+\frac{1}{2}}(\mathbf{U}_{i+1} - \mathbf{U}_i) = \sum_{k=1}^{M} \tilde{a}^k_{i+\frac{1}{2}} \alpha^k_{i+\frac{1}{2}} \tilde{\mathbf{e}}^k_{i+\frac{1}{2}}. \tag{5.8}$$

Then the Roe numerical flux can be written as

$$\mathbf{F}^{\mathrm{ROE}}_{i+\frac{1}{2}} = \frac{1}{2}\left[\mathbf{F}_{i+1} + \mathbf{F}_i - \sum_{k=1}^{M} |\tilde{a}^k_{i+\frac{1}{2}}| \alpha^k_{i+\frac{1}{2}} \tilde{\mathbf{e}}^k_{i+\frac{1}{2}}\right]. \tag{5.9}$$

The $\alpha_{i+\frac{1}{2}}$ (wave strengths) values follow from (5.7).

. The integral of $\mathbf{G}$, in relation (5.3), over the cell $[x_{i-\frac{1}{2}}, x_{i+\frac{1}{2}}]$ is first split into a sum of two integrals in the $[x_{i-\frac{1}{2}}, x_i]$, $[x_i, x_{i+\frac{1}{2}}]$ as

$$\frac{1}{\Delta x_i}\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \mathbf{G}(x, \mathbf{U}^n)dx = \frac{1}{\Delta x_i}\left[\int_{x_{i-\frac{1}{2}}}^{x_i} \mathbf{G}(x, \mathbf{U}^n)dx + \int_{x_i}^{x_{i+\frac{1}{2}}} \mathbf{G}(x, \mathbf{U}^n)dx\right]. \tag{5.10}$$

The upwind definition of the approximated source terms $\widetilde{\mathbf{G}}^n_i$ for nonuniform grids is given by

$$\widetilde{\mathbf{G}}^n_i = \frac{1}{\Delta x_i}\left[\frac{(x_i - x_{i-1})}{2}\widetilde{\mathbf{G}}^L(x_{i-1}, x_i, \mathbf{U}^n_{i-1}, \mathbf{U}^n_i) + \frac{(x_{i+1} - x_i)}{2}\widetilde{\mathbf{G}}^R(x_i, x_{i+1}, \mathbf{U}^n_i, \mathbf{U}^n_{i+1})\right], \tag{5.11}$$

where $\widetilde{\mathbf{G}}^L$ and $\widetilde{\mathbf{G}}^R$ are continuous functions, giving the averages of $\mathbf{G}(x, \mathbf{U}^n)$ on the subcells $[x_{i-\frac{1}{2}}, x_i]$, $[x_i, x_{i+\frac{1}{2}}]$, respectively. These functions give the upwind values of the source terms and following [7] can be defined as

$$\widetilde{\mathbf{G}}^L_{i-\frac{1}{2}} = \left[\mathbf{I} + |\mathbf{J}_{i-\frac{1}{2}}|\mathbf{J}^{-1}_{i-\frac{1}{2}}\right] \widetilde{\mathbf{G}}_{i-\frac{1}{2}}, \quad \widetilde{\mathbf{G}}^R_{i+\frac{1}{2}} = \left[\mathbf{I} - |\mathbf{J}_{i+\frac{1}{2}}|\mathbf{J}^{-1}_{i+\frac{1}{2}}\right] \widetilde{\mathbf{G}}_{i+\frac{1}{2}}. \tag{5.12}$$

The definition of the $\mathbf{J}_{i\pm\frac{1}{2}}$ and $|\mathbf{J}_{i\pm\frac{1}{2}}|$ matrices follow from (5.6) for the Roe linearization and are defined in the appendix. The terms $\widetilde{\mathbf{G}}_{i\pm\frac{1}{2}}$ represent approximations of $\mathbf{G}$ at cells $(i-1, i)$ and $(i, i+1)$, respectively.

Alternatively, we rewrite (5.12) in decomposed form,

$$\widetilde{\mathbf{G}}^L_{i-\frac{1}{2}} = \sum_{k=1}^{M} \beta^k_{i-\frac{1}{2}} \tilde{\mathbf{e}}^k_{i-\frac{1}{2}} \left(1 + \operatorname{sgn}(\tilde{a}^k_{i-\frac{1}{2}})\right), \quad \widetilde{\mathbf{G}}^R_{i+\frac{1}{2}} = \sum_{k=1}^{M} \beta^k_{i+\frac{1}{2}} \tilde{\mathbf{e}}^k_{i+\frac{1}{2}} \left(1 - \operatorname{sgn}(\tilde{a}^k_{i+\frac{1}{2}})\right), \tag{5.13}$$

with the values of $\beta^k_{i\pm\frac{1}{2}}$ determined from

$$\sum_{k=1}^{M} \beta^k_{i\pm\frac{1}{2}} \tilde{\mathbf{e}}^k_{i\pm\frac{1}{2}} = \widetilde{\mathbf{G}}_{i\pm\frac{1}{2}}. \tag{5.14}$$

5.2. The LW scheme. The numerical flux for the classical second order LW scheme for nonlinear systems, on nonuniform grids, can be written, following [18], as

$$\mathbf{F}^{\mathrm{LW}}_{i+\frac{1}{2}} = \mathbf{F}^{\mathrm{ROE}}_{i+\frac{1}{2}} + \frac{1}{2} \sum_{k=1}^{M} \left(\frac{\Delta x_i}{\Delta x_{i+\frac{1}{2}}} - \frac{\Delta t}{\Delta x_{i+\frac{1}{2}}} |\tilde{a}^k_{i+\frac{1}{2}}| \right) |\tilde{a}^k_{i+\frac{1}{2}}| \alpha^k_{i+\frac{1}{2}} \tilde{\mathbf{e}}^k_{i+\frac{1}{2}}, \tag{5.15}$$

where $\Delta x_{i+\frac{1}{2}} = (\Delta x_{i+1} + \Delta x_i)/2$.

A flux-limited second order TVD version of the LW scheme is then given by

$$\mathbf{F}^{\mathrm{LWTVD}}_{i+\frac{1}{2}} = \mathbf{F}^{\mathrm{ROE}}_{i+\frac{1}{2}} + \frac{1}{2} \sum_{k=1}^{M} \left(\frac{\Delta x_i}{\Delta x_{i+\frac{1}{2}}} - \frac{\Delta t}{\Delta x_{i+\frac{1}{2}}} |\tilde{a}^k_{i+\frac{1}{2}}| \right) \Phi(\theta^k_{i+\frac{1}{2}}) |\tilde{a}^k_{i+\frac{1}{2}}| \alpha^k_{i+\frac{1}{2}} \tilde{\mathbf{e}}^k_{i+\frac{1}{2}}, \tag{5.16}$$

where

$$\theta^k_{i+\frac{1}{2}} = \frac{\alpha^k_{I+\frac{1}{2}}}{\alpha^k_{i+\frac{1}{2}}} \qquad \text{with} \qquad I = \begin{cases} i-1 & \text{if} \quad \tilde{a}^k_{i+\frac{1}{2}} > 0, \\ i+1 & \text{if} \quad \tilde{a}^k_{i+\frac{1}{2}} < 0; \end{cases} \tag{5.17}$$

here Φ can be any of the limiter functions presented in section 3.

The Roe averaged values can be used for the source term approximations, and we obtain a flux-limited second order approximation of the source term in the numerical source term (5.10), similar to the one presented in [16], by setting

$$\widetilde{\mathbf{G}}^L_{i-\frac{1}{2}} = \sum_{k=1}^{M} \beta^k_{i-\frac{1}{2}} \tilde{\mathbf{e}}^k_{i-\frac{1}{2}} \left[1 + \operatorname{sgn}(\tilde{a}^k_{i-\frac{1}{2}}) \left(1 - \Phi(\theta^k_{i-\frac{1}{2}}) \left(\frac{\Delta x_i}{\Delta x_{i-\frac{1}{2}}} - \frac{\Delta t}{\Delta x_{i-\frac{1}{2}}} |\tilde{a}^k_{i-\frac{1}{2}}| \right)\right)\right],$$

$$\widetilde{\mathbf{G}}^R_{i+\frac{1}{2}} = \sum_{k=1}^{M} \beta^k_{i+\frac{1}{2}} \tilde{\mathbf{e}}^k_{i+\frac{1}{2}} \left[1 - \operatorname{sgn}(\tilde{a}^k_{i+\frac{1}{2}}) \left(1 - \Phi(\theta^k_{i+\frac{1}{2}}) \left(\frac{\Delta x_i}{\Delta x_{i+\frac{1}{2}}} - \frac{\Delta t}{\Delta x_{i+\frac{1}{2}}} |\tilde{a}^k_{i+\frac{1}{2}}| \right)\right)\right].$$

5.3. The MacCormack scheme. The MacCormack scheme adapted to approximate systems of conservation laws with a source term can be written as

$$\mathbf{U}_i^{n+1} = \frac{1}{2}\left(\mathbf{U}_i^n + \mathbf{U}_i^{(1)}\right) - \frac{\Delta t}{2\Delta x_i}\left(\mathbf{F}_i^{(1)} - \mathbf{F}_{i-1}^{(1)}\right) + \frac{\Delta t}{2\Delta x_i}\widetilde{\mathbf{G}}_{i-\frac{1}{2}}^{(1)}, \tag{5.18}$$

where

$$\mathbf{U}_i^{(1)} = \mathbf{U}_i^n - \frac{\Delta t}{\Delta x_i}\left(\mathbf{F}_{i+1}^n - \mathbf{F}_i^n\right) + \frac{\Delta t}{\Delta x_i}\widetilde{\mathbf{G}}_{i+\frac{1}{2}}^n.$$

This scheme is a predictor-corrector one and has the advantage that we do not need to approximate the eigenvalues and eigenvectors of the Jacobian matrix. However, spurious oscillations may occur in the solution, especially when discontinuities are present, even when we choose the $\widetilde{\mathbf{G}}$ discretizations in (5.18) in such a way as to satisfy (5.4).

6. The shallow water model. We consider the well-known one-dimensional shallow water system, with a geometrical source term (the bottom topography) added, written in differential conservation law form as a single vector equation:

$$\mathbf{U}_t + \mathbf{F}(\mathbf{U})_x = \mathbf{G}(\mathbf{U}), \tag{6.1}$$

with

$$\mathbf{U} = \begin{bmatrix} h \\ hu \end{bmatrix}, \qquad \mathbf{F}(\mathbf{U}) = \begin{bmatrix} hu \\ hu^2 + \frac{g}{2}h^2 \end{bmatrix}, \qquad \mathbf{G}(\mathbf{U}) = \begin{bmatrix} 0 \\ -ghZ' \end{bmatrix}.$$

System (6.1) describes the flow at time $t \geq 0$ at point $x \in \mathbb{R}$, where $h(x,t) \geq 0$ is the total water height above the bottom, $u(x,t)$ is the average horizontal velocity, $Z(x)$ is the bottom height function, and g the gravitational acceleration. In the following we will denote by $q = hu$ the water unit discharge.

An important property of system (6.1) is related to the source term: the shallow water system admits nontrivial steady states. They are characterized by

$$(hu)_x = 0, \tag{6.2}$$

$$\left(hu^2 + \frac{g}{2}h^2\right)_x = ghZ', \tag{6.3}$$

i.e.,

$$q = (hu) = \text{constant}, \tag{6.4}$$

$$hu^2 + g(h+Z) = \text{constant}. \tag{6.5}$$

A particular case that provides a benchmark for many approximating schemes is the still water steady state (flow at rest), i.e., when $u = 0$ and $h + Z = \text{constant}$.

In [7] the concept of Property C was introduced. A given scheme would satisfy Property C if, in the case of a flow at rest, there is an exact balance between the discrete components of the flux and a given source term treatment based on (5.4), approximating this stationary solution. Thus, when numerically treating the source terms, for Property C to be satisfied exactly or approximately (to order $O(\Delta x^2)$) one must ensure that this equilibrium solution would not be perturbed. The source term discretizations presented in the previous sections along with those presented in the appendix for the shallow water system satisfy this property.

In the numerical examples presented next we use again a variable time step as to satisfy the CFL ≤ 1 stability condition:

$$\text{CFL} = \Delta t \max_{i,k} \left\{ \left| \frac{\tilde{a}^{k^+}_{i-\frac{1}{2}}}{\Delta x_i} \right|, \left| \frac{\tilde{a}^{k^-}_{i+\frac{1}{2}}}{\Delta x_i} \right| \right\}, \tag{6.6}$$

where for the shallow water model values $\tilde{a}^{k^+}_{i-\frac{1}{2}}$ and $\tilde{a}^{k^-}_{i+\frac{1}{2}}$ are defined in the appendix.

6.1. Idealized dam-break flow. We first consider a nonstationary case, the dam-break problem in a rectangular channel with flat bottom, $Z = 0$. We computed the solution on a channel of length $L = 2000$m for time $t = 50$s and with initial conditions

$$u(x,0) = 0,$$
$$h(x,0) = \begin{cases} h_1, & x \leq 1000, \\ h_0, & x > 1000, \end{cases}$$

with $h_1 > h_0$. This is the corresponding Riemann problem for the homogeneous problem. The water depth ratio is given by h_0/h_1. The dam collapses at $t = 0$s and the resulting flow consists of a shock wave (bore) traveling downstream and a rarefaction wave (depression wave) traveling upstream. The upstream depth h_1 was set at 10m. When the depth ratio is greater than 0.5, the flow throughout the channel remains subcritical. For depth ratios smaller than 0.5, the flow downstream of the dam position is supercritical while remaining subcritical upstream. For very small values of the ratio h_0/h_1 the flow regime becomes strongly supercritical downstream and the shock wave can be difficult to capture; see [10], for example. Analytical solutions to this problem can be found, for example, in [24].

Comparative results are presented in a 101-point grid and depth ratio 0.005, for all schemes in Figures 16–18, using CFL = 0.9. All the results produced with the adaptive method are significantly better in terms of shock resolution and nonoscillatory behavior in both components of the solution. No entropy problems appeared for all adaptive schemes. The grid clustering and the close following of the downstream shocks can also be seen. It is impressive that the LW and MacCormack schemes are able to produce quality results with no extra dissipative mechanism imposed other than the adaptive one. The performance of the TVD scheme, using the VL limiter, was also improved.

A more quantitative comparison is afforded in Table 1, where we compare the adaptive and nonadaptive Roe and TVD schemes in terms of computational costs, reported as the CPU times, the number of time steps (NT), and the L^1 errors for h and q. It is clear that the adaptive schemes produce accurate results with fewer grid points than the nonadaptive schemes, and even though the total number of time steps is increased, for the adaptive schemes, CPU times are substantially smaller (since in nonadaptive schemes one has to obtain solutions to many more grid points). For example, the adaptive grid calculations with 101 grid points are as accurate as for a fixed grid with 401 points, and require less CPU time. The effect of the parameter D can also be seen in Table 1.

6.2. Flow at rest over topography. We consider system (6.1) with initial conditions

$$u(x,0) = 0 \quad \forall x \in \mathbb{R},$$
$$h(x,0) + Z(x) = H \quad \forall x \in \mathbb{R}.$$

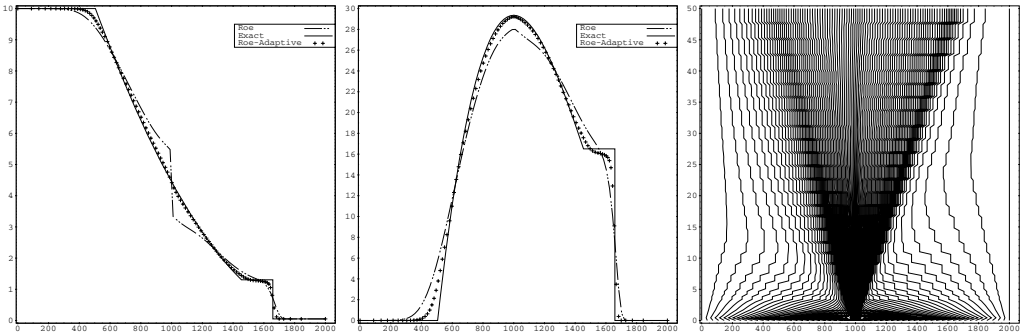


FIG. 16. *Idealized dam-break flow: results with Roe's scheme for h (left), q (middle), and grid point trajectories (right) for the adaptive Roe scheme* ($p = 0.0518$, $D = 0.0065$).

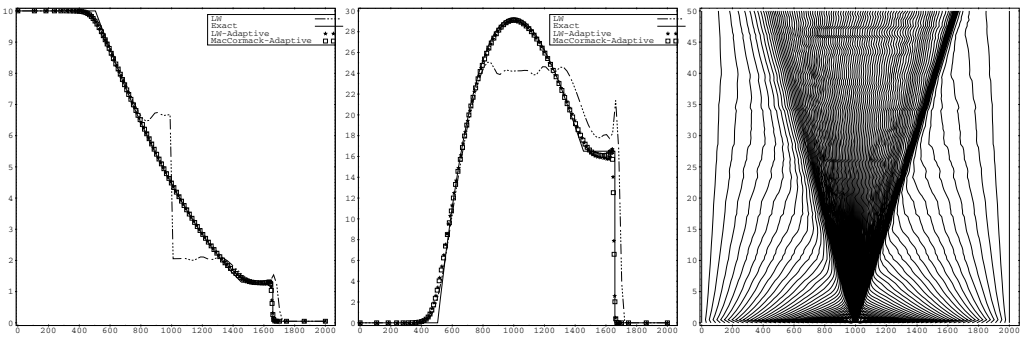


FIG. 17. *Idealized dam-break flow: results with the LW and MacCormack schemes for h (left), q (middle), and grid point trajectories (right) for the adaptive LW scheme* ($p = 0.10$, $D = 0$).

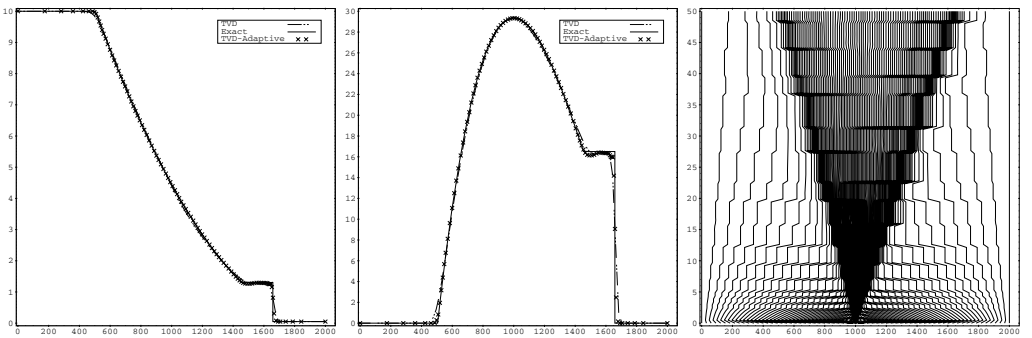


FIG. 18. *Idealized dam-break flow: results with the TVD scheme for h (left), q (middle), and grid point trajectories (right) for the adaptive TVD scheme* ($p = 0.10$, $D = 0.0125$).

Then, clearly,

$$u(x,t) = 0 \quad \forall x \in \mathbb{R}, t \geq 0,$$
$$h(x,t) + Z(x) = H \quad \forall x \in \mathbb{R}, t \geq 0$$

is a solution to (6.1).

We test the schemes with the adaptive mechanism in order to study their behavior

TABLE 1
Idealized dam-break flow: comparative performance of the Roe and TVD schemes.

Description	CPU time (sec)	NT	L^1 error (h)	L^1 error (q)
Roe Fixed grid:				
$N = 51$	0.0051	21	0.0170	0.0393
$N = 101$	0.0206	43	0.0113	0.0240
$N = 201$	0.0828	87	0.0077	0.0169
$N = 401$	0.3401	178	0.0049	0.0102
Roe + AGR:				
$N = 51$, $p = 0.0500$, $D = 0.0065$	0.0234	77	0.0091	0.0238
$N = 101$, $p = 0.0350$, $D = 0$	0.0583	97	0.0080	0.0220
$N = 101$, $p = 0.0500$, $D = 0$	0.1072	178	0.0061	0.0176
$N = 101$, $p = 0.0500$, $D = 0.0065$	0.0931	158	0.0049	0.0137
TVD Fixed grid:				
$N = 51$	0.0112	22	0.0050	0.0156
$N = 101$	0.0450	44	0.0028	0.0078
$N = 201$	0.1837	90	0.0009	0.0025
$N = 401$	0.7432	180	0.0007	0.0020
TVD + AGR:				
$N = 51$, $p = 0.1$, $D = 0.0125$	0.0788	136	0.0029	0.0076
$N = 101$, $p = 0.1$, $D = 0$	0.4292	369	0.0018	0.0055
$N = 101$, $p = 0.1$, $D = 0.0125$	0.3927	342	0.0007	0.0023

to this benchmark problem [13, 9, 11] with $Z(x)$ given by

$$Z(x) = \begin{cases} 0.2 - 0.05(x-10)^2, & 8 \le x \le 12, \\ 0 \quad \text{otherwise}, \end{cases} \tag{6.7}$$

in a channel of length $L = 25$m and $H = 2$m. Figures 19 and 20 display the final water level and the final unit discharge values, respectively, for the Roe and MacCormack schemes at time $t = 200$s and in a grid of 101 points. Both schemes with the adaptation method perfectly preserve this steady state, up to machine accuracy. Grid points are concentrated where most needed, in the topography area. The movement of the grid points is stopping much faster for the MacCormack scheme being a second order method in time. Similar results and observations were made for the TVD scheme that are not presented here for brevity.

6.3. Steady transcritical flow. In this case, and for the same initial conditions as in section 6.2, we impose an upstream boundary condition for the discharge $q = 1.53\text{m}^2/\text{s}$ and a downstream boundary condition for the water level $H_0 = 0.66$m only in the case where the flow is subcritical. If the flow becomes supercritical downstream, no condition for the water level is imposed. Results at $t = 250$s in a 51-point grid are presented in Figures 21 and 22 for the Roe and TVD schemes (using the MM limiter) and CFL = 0.9.

In Table 2 we quantify the errors and performance of the schemes for this test problem. Similar observations with those in section 6.1 can be deduced. The importance of using the parameter D for steady state problems can be clearly seen.

6.4. Steady transcritical flow with shock. The initial conditions in this case was taken to be as in the previous section, with H being the constant water level downstream provided by the boundary condition. In this test case the upstream boundary condition for the discharge was $q = 0.18\text{m}^2/\text{s}$ and the downstream boundary condition for the water level was $H = 0.33$m.

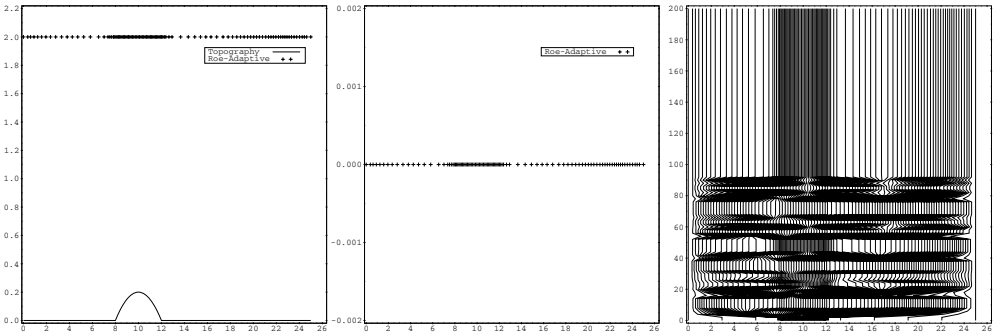


FIG. 19. *Flow at rest over topography: results with the adaptive Roe scheme for $h + Z$ (left), q (middle), and grid point trajectories (right)* ($p = 0.06$, $D = 0.0525$).

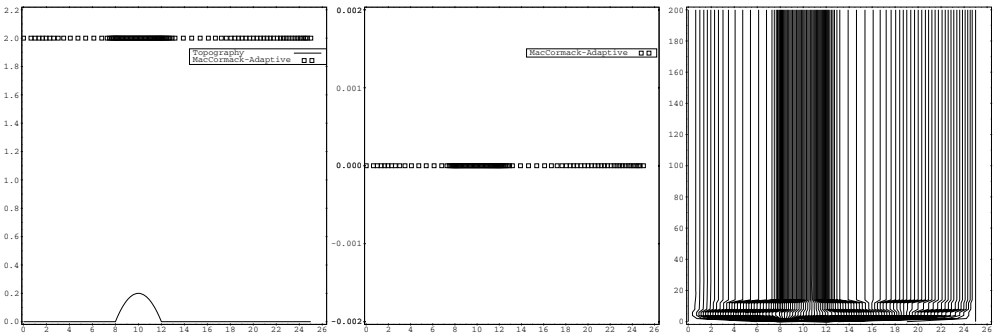


FIG. 20. *Flow at rest over topography: results with the adaptive MacCormack scheme for $h + Z$ (left), q (middle), and grid point trajectories (right)* ($p = 0.06$, $D = 0.0525$).

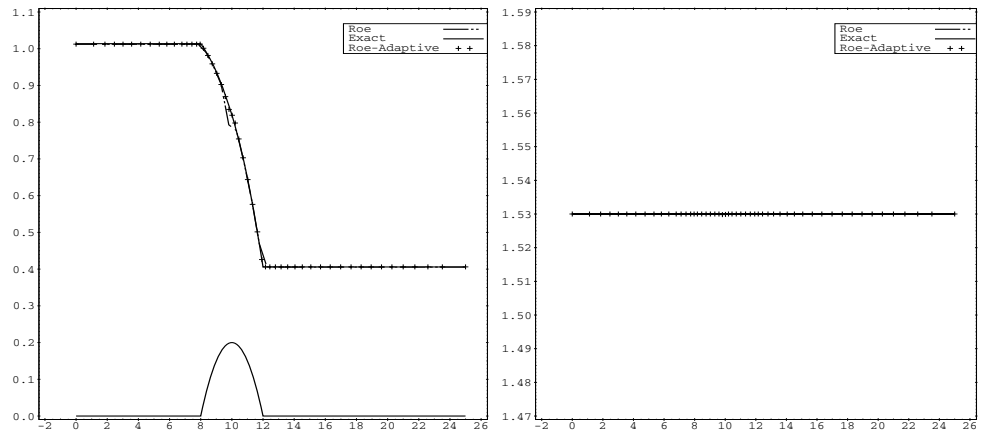


FIG. 21. *Steady transcritical flow: results with Roe's scheme for $h + Z$ (left), discharge (q) (right)* ($p = 0.075$, $D = 0.06$).

Results for a coarse 51-point grid for the Roe and TVD schemes and also on a 101-point grid for the TVD scheme with the MM limiter used are shown in Figures 23–25. The improvement in the results compared to the uniform grid results is clear. The shock resolution is very good and the steady state is perfectly preserved for a

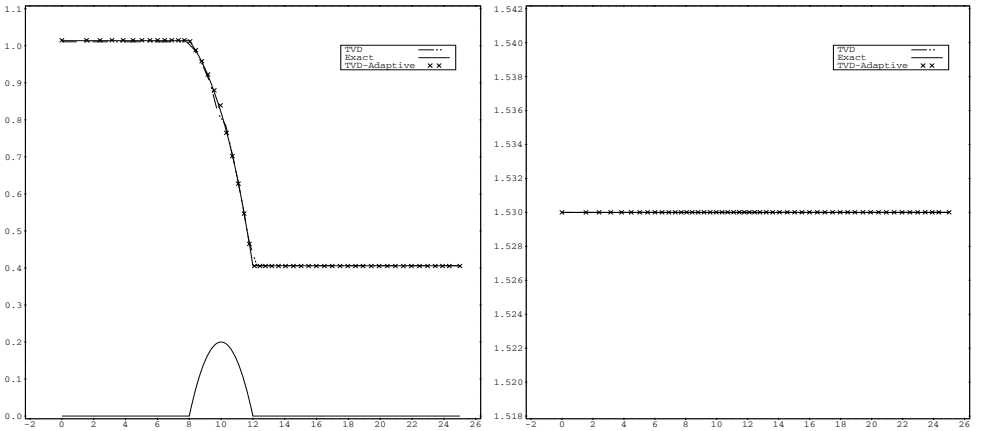


FIG. 22. *Steady transcritical flow: results with the TVD scheme for depth $h+Z$ (left), discharge (q) (right)* ($p = 0.1$, $D = 0.05$).

TABLE 2
Steady transcritical flow: performance of the Roe and TVD schemes.

Description	CPU time (sec)	NT	L^1 error (h)	L^1 error (q)
Roe Fixed grid:				
$N = 101$	3.352	6115	$1.85\ 10^{-3}$	$1.54\ 10^{-5}$
$N = 201$	13.25	12177	$8.81\ 10^{-4}$	$9.63\ 10^{-6}$
$N = 401$	53.51	24304	$2.49\ 10^{-4}$	$5.54\ 10^{-6}$
Roe + AGR:				
$N = 51$, $p = 0.0750$, $D = 0.06$	2.150	6504	$1.18\ 10^{-3}$	$9.64\ 10^{-6}$
$N = 101$, $p = 0.0750$, $D = 0.06$	9.563	14824	$4.94\ 10^{-4}$	$7.78\ 10^{-6}$
$N = 101$, $p = 0.0750$, $D = 0$	9.529	14364	$1.04\ 10^{-2}$	$1.63\ 10^{-3}$
TVD Fixed grid:				
$N = 101$	6.292	6166	$1.27\ 10^{-3}$	$4.83\ 10^{-7}$
$N = 201$	24.84	12273	$3.71\ 10^{-4}$	$3.24\ 10^{-7}$
TVD + AGR:				
$N = 51$, $p = 0.1$, $D = 0.05$	3.350	5975	$6.04\ 10^{-4}$	$2.60\ 10^{-7}$
$N = 101$, $p = 0.1$, $D = 0.05$	23.17	20747	$4.56\ 10^{-4}$	$2.53\ 10^{-7}$
$N = 101$, $p = 0.1$, $D = 0$	21.31	18693	$9.16\ 10^{-3}$	$1.54\ 10^{-3}$

long time calculation of $t = 800$s. The CFL value used in all tests was 0.8.

Finally in Figure 26 the numerical results for the depth are shown for the adaptive MacCormack scheme, where in spite of the formation of a strong shock the scheme can produce a nonoscillatory steady state solution, especially when applying the parameter D.

6.5. Dam-break flow over topography. In this example we solve the shallow water equations with a wavy bottom $Z(x)$,

$$Z(x) = \begin{cases} 0.3(\cos(\pi(x-1)/2))^{30}, & |x-1| \le 1, \\ 0 & \text{otherwise}, \end{cases} \tag{6.8}$$

and initial conditions

$$h(x,0) = \begin{cases} 2.0 - Z(x), & -10 \le x < 1, \\ .35 - Z(x), & 1 \le x < 10, \end{cases} \qquad u(x,0) = \begin{cases} 1, & -10 \le x < 1, \\ 0, & 1 \le x < 10, \end{cases} \tag{6.9}$$

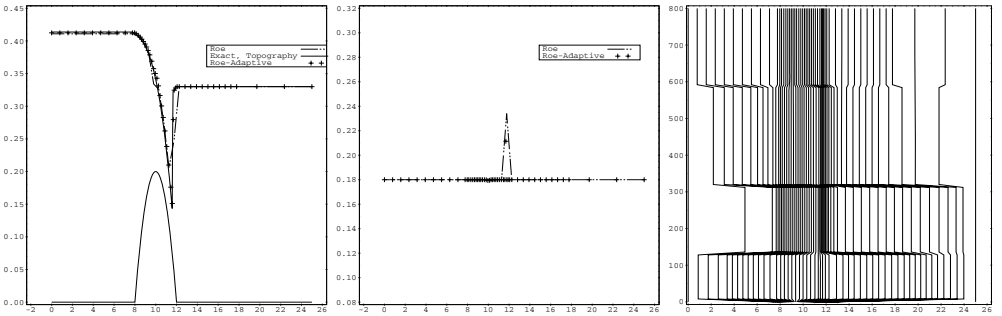


FIG. 23. *Steady transcritical flow with shock: results with Roe's scheme for $h + Z$ (left), q (middle), and grid point trajectories (right) for the adaptive Roe scheme* ($p = 0.08$, $D = 0.07$).

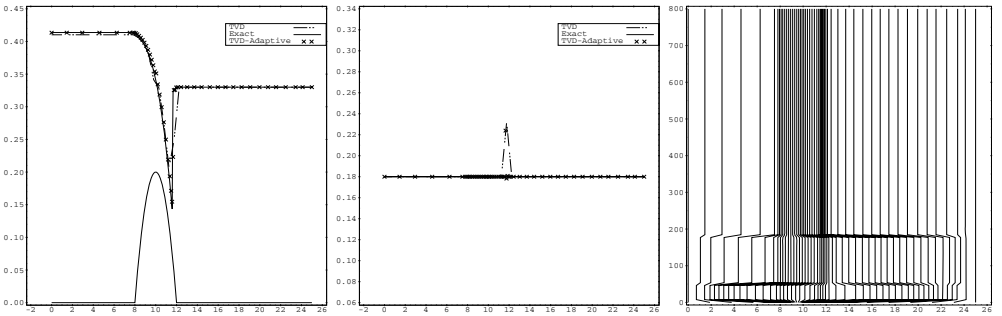


FIG. 24. *Steady transcritical flow with shock: results with the TVD scheme for $h + Z$ (left), q (middle), and grid point trajectories (right) for the adaptive TVD scheme on a* 51*-point grid* ($p = 0.08$, $D = 0.075$).

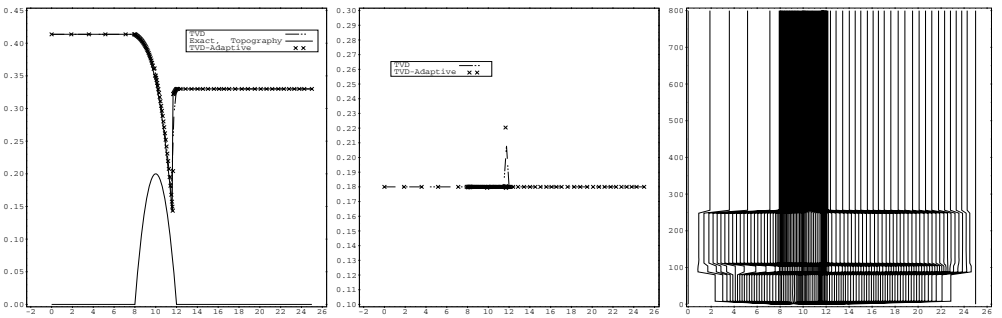


FIG. 25. *Steady transcritical flow with shock: results with the TVD scheme for $h + Z$ (left), q (middle), and grid point trajectories (right) for the adaptive TVD scheme on a* 101*-point grid* ($p = 0.08$, $D = 0.075$).

in order to test the adaptive behavior of the schemes in an unsteady flow over topography; this test was also presented in [28]. The solution profiles for the water depth and velocity profile at time $t = 1$s are shown in Figures 27–29. A grid of 101 points was used in all calculations. The exact solution for this problem is a reference solution calculated in a uniform grid with 4001 grid points. The improvements in the numerical solution can be observed for all the schemes; even the MacCormack scheme can produce a stable solution when compared with the one calculated in a uniform grid.

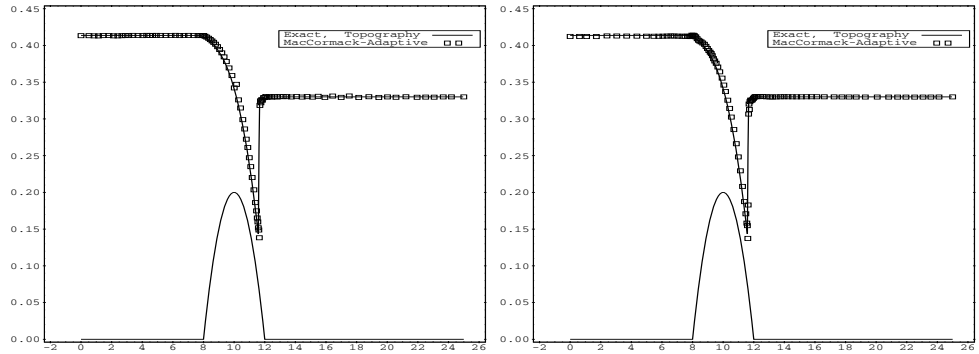


FIG. 26. *Steady transcritical flow with shock: results for $h + Z$ with the adaptive MacCormack scheme with $p = 0.12$, $D = 0$ (left) and with $p = 0.12$, $D = 0.003$ (right).*

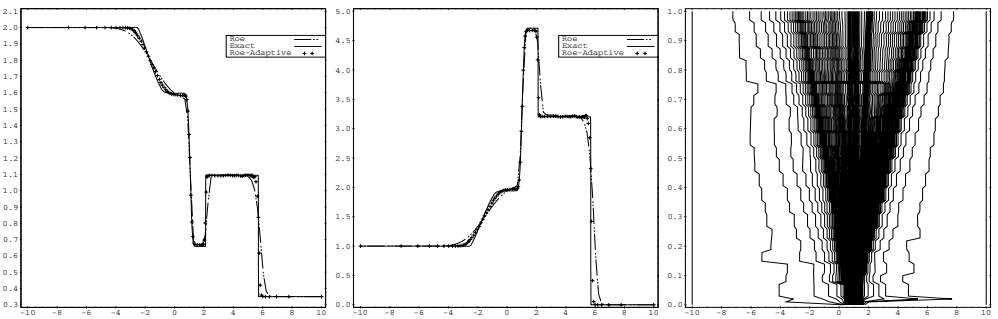


FIG. 27. *Dam-break flow over topography: results with Roe's scheme for h (left), u (middle), and grid point trajectories (right) for the adaptive Roe scheme ($p = 0.109$, $D = 0.005$).*

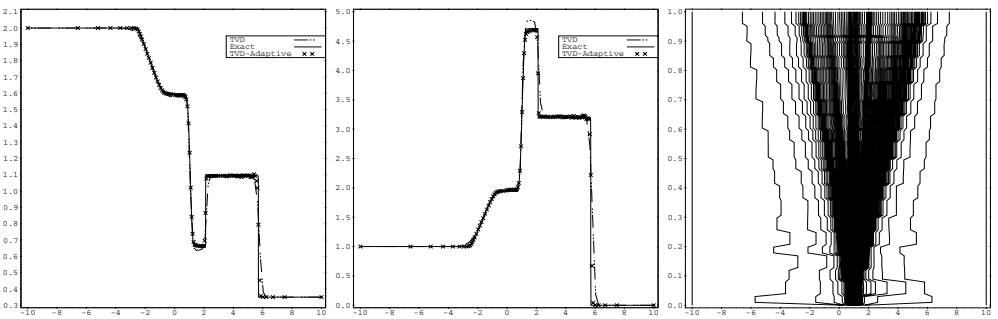


FIG. 28. *Dam-break flow over topography: results with the TVD scheme for h (left), u (middle), and grid point trajectories (right) for the adaptive TVD scheme ($p = 0.11$, $D = 0.006$).*

7. Conclusions. In this work we have rigorously investigated, in a general framework, the numerical behavior of an adaptive grid redistribution (AGR) mechanism, when combined with classical finite volume numerical methods with well-known characteristics, for solving one-dimensional hyperbolic conservation laws. The evolving mesh is constructed such that its spatial resolution is controlled via selective geometrical characteristics of the numerical solution, by choosing some power of the curvature of the solution as the estimator function. A conservative reconstruction procedure for the numerical solution is also applied at each evolution step.

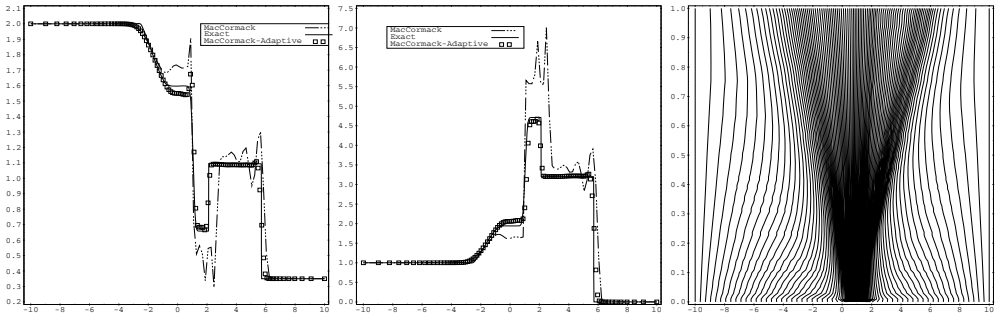


FIG. 29. *Dam-break flow over topography: results with the MacCormack scheme for h (left), u (middle), and grid point trajectories (right) for the adaptive MacCormack scheme* ($p = 0.0345$, $D = 0.001$).

From the numerical experiments produced in the course of this work we concluded the following advantages of the use of this particular AGR method. The method, when applied to classical second order schemes that compute oscillatory solutions, suppresses the oscillations produced, and as such, classical schemes like the Lax–Wendroff or the MacCormack scheme become stable and can produce reliable solutions. When applied to numerical schemes that do not satisfy entropy conditions, the mechanism can converge to the entropy satisfying solution. The method performs well with hyperbolic problems with source terms by producing stable solutions for first and second order balanced schemes that converge to the correct steady states. The method can automatically detect, resolve, and track steep wave fronts and discontinuities, without having to resort to finer grids. It can be combined with a high resolution flux or slope-limiting TVD scheme and improve the numerical solutions obtained. Only two computational parameters have to be tuned for each problem: the power of the estimator function and the tolerance that measures the relative mean displacement between two consecutive grid distributions, under which the reconstruction step is avoided. The mechanism has been proved robust for all applications presented and schemes used.

There is a need for further theoretical work and development of the method, as well as extension of the ideas presented in this work to higher dimensions. These will provide a more in-depth understanding of the method's behavior, the advantages gained when the adaptive mechanism is combined with existing numerical schemes, and, more generally, the influence of evolving adaptive grids to the numerical solution of evolution PDEs.

Appendix. Following [7] and [9] the average velocity and celerity, $\tilde{u}_{i+\frac{1}{2}}$ and $\tilde{c}_{i+\frac{1}{2}}$, for the eigenvalues $\tilde{a}^{1,2}_{i+\frac{1}{2}} = \tilde{u}_{i+\frac{1}{2}} \pm \tilde{c}_{i+\frac{1}{2}}$ and eigenvectors, and $\tilde{\mathbf{e}}^{1,2}_{i+\frac{1}{2}} = \left(1, \tilde{a}^{1,2}_{i+\frac{1}{2}}\right)^{\mathrm{T}}$ of Roe's linearized Jacobian $\mathbf{J}_{i+\frac{1}{2}}$ for the shallow water system, are calculated as

$$\tilde{u}_{i+\frac{1}{2}} = \frac{u_{i+1}\sqrt{h_{i+1}} + u_i\sqrt{h_i}}{\sqrt{h_{i+1}} + \sqrt{h_i}}, \qquad \tilde{c}_{i+\frac{1}{2}} = \sqrt{g\tilde{h}}, \tag{A.1}$$

with $\tilde{h} = (h_{i+1} + h_i)/2$.

The wave strengths $\alpha^{1,2}_{i+\frac{1}{2}}$ follow solving (5.7), and they are

$$\alpha^{1,2}_{i+\frac{1}{2}} = \frac{1}{2}\Delta_{i+\frac{1}{2}}h \pm \frac{1}{2\tilde{c}_{i+\frac{1}{2}}}(\Delta_{i+\frac{1}{2}}(uh) - \tilde{u}_{i+\frac{1}{2}}\Delta_{i+\frac{1}{2}}h), \tag{A.2}$$

where the operator $\Delta_{i+\frac{1}{2}}(\cdot) = (\cdot)_{i+1} - (\cdot)_i$.

For a general system of equations and temporarily dropping the subscript indices we define $\mathbf{\Lambda}$ to be the diagonal form of $\mathbf{J}$ given by its eigenvalues, with

$$\mathbf{J} = \mathbf{R}\mathbf{\Lambda}\mathbf{R}^{-1}, \tag{A.3}$$

where $\mathbf{R}$ is the matrix of the eigenvectors of $\mathbf{J}$. We can now define the $|\mathbf{\Lambda}|$, $\mathbf{\Lambda}^+$, and $\mathbf{\Lambda}^-$ matrices as

$$|\mathbf{\Lambda}| = \operatorname{diag}(|a^1|, \ldots, |a^M|), \quad \mathbf{\Lambda}^{\pm} = \operatorname{diag}(a^{1^{\pm}}, \ldots, a^{M^{\pm}}), \tag{A.4}$$

where $a^{k^+} = \max(0, a^k)$ and $a^{k^-} = \min(0, a^k)$. We can now define

$$|\mathbf{J}| = \mathbf{R}|\mathbf{\Lambda}|\mathbf{R}^{-1}, \tag{A.5}$$

$$\mathbf{J}^{\pm} = \mathbf{R}\mathbf{\Lambda}^{\pm}\mathbf{R}^{-1}. \tag{A.6}$$

Therefore,

$$|\mathbf{J}| = \mathbf{J}^+ - \mathbf{J}^-, \quad \mathbf{J}^{\pm} = \frac{1}{2}(\mathbf{J} \pm |\mathbf{J}|). \tag{A.7}$$

Now at the discrete level the $\mathbf{J}_{i+\frac{1}{2}}$ and $|\mathbf{J}_{i+\frac{1}{2}}|$ are defined as

$$\mathbf{J}_{i+\frac{1}{2}} = \mathbf{R}_{i+\frac{1}{2}}\mathbf{\Lambda}_{i+\frac{1}{2}}\mathbf{R}^{-1}_{i+\frac{1}{2}}, \tag{A.8}$$

$$|\mathbf{J}_{i+\frac{1}{2}}| = \mathbf{R}_{i+\frac{1}{2}}|\mathbf{\Lambda}_{i+\frac{1}{2}}|\mathbf{R}^{-1}_{i+\frac{1}{2}}. \tag{A.9}$$

Using the above formulation to calculate the upwind approximation of the source term $\mathbf{G}$, we need to evaluate $\widetilde{\mathbf{G}}_{i\pm\frac{1}{2}}$ approximations, that is, some average of the source term on the left and right of the cell interfaces. Then a well-balanced approximation of the source term, at each computational cell, is

$$\widetilde{\mathbf{G}}_{i\pm\frac{1}{2}} = \left[0, -\frac{g}{2}(h_{i+1} + h_i)(Z_{i+1} - Z_i)\right]^{\mathrm{T}}. \tag{A.10}$$

The values of $\beta^{1,2}_{i\pm\frac{1}{2}}$ in (5.14) are the components of $\mathbf{R}^{-1}_{i+\frac{1}{2}}\widetilde{\mathbf{G}}_{i\pm\frac{1}{2}}$, and for the shallow water system they are

$$\beta^{1,2}_{i\pm\frac{1}{2}} = \mp\frac{\tilde{c}_{i+\frac{1}{2}}(Z_{i+1} - Z_i)}{2}.$$

Acknowledgment. The authors wish to thank Prof. Ch. Makridakis for his helpful discussions during the course of this work.

REFERENCES

[1] Ch. Arvanitis, Th. Katsaounis, and Ch. Makridakis, *Adaptive finite element relaxation schemes for hyperbolic conservations laws*, M2AN Math. Model. Numer. Anal., 35 (2001), pp. 17–33.

[2] Ch. Arvanitis, Ch. Makridakis, and A. E. Tzavaras, *Stability and convergence of a class of finite element schemes for hyperbolic systems of conservation laws*, SIAM J. Numer. Anal., 42 (2004), pp. 1357–1393.

[3] Ch. Arvanitis, *Mesh redistribution strategies and finite element schemes for hyperbolic conservation laws*, submitted.
[4] B. N. Azarenok and T. Tang, *Second-order Godunov-type scheme for reactive flow calculations on moving meshes*, J. Comput. Phys., 206 (2005), pp. 48–80.
[5] B. N. Azarenok, S. A. Ivanenko, and T. Tang, *Adaptive mesh redistribution method based on Godunov's scheme*, Commun. Math. Sci., 1 (2003), pp. 152–179.
[6] G. Beckett and J. A. Mackenzie, *Convergence analysis of finite difference approximations on equidistributed grids to a singularly perturbed boundary value problem*, Appl. Numer. Math., 35 (2000), pp. 87–109.
[7] A. Bermudez and M. E. Vázquez, *Upwind methods for hyperbolic conservation laws with source terms*, Comput. & Fluids, 23 (1994), pp. 1049–1071.
[8] L. Chen, P. Sun, and J. Xu, *Optimal anisotropic meshes for minimizing interpolation errors in L^p-norm*, Math. Comp., to appear.
[9] A. I. Delis, *Improved application of the HLLE Riemann solver for the shallow water equations with source terms*, Comm. Numer. Methods Engrg., 19 (2003), pp. 59–83.
[10] A. I. Delis and C. P. Skeels, *TVD schemes for open channel flow*, Internat. J. Numer. Methods Fluids, 26 (1998), pp. 791–809.
[11] A. I. Delis and Th. Katsaounis, *Relaxation schemes for the shallow water equations*, Internat. J. Numer. Methods Fluids, 41 (2003), pp. 695–719.
[12] R. Fazio and R. J. LeVeque, *Moving-mesh methods for one-dimensional hyperbolic problems using CLAWPACK*, Comput. Math. Appl., 45 (2003), pp. 273–298.
[13] N. Goutal and F. Maurel, in Proceedings of the 2nd Workshop on Dam-Break Wave Simulation, HE-43/97/016/B, Département Laboratoire National d'Hydraulic, Groupe Hydraulic Fluviale Electricité de France, France, 1997.
[14] A. Harten and J. M. Hyman, *Self adjusting grid methods for one-dimensional hyperbolic conservation laws*, J. Comput. Phys., 50 (1983), pp. 17–33.
[15] C. Hirch, *Numerical Computation of Internal and External Flows*, Wiley, New York, 1988.
[16] M. E. Hubbard and P. Garcia-Navarro, *Flux difference splitting and the balancing of source terms and flux gradients*, J. Comput. Phys., 165 (2000), pp. 89–125.
[17] J. M. Hyman, S. Li, and L. Petzold, *An adaptive moving mesh method with static rezoning for partial differential equations*, Comput. Math. Appl., 10–11 (2003), pp. 1511–1524.
[18] R. J. LeVeque, *Finite Volume Methods for Hyperbolic Problems*, Cambridge University Press, Cambridge, UK, 2002.
[19] R. Li, T. Tang, and P. Zhang, *Moving mesh methods in multiple dimensions based on harmonic maps*, J. Comput. Phys., 170 (2001), pp. 562–588.
[20] S. Li and L. Petzold, *Moving mesh methods with upwind schemes for time dependent PDEs*, J. Comput. Phys., 131 (1997), pp. 368–377.
[21] S. Li, L. Petzold, and Y. Ren, *Stability of moving mesh systems of partial differential equations*, SIAM J. Sci. Comput., 20 (1998), pp. 719–738.
[22] S. V. Pennington and M. Berzins, *New NAG software for first order partial differential equations*, ACM Trans. Math. Software, 20 (1994), pp. 63–99.
[23] J. M. Stockie, J. A. Mackenzie, and R. D. Russel, *A moving mesh method for one-dimensional hyperbolic conservation laws*, SIAM J. Sci. Comput., 22 (2001), pp. 1791–1813.
[24] J. J. Stoker, *Water Waves*, Interscience, New York, 1957.
[25] P. K. Sweby, *High resolution schemes using flux limiters for hyperbolic conservation laws*, SIAM J. Numer. Anal., 21 (1984), pp. 995–1011.
[26] E. Tadmor and J. Tanner, *An adaptive order Godunov type central scheme*, in Hyperbolic Problems: Theory, Numerics, Applications, Proceedings of the 9th International Conference in Pasadena, 2002, T. Hou and E. Tadmor, eds., Springer-Verlag, Berlin, 2003, pp. 871–880.
[27] Z. Tan, Z. Zhang, Y. Huang, and T. Tang, *Moving mesh methods with locally varying time steps*, J. Comput. Phys., 35 (2004), pp. 17–33.
[28] H. Tang, *Solution of the shallow-water equations using an adaptive moving mesh method*, Internat. J. Numer. Methods Fluids, 44 (2004), pp. 789–810.
[29] H. Tang and T. Tang, *Adaptive mesh methods for one- and two-dimensional hyperbolic conservation laws*, SIAM J. Numer. Anal., 41 (2003), pp. 487–515.
[30] H. Z. Tang, T. Tang, and P. W. Zhang, *An adaptive mesh redistribution method for nonlinear Hamilton-Jacobi equations in two and three dimensions*, J. Comput. Phys., 188 (2003), pp. 543–572.

SIAM J. SCI. COMPUT.
Vol. 28, No. 5, pp. 1957–1973
© 2006 Society for Industrial and Applied Mathematics

SEGMENTATION WITH DEPTH: A LEVEL SET APPROACH*

WEI ZHU†, TONY CHAN‡, AND SELIM ESEDOĞLU‡

Abstract. Segmentation with depth is the challenging problem of obtaining three dimensional information from a single two dimensional image. Unlike the standard segmentation problem, the goal of segmentation with depth is to determine not only the boundaries of objects that appear in the image, but also their relative distances to the observer by making use of occlusion relations. Nitzberg, Mumford, and Shiota [*The* 2.1*D sketch*, in Proceedings of the Third International Conference on Computer Vision, Osaka, Japan, 1990; *Filtering, Segmentation, and Depth*, Lecture Notes in Comput. Sci. 662, Springer-Verlag, Berlin, 1993] proposed a variational formulation of this problem; according to their model, called the 2.1D sketch model, the regions that the objects occupy and their relative depth are to be extracted from the two dimensional image by minimizing a curvature based functional. Numerically, this is a highly nontrivial problem as the functional involves curvatures of the unknown contours. In this paper, we develop a level set based procedure for minimizing the Nitzberg–Mumford–Shiota energy. Unlike the minimization technique of Nitzberg–Mumford–Shiota [*Filtering, Segmentation, and Depth*, Lecture Notes in Comput. Sci. 662, Springer-Verlag, Berlin, 1993], our technique represents contours implicitly and thus allows for connections between T-junctions to take place automatically.

Key words. illusory contours, level set methods, Euler's elastica

AMS subject classifications. 65K10, 68U10, 94A08

DOI. 10.1137/050622213

1. Introduction.

We develop level set methods [27, 29] to numerically minimize the Nitzberg–Mumford–Shiota (NMS) functional that was proposed for the segmentation with depth problem [26, 25]. Segmentation with depth fundamentally differs from standard segmentation [16, 23, 24, 21, 6, 7, 9, 10] (see Figure 1). Indeed, the typical goal of a standard segmentation procedure is to determine what parts of the image domain belong to distinct objects in the depicted scene. Segmentation with depth, on the other hand, refers to the difficult task of extracting three dimensional information from a single two dimensional image that depicts a scene with several objects in it. Unlike the standard segmentation problem, segmentation with depth attempts to reconstruct both the regions occupied by different objects in the given two dimensional image and their relative distances (depth) to the observer. Recovering depth information about objects contained in a scene is of course an important step toward understanding the three dimensional layout of the scene. In turn, information about the three dimensional layout of a scene is invaluable in many applications that require automatic interpretation of the environment, such as navigation. To accomplish this, the model of NMS tries to decompose the given image into regions that are allowed to overlap, and includes terms that depend on the curvature of these regions' boundaries. Because of its nonlinearity and the difficulty to express it effectively, the

*Received by the editors January 6, 2005; accepted for publication (in revised form) May 22, 2006; published electronically November 24, 2006. This work has been supported by ONR contract N00014-03-1-0888, NSF contract DMS-9973341, NIH contract P20 MH65166, and NSF contract DMS-0410085.

http://www.siam.org/journals/sisc/28-5/62221.html

†Current address: Courant Institute of Mathematical Sciences, 251 Mercer Street, New York, NY 10012 (wzhu@ cims.nyu.edu). Address at the time of this work: Department of Mathematics, University of California, Los Angeles, 405 Hilgard Avenue, Los Angeles, CA 90095.

‡Department of Mathematics, University of California, Los Angeles, 405 Hilgard Avenue, Los Angeles, CA 90095 (TonyC@college.ucla.edu, esedoglu@math.ucla.edu).

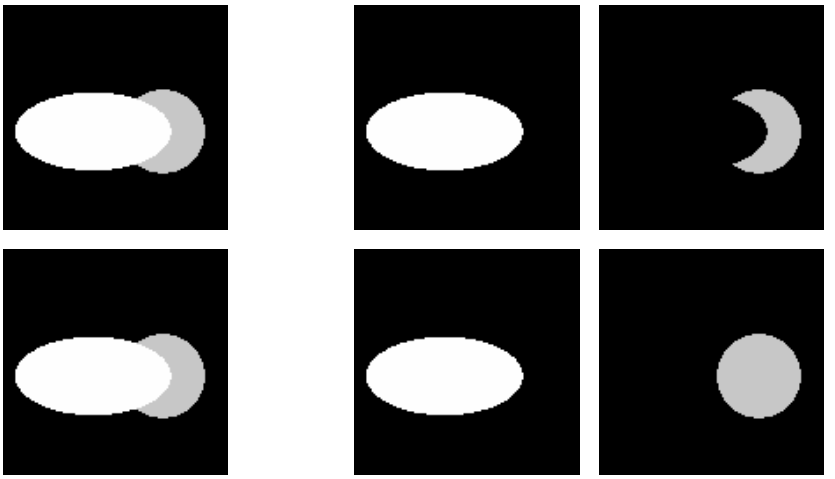


FIG. 1. *The first row contains the original image (left) with two regions (middle and right) given by any multiphase gray-scale intensity based segmentation method, and the second row illustrates the results from a segmentation with depth model. Besides the occlusion relation between the two objects ("Front" and "Behind" in this example), the missing boundary of the Behind object is also restored.*

curvature dependence of the functional involved is one of the most difficult aspects of the model. Specifically, the energy has terms of the form

$$\int_{\partial R} \phi(\kappa) ds, \tag{1}$$

where κ denotes the curvature of the boundary of the region R and ϕ is a particular function.

In [26], the authors avoid minimizing their functional directly. Instead, they propose a combinatorial algorithm that first detects edges and T-junctions in the given two dimensional image and then considers all possible ways of connecting the T-junctions by new edges that are consistent with a given ordering hypothesis. The ordering hypothesis with respect to which the detected T-junctions can be connected using the smallest amount of energy is found to be the correct depth relation between the (possibly overlapping) regions. This is thus a combinatorial technique with very high computational complexity. Moreover, it requires explicitly detecting T-junctions in the image as a preprocessing step, which can be a significant source of errors (in the form of false positives and misses). On the other hand, the model itself makes no explicit reference to T-junctions; therefore it should be possible to minimize the energy without detecting such features first.

In [11], Esedoḡlu and March propose a new, direct way of minimizing the NMS energy that is based on curve evolution and gradient descent. It works in much the same way that the Chan–Vese algorithm [9] minimizes the Mumford–Shah segmentation functional [23, 24]. Their technique makes essential use of Γ-convergence ideas [18] and a conjecture of De Giorgi [14] to approximate the NMS functional by elliptic ones which are numerically much more convenient; see also [1, 19, 20] for other related examples of such approximations in image processing. This approach avoids the need for detecting and connecting T-junctions explicitly; these happen automatically

as a consequence of the curve evolution (and involve changes in the topology of the curves). However, the approximation technique used by the authors forces them to consider only quadratic dependence on curvature (i.e., $\phi(x) = x^2$), which is an approximation to but is different from the correct choice prescribed by NMS in their original formulation. The difference is significant as explained by the authors in [11].

In this paper, we give a level set based formulation [27, 29, 28] to minimize the NMS functional that is also based on curve evolution and gradient descent. However, our techniques allow us to use the more general curvature dependence required by the original model of NMS. This way we get an implementation that remains faithful to the model and avoids the issues caused by using quadratic dependence as in [11]. However, we should point out that it is also possible to approximate via Gamma convergence the correct curvature dependence in the NMS model; see, for example, papers [2, 3, 4, 5] in this regard, which propose alternative methods to that of this paper.

The level set method of Osher and Sethian [27] is particularly well suited to handling curvature dependent functionals, since the curvature of a contour has a very simple expression in terms of the level set function that represents it. For example, let ψ be a level set function; then the curvature κ along any level curve $\psi = c$ (c is any real value) can be represented as

$$\kappa = \nabla \cdot \left(\frac{\nabla \psi}{|\nabla \psi|} \right). \tag{2}$$

With this representation, it is also straightforward to handle the absolute value of curvature that appears in the above functional. By using level set methods, the problem of minimizing the NMS functional turns out to be solving a system of fourth order partial differential equations. These kinds of equations were also solved by the first two authors in [32] for capturing illusory contours. In this paper, we also employ a semi-implicit discretization scheme due to Smereka [30] to solve the resulting fourth order equations. This method can speed up computations significantly, since explicit schemes for these equations are notoriously slow to converge to a stationary state.

The rest of this paper is organized as follows. In the next section, we introduce the background of segmentation with depth as well as the NMS functional. In section 3, we rewrite the functional in a level set formulation and derive the associated gradient descent flows. We then discuss numerical solution techniques including Smereka's semi-implicit scheme in the subsequent section 4. In section 5, we present some numerical experiments on synthetic images that appear in [11]. Section 6 is devoted to our conclusions.

2. NMS functional. In this section, we sketch the problem of segmentation with depth and present the NMS functional. Further details can be found in [26, 11].

The most important unknowns of the NMS model are the shapes of the objects that make up the scene and their relative distances to the observer (i.e., their depth). The relative distances of the objects can be described by an ordering between them. The shapes of the objects are described by subsets of the image domain. Since the model allows for occlusions, these regions need not be disjoint: Parts of regions in the back (further from the observer) might be blocked from view by parts of regions in front (closer to the observer). The model thus inevitably involves reconstructing invisible parts of the regions' boundaries. This calls for the notion of the "most likely completion" of a boundary contour. In the NMS model, the most likely completion of a curve is expressed as the minimizer of a curvature dependent functional.

The 2.1D model of NMS makes the following simplifying assumptions about the three dimensional scene:

- Objects in the scene are not entangled.
- There are no self-occlusions.
- Gray-scale intensities of different objects are approximately uniform and different from each other.

The first two assumptions can be summarized as follows: the 2.1D sketch model of NMS assumes that the objects making up the three dimensional scene live in distinct planes which are parallel to each other and which are perpendicular to the line of sight of the observer. The unknowns of the problem are

- the ordering relation between the regions (which is in front of which),
- the (unoccluded) shapes of the regions, and
- the approximate gray-scale intensity of the corresponding objects.

Let $f : \Omega \to R^1$ be a two dimensional image defined on $\Omega \subset R^2$, and $R_1, \ldots, R_n$ be the regions occupied by objects inside the image. Denote the visible parts of the above objects by $R_1^{'}, \ldots, R_n^{'}$, respectively.

Suppose the regions $R_1, \ldots, R_n$ are listed in order of increasing distance to the observer (i.e., depth), so that R_1, R_n are the nearest and farthest regions, respectively. Then the visible part $R_i^{'}$ of R_i is given by

$$R_1^{'} = R_1, \qquad R_i^{'} = R_i - \bigcup_{j<i} R_j \quad \text{for } i = 2, \ldots, n.$$

For simplicity, we denote $R_{n+1}^{'} = \Omega - \bigcup_{j<n+1} R_j$ as the background. Then the NMS functional reads

$$E = \sum_{i=1}^{n} \int_{\partial R_i \bigcap \Omega} [\alpha + \beta\phi(\kappa)]ds + \sum_{i=1}^{n+1} \int_{R_i^{'}} (f(x) - c_i)^2 dx, \tag{3}$$

where α, β are two nonnegative parameters, and the unknowns $c_i \in R^1$ denote the approximate gray-scale intensities (color) of the corresponding objects. An important point is that the contour integrals in energy (3) are taken along the entire boundaries of the regions inside the image domain Ω, not just the visible parts of these boundaries; i.e., instead of on the visible boundaries $\partial R_i^{'} \bigcap \Omega$, the contour integrals are taken on $\partial R_i \bigcap \Omega$. These terms therefore dictate how the occluded parts of boundary curves are best reconstructed. The area integrals in the energy (3) make up the "fidelity" terms in the model. They ensure that the reconstructed configuration of regions and their ordering in three dimensional space yield a two dimensional image that agrees with the observed one, namely, $f(x)$. The function ϕ determines how the curvature information will be incorporated in the functional. The authors in [25, 26] choose ϕ as follows:

$$\phi(x) = \begin{cases} x^2, & |x| \le 1, \\ |x|, & |x| > 1. \end{cases} \tag{4}$$

Corners are allowed when (4) is used because of the linear growth of ϕ at infinity. In [11], the authors select the quadratic dependence on curvature, i.e., $\phi(x) = x^2$, which inevitably smears corners. Furthermore, quadratic growth of ϕ at infinity leads to a stability issue as explained in that paper, where small disks are encouraged by the energy to grow. In [11], quadratic dependence on curvature is chosen because the approximation procedure used there requires it. In contrast, the techniques we use

in this paper allow for a more general dependence on curvature, including the choice (4), and hence allow us to avoid the stability and smearing drawbacks.

3. Level set method for minimizing the NMS functional. As noted in the introduction, the level set method of Osher and Sethian [27] is very well suited for the representation of functionals that involve geometric quantities such as curvature. In this section, we first rewrite the NMS functional in terms of level set functions that represent the unknown regions R_i. Subsequently, we derive the associated gradient descent flows in terms of the level set functions to minimize the NMS functional.

3.1. Level set formulation. The regions R_i are among the unknowns of the optimization problem (3) and therefore need to be updated during our iterative solution procedure. To represent them, we introduce the level set functions ψ_i so that $R_i = \{\psi_i\}$, $i = 1, \ldots, n$.

Then, the first sum of (3) can be rewritten as

$$\sum_{i=1}^{n} \int_{\Omega} [\alpha + \beta\phi(\kappa_i)]|\nabla\psi_i|\delta(\psi_i)dx, \tag{5}$$

where $\kappa_i = \nabla \cdot (\frac{\nabla\psi_i}{|\nabla\psi_i|})$ is the curvature of any level curve, and $\delta(x)$ is the derivative of the Heaviside function

$$H(x) = \begin{cases} 1, & , > 0, \\ 0, & , \leq 0, \end{cases} \tag{6}$$

in the distribution sense, or the Dirac delta function. Of course, both in the derivations below and in our numerical implementation, H and δ are taken to be smooth approximations to the Heaviside and delta functions.

On the other hand, since $R_i^{'} = R_i - \bigcup_{j<i} R_j$, i.e., $R_i^{'} = \{\psi_i > 0\} \bigcap (\bigcup_{j<i} \{\psi_j \leq 0\})$, and $R_{n+1}^{'} = \Omega - \bigcup_{j<n+1} R_j = \bigcap_{j=1}^{n} (\Omega - R_j)$, i.e., $R_{n+1}^{'} = \bigcap_{j=1}^{n} \{\psi_j < 0\}$, we may rewrite the second sum of (3) as

$$\sum_{i=1}^{n} \Big\{ \int_{\Omega} (f - c_i)^2 H(\psi_i) \prod_{j=1}^{i-1} (1 - H(\psi_j))dx \Big\} + \int_{\Omega} (f - c_{n+1})^2 \prod_{j=1}^{n} (1 - H(\psi_j))dx, \tag{7}$$

where H is the Heaviside function.

Combining (5) and (7), we obtain the NMS functional in a level set based formulation as follows:

$$\begin{aligned} E = & \sum_{i=1}^{n} \int_{\Omega} [\alpha + \beta\phi(\kappa_i)]|\nabla\psi_i|\delta(\psi_i)dx \\ & + \sum_{i=1}^{n} \Big\{ \int_{\Omega} (f - c_i)^2 H(\psi_i) \prod_{j=1}^{i-1} (1 - H(\psi_j))dx \Big\} \\ & + \int_{\Omega} (f - c_{n+1})^2 \prod_{j=1}^{n} (1 - H(\psi_j))dx. \end{aligned} \tag{8}$$

We remind the reader that here ψ_i are the level set functions describing the regions occupied by the n objects in the scene, κ_i are the curvatures of their boundaries, c_i are the (unknown) average gray-scale intensities within these regions, α and β are two positive parameters, Ω is the image domain, and f is the given image.

3.2. Derivation of the gradient descent flows. To derive the gradient descent flows for minimizing the functional (8), we here consider only the curvature related terms, since the fidelity terms are easy to handle. For simplicity, let us consider a functional $\widehat{E}(\psi) = \int_\Omega \Psi(\kappa_\psi)|\nabla\psi|\delta(\psi)dx$, where $\Psi(x)$ can be a general function. In this paper, $\Psi(x) = \alpha + \beta\phi(x)$ as in the functional (8). Derivations of gradient descent flows for some similar functionals can be found in [8, 12]. We emphasize that the functional we consider here is an integral along a contour instead of on the whole domain Ω.

Before deriving the gradient descent flows of ψ according to the minimization of $\widehat{E}(\psi)$, we use observations made in [8] in the following calculations. The authors of [8] computed gradient descent flows for minimizing similar functionals.

LEMMA 1. $\overrightarrow{n} \in R^2$ $I, P_{\overrightarrow{n}} : R^2 \to R^2$ $I(\overrightarrow{v}) = \overrightarrow{v}$ $P_{\overrightarrow{n}}(\overrightarrow{v}) = (\overrightarrow{v}\cdot\overrightarrow{n})\overrightarrow{n}$ $\overrightarrow{v} \in R^2$ $\psi : R^2 \times [0, +\infty) \to R^1$ $\kappa = \nabla\cdot(\frac{\nabla\psi}{|\nabla\psi|})$ $\frac{\partial\kappa}{\partial t} = \nabla\cdot[(I - P_{\overrightarrow{n}})\frac{\nabla\psi_t}{|\nabla\psi|}]$ $\overrightarrow{n} = \frac{\nabla\psi}{|\nabla\psi|}$

We simply compute

$$\frac{\partial\kappa}{\partial t} = \nabla\cdot\left(\frac{\partial}{\partial t}\frac{\nabla\psi}{|\nabla\psi|}\right) = \nabla\cdot\left[\frac{\nabla\psi_t}{|\nabla\psi|} - \left(\frac{\nabla\psi_t}{|\nabla\psi|}\cdot\frac{\nabla\psi}{|\nabla\psi|}\right)\frac{\nabla\psi}{|\nabla\psi|}\right] = \nabla\cdot\left[(I - P_{\overrightarrow{n}})\frac{\nabla\psi_t}{|\nabla\psi|}\right]. \qquad \square$$

LEMMA 2. 1 $(I - P_{\overrightarrow{n}})\overrightarrow{v}\cdot\overrightarrow{w} = (I - P_{\overrightarrow{n}})\overrightarrow{w}\cdot\overrightarrow{v}$ $\overrightarrow{v}, \overrightarrow{w} \in R^2$

We may thus obtain the following theorem.

THEOREM 1. $\widehat{E}(\psi) = \int_\Omega \Psi(\kappa)|\nabla\psi|\delta(\psi)dx$ $\Psi : R^1 \to R^1$ $\Psi'(0) = 0$ P $P_{\frac{\nabla\psi}{|\nabla\psi|}}$

$$\frac{\partial\psi}{\partial t} = |\nabla\psi|\nabla\cdot\left[\frac{\nabla\psi}{|\nabla\psi|}\Psi(\kappa) - \frac{1}{|\nabla\psi|}(I - P)(\nabla(\Psi'(\kappa)|\nabla\psi|))\right], \tag{9}$$

$\frac{\partial\psi}{\partial\overrightarrow{\nu}}|_{\partial\Omega} = 0$ $\kappa_\psi|_{\partial\Omega} = 0$ $\overrightarrow{\nu}$ $\partial\Omega$

To get the associated gradient descent flow, let us take the derivative of $\widehat{E}$ according to time t and then determine the derivative of ψ with respect to t by decreasing the functional $\widehat{E}$.

In fact, by Lemma 1, we have

$$\begin{aligned}
\frac{d\widehat{E}}{dt} &= \int_\Omega [\Psi'(\kappa)\kappa_t|\nabla\psi|\delta(\psi) + \Psi(\kappa)|\nabla(\psi)|_t\delta(\psi) + \Psi(\kappa)|\nabla(\psi)|\delta'(\psi)\psi_t]dx \\
&= \int_\Omega \left[\Psi'(\kappa)\nabla\cdot\left[(I - P)\left(\frac{\nabla\psi_t}{|\nabla\psi|}\right)\right]|\nabla\psi|\delta(\psi) + \Psi(\kappa)\nabla\psi_t\cdot\frac{\nabla\psi}{|\nabla\psi|}\delta(\psi)\right]dx \\
&\quad + \int_\Omega \Psi(\kappa)|\nabla(\psi)|\delta'(\psi)\psi_t dx \\
&= \int_\Omega \nabla\cdot\left[(I - P)\left(\frac{\nabla\psi_t}{|\nabla\psi|}\right)\Psi'(\kappa)|\nabla\psi|\delta(\psi)\right]dx + \int_\Omega \nabla\cdot\left[\psi_t\frac{\nabla\psi}{|\nabla\psi|}\Psi(\kappa)\delta(\psi)\right]dx \\
&\quad - \int_\Omega \left\{(I - P)\left(\frac{\nabla\psi_t}{|\nabla\psi|}\right)\cdot\nabla(\Psi'(\kappa)|\nabla\psi|\delta(\psi)) + \psi_t\nabla\cdot\left[\frac{\nabla\psi}{|\nabla\psi|}\Psi(\kappa)\delta(\psi)\right]\right\}dx \\
&\quad + \int_\Omega \Psi(\kappa)|\nabla(\psi)|\delta'(\psi)\psi_t dx. \qquad (10)
\end{aligned}$$

By the Neumann boundary condition, i.e., $\frac{\partial \psi}{\partial \overrightarrow{\nu}} = \nabla\psi \cdot \overrightarrow{\nu} = 0$, we see that $\nabla\psi_t \cdot \overrightarrow{\nu} = (\nabla\psi \cdot \overrightarrow{\nu})_t - \nabla\psi \cdot \overrightarrow{\nu}_t = 0$; therefore by applying the Green–Gauss theorem, we may drop the two divergence terms and get

$$\begin{aligned}\frac{d\widehat{E}}{dt} &= \int_\Omega -(I-P)\left(\frac{\nabla\psi_t}{|\nabla\psi|}\right)\cdot\nabla(\Psi'(\kappa)|\nabla\psi|\delta(\psi))dx \\ &\quad - \int_\Omega \left\{\psi_t\nabla\cdot\left[\frac{\nabla\psi}{|\nabla\psi|}\Psi(\kappa)\delta(\psi)\right] - \Psi(\kappa)|\nabla(\psi)|\delta'(\psi)\psi_t\right\}dx. \qquad (11)\end{aligned}$$

By Lemma 2, $(I-P)\overrightarrow{v}\cdot\overrightarrow{w} = \overrightarrow{v}\cdot(I-P)\overrightarrow{w}$, and we have

$$\begin{aligned}\frac{d\widehat{E}}{dt} &= \int_\Omega \Big\{ -\frac{\nabla\psi_t}{|\nabla\psi|}\cdot(I-P)\nabla(\Psi'(\kappa)|\nabla\psi|\delta(\psi)) \\ &\quad -\psi_t\nabla\cdot\left[\frac{\nabla\psi}{|\nabla\psi|}\Psi(\kappa)\delta(\psi)\right]\Big\}dx + \int_\Omega \Psi(\kappa)|\nabla(\psi)|\delta'(\psi)\psi_t dx \\ &= \int_\Omega \Big\{ -\nabla\cdot\left[\psi_t\frac{1}{|\nabla\psi|}(I-P)\nabla(\Psi'(\kappa)|\nabla\psi|\delta(\psi))\right] \\ &\quad +\psi_t\nabla\cdot\left[\frac{1}{|\nabla\psi|}(I-P)\nabla(\Psi'(\kappa)|\nabla\psi|\delta(\psi))\right]\Big\}dx \\ &\quad -\int_\Omega \psi_t\left[\nabla\cdot\left(\frac{\nabla\psi}{|\nabla\psi|}\Psi(\kappa)\right)\delta(\psi) + \frac{\nabla\psi}{|\nabla\psi|}\Psi(\kappa)\cdot\delta'(\psi)\nabla\psi\right]dx \\ &\quad +\int_\Omega \Psi(\kappa)|\nabla(\psi)|\delta'(\psi)\psi_t dx. \qquad (12)\end{aligned}$$

Notice that $\nabla(\Psi'(\kappa)|\nabla\psi|\delta(\psi)) = \nabla(\Psi'(\kappa)|\nabla\psi|)\delta(\psi) + \Psi'(\kappa)|\nabla\psi|\delta'(\psi)\nabla\psi$; then $(I-P)\nabla(\Psi'(\kappa)|\nabla\psi|\delta(\psi)) = \delta(\psi)(I-P)\nabla(\Psi'(\kappa)|\nabla\psi|)$, since $(I-P)\nabla\psi = 0$. Therefore, by applying the Green–Gauss theorem again and noting that $\Psi'(\kappa)|_{\partial\Omega} = 0$, we drop the divergence part in (12) and obtain

$$\begin{aligned}\frac{d\widehat{E}}{dt} &= \int_\Omega \psi_t\nabla\cdot\left[\frac{1}{|\nabla\psi|}(I-P)\nabla(\Psi'(\kappa)|\nabla\psi|)\delta(\psi)\right]dx - \int_\Omega \psi_t\nabla\cdot\left[\frac{\nabla\psi}{|\nabla\psi|}\Psi(\kappa)\right]\delta(\psi)dx \\ &= \int_\Omega \psi_t\Bigg\{\nabla\cdot\left[\frac{1}{|\nabla\psi|}(I-P)\nabla(\Psi'(\kappa)|\nabla\psi|)\right]\delta(\psi) \\ &\quad +\frac{1}{|\nabla\psi|}(I-P)\nabla(\Psi'(\kappa)|\nabla\psi|)\cdot\delta'(\psi)\nabla\psi\Bigg\}dx - \int_\Omega \psi_t\nabla\cdot\left[\frac{\nabla\psi}{|\nabla\psi|}\Psi(\kappa)\right]\delta(\psi)dx \\ &= \int_\Omega \psi_t\delta(\psi)\nabla\cdot\left[\frac{1}{|\nabla\psi|}(I-P)\nabla(\Psi'(\kappa)|\nabla\psi|) - \frac{\nabla\psi}{|\nabla\psi|}\Psi(\kappa)\right]dx. \qquad (13)\end{aligned}$$

Therefore, if we choose $\psi_t = -\delta(\psi)\nabla\cdot[\frac{1}{|\nabla\psi|}(I-P)(\nabla(\Psi'(\kappa)|\nabla\psi|)) - \frac{\nabla\psi}{|\nabla\psi|}\Psi(\kappa)]$, we see $\frac{d\widehat{E}}{dt} \le 0$. Moreover, by employing the standard technique which replaces $\delta(\psi)$ by $|\nabla\psi|$ in [31] (the factor $|\nabla\psi|$ also helps to accelerate the evolution process as discussed in [22]), we then have the gradient descent flow:

$$\frac{\partial\psi}{\partial t} = |\nabla\psi|\nabla\cdot\left[\frac{\nabla\psi}{|\nabla\psi|}\Psi(\kappa) - \frac{1}{|\nabla\psi|}(I-P)(\nabla(\Psi'(\kappa)|\nabla\psi|))\right]. \qquad (14)$$

□

Indeed, replacing $\delta(\psi)$ by $|\nabla\psi|$ does not change the property that the flow decreases the energy.

1. We do not know if there is a rigorous well posedness theory for the gradient flow equations obtained above. The factor $\delta(\psi)$ is replaced with $|\nabla\psi|$ mostly for numerical reasons, but this might also make the equations better behaved and perhaps theoretically more approachable. Nevertheless, in this paper we restrict ourselves to a numerical study of these flows. Some analytical results on related problems can be found in [13].

Let us turn to the original NMS functional (8). From Theorem 1, the system of the gradient flows reads

$$\begin{aligned}\frac{\partial\psi_i}{\partial t} &= |\nabla\psi_i|\nabla\cdot\left[\frac{\nabla\psi_i}{|\nabla\psi_i|}\Psi(\kappa_i) - \frac{1}{|\nabla\psi_i|}(I - P_{\frac{\nabla\psi_i}{|\nabla\psi_i|}})(\nabla(\Psi^{'}(\kappa_i)|\nabla\psi_i|))\right]\\ &\quad -|\nabla\psi_i|(f-c_i)^2\prod_{j=1}^{i-1}(1-H(\psi_j))\\ &\quad +|\nabla\psi_i|\sum_{s=i+1}^{n+1}\left\{(f-c_s)^2H(\psi_s)\prod_{j=1}^{i-1}(1-H(\psi_j))\prod_{j=i+1}^{s-1}(1-H(\psi_j))\right\},\end{aligned} \tag{15}$$

with the boundary conditions $\frac{\partial\psi_i}{\partial\overrightarrow{\nu}}|_{\partial\Omega} = 0$, $\kappa_i|_{\partial\Omega} = 0$, $i = 1,\ldots,n$, where $H(\psi_{n+1}) = 1$.

4. Numerical implementation. In this section, we show how to discretize each term in (15) and explain how to apply Smereka's semi-implicit method to solve the equations.

4.1. Discretization of the gradient descent flows. We first discuss the schemes for the fourth order terms in (15). The discretization is essentially the same as the one in [8]. We include its details for the sake of completeness of the present discussion.

As in [8], we rewrite them as follows:

$$\frac{\partial\psi}{\partial t} = |\nabla\psi|\nabla\cdot\overrightarrow{V}, \tag{16}$$

where $\overrightarrow{V} = (V_1, V_2) = \frac{\nabla\psi}{|\nabla\psi|}\Psi(\kappa) - \frac{1}{|\nabla\psi|}(I-P)(\nabla(\Psi^{'}(\kappa)|\nabla\psi|))$.

Then, denoting $\Psi^{'}(\kappa)|\nabla\psi|$ simply by A, we get

$$V_1 = \Psi(\kappa)\frac{\psi_x}{|\nabla\psi|} - \frac{\psi_y}{|\nabla\psi|^3}[A_x\psi_y - A_y\psi_x] \tag{17}$$

and

$$V_2 = \Psi(\kappa)\frac{\psi_y}{|\nabla\psi|} - \frac{\psi_x}{|\nabla\psi|^3}[A_y\psi_x - A_x\psi_y], \tag{18}$$

where $A_x = \partial A/\partial x$, $A_y = \partial A/\partial y$.

Let (i,j) denote the pixel locations, h be the spatial step size, and the time be discretized to be $\{ndt\}$, where $n = 0,1,2,\ldots$, and dt is the time step. Then $\psi^n_{i,j}$ refers to the value of ψ at pixel (i,j) at time ndt.

Let us discuss the discretization of each term in (16) to (18). In (16), we approximate $|\nabla\psi|$ by central difference as

$$|\nabla\psi|_{i,j} = \sqrt{\left[\frac{\psi_{(i+1,j)} - \psi_{(i-1,j)}}{2h}\right]^2 + \left[\frac{\psi_{(i,j+1)} - \psi_{(i,j-1)}}{2h}\right]^2}. \tag{19}$$

As for the terms in (17), (18), since

$$\begin{aligned}\nabla \cdot \overrightarrow{V}_{(i,j)} &= V_{1x(i,j)} + V_{2y(i,j)} \\ &= \frac{V_{1(i+1/2,j)} - V_{1(i-1/2,j)}}{h} + \frac{V_{2(i,j+1/2)} - V_{2(i,j-1/2)}}{h},\end{aligned} \tag{20}$$

we need to calculate the values of V_1 at the x-half-pixel $(i+1/2, j)$, while the values of V_2 at the y-half-pixel $(i, j+1/2)$.

We approximate the terms in (17) at $(i+1/2, j)$ as follows:

$$\begin{aligned}\kappa &= \text{minmod}(\kappa_{(i,j)}, \kappa_{(i+1,j)}), \\ \psi_x &= \frac{\psi_{(i+1,j)} - \psi_{(i,j)}}{h}, \\ \psi_y &= \text{minmod}\left(\frac{\psi_{(i+1,j+1)} - \psi_{(i+1,j-1)}}{2h}, \frac{\psi_{(i,j+1)} - \psi_{(i,j-1)}}{2h}\right), \\ |\nabla\psi| &= \sqrt{\psi_x^2 + \psi_y^2},\end{aligned} \tag{21}$$

$$\begin{aligned}A_x &= \frac{A_{(i+1,j)} - A_{(i,j)}}{h} \\ &= \frac{\Psi'(\kappa_{(i+1,j)})|\nabla\psi|_{(i+1,j)} - \Psi'(\kappa_{(i,j)})|\nabla\psi|_{(i,j)}}{h}, \\ A_y &= \text{minmod}\Bigg(\frac{\Psi'(\kappa_{(i+1,j+1)})|\nabla\psi|_{(i+1,j+1)} - \Psi'(\kappa_{(i+1,j-1)})|\nabla\psi|_{(i+1,j-1)}}{2h}, \\ &\qquad \frac{\Psi'(\kappa_{(i,j+1)})|\nabla\psi|_{(i,j+1)} - \Psi'(\kappa_{(i,j+1)})|\nabla\psi|_{(i,j-1)}}{2h}\Bigg),\end{aligned} \tag{22}$$

where

$$\text{minmod}(x, y) = \frac{\text{sgn}(x) + \text{sgn}(y)}{2}\min(|x|, |y|).$$

Of course, there are other ways to discretize the terms than using the minmod function (see Chapter 16 in [17]).

Similarly, we may approximate the terms in (18).

It is subtle to discretize the remaining terms in (15). Specifically, one should be careful to approximate $|\nabla\psi|$. In this paper, we use the technique that is discussed in [29] (see Chapter 6). The detail can be found in our previous work [32]. We omit them here.

Moreover, we approximate the Heaviside function $H(x)$ by a standard regularized function as in [9], which reads

$$H_\epsilon(x) = \frac{1}{2}\left(1 + \frac{2}{\pi}\arctan\left(\frac{x}{\epsilon}\right)\right),$$

where ϵ is a fixed small number. In the experiments, we choose ϵ as small as h^2, where h is the mesh size.

2. The gradient descent flow (15) is in essence a degenerate parabolic equation. It is like a higher order version of the more familiar mean curvature flow. Our discretization is similar to the ones used for this type of parabolic equations [8].

4.2. Smereka's semi-implicit method. Since the equations in (15) are fourth order and parabolic, we would expect the Courant–Friedrichs–Levy (CFL) condition (see, for instance, [15]) for an explicit scheme to require the time step size to be proportional to the fourth power of spatial step size, i.e., $dt \sim dx^4$. This restriction would result in an intolerably long time for the minimization process. To ease the time step restriction, we will apply a semi-implicit discretization method due to Smereka [30], which significantly accelerates the convergence of the gradient descent.

For simplicity, as in [30], let us denote the gradient descent flows as

$$\psi_t = S(\psi); \tag{23}$$

then, we can rewrite it as

$$\psi_t = -\lambda\Delta^2\psi + \lambda\Delta^2\psi + S(\psi), \tag{24}$$

where Δ^2 is a bi-Laplacian (since equations (15) are fourth order equations) and λ is a positive parameter. Then we may discretize (24) in time as

$$\psi^{n+1} - \psi^n = dt(-\lambda\Delta^2\psi^{n+1} + \lambda\Delta^2\psi^n + S(\psi^n)),$$

and then

$$\psi^{n+1} = \psi^n + dt(1 + dt\lambda\Delta^2)^{-1}S(\psi^n). \tag{25}$$

The operator $(1 + dt\lambda\Delta^2)$ is positive definite, and it can be efficiently inverted by the fast Fourier transform (FFT).

Therefore, in our case, we need only to calculate the values on the right side of (15) at time nh as $S(\psi^n)$ for each function ψ_i, and then we can get the values of ψ_i at the new time $(n+1)h$ by using FFT.

Smereka's semi-implicit method will relax the CFL condition considerably, since the operator Δ^2 helps to smooth possible high oscillating waves. Numerical experiments in the following section also show that we may choose a much larger time step size than what is required by CFL condition.

4.3. Procedure for minimizing NMS functional. During the evolution of the variables ψ_i according to (15), the unknown intensities c_i also need to be updated. Variations of the objective energy E with respect to the c_i yield the following optimality conditions:

$$c_i = \frac{\int_\Omega f H(\psi_i)\prod_{j=1}^{i-1}(1 - H(\psi_j))}{\int_\Omega H(\psi_i)\prod_{j=1}^{i-1}(1 - H(\psi_j))}, \qquad i = 1, \ldots, n, \tag{26}$$

and

$$c_{n+1} = \frac{\int_\Omega f \prod_{j=1}^{n}(1 - H(\psi_j))}{\int_\Omega \prod_{j=1}^{n}(1 - H(\psi_j))}. \tag{27}$$

They are just the averages of the gray-scale intensities on the appropriate unoccluded domains.

Then, given the initial guesses for the level set functions ψ_i and the intensity constants c_i, we iteratively solve the equations (15) and (26), (27).

Now, let us discuss how to choose the initial guesses. We first employ any multiphase segmentation method, for instance, Chan–Vese's method [10], to segment the

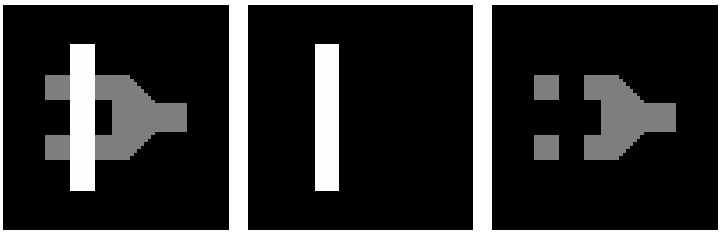


FIG. 2. *Bar and Fork. From left to right, these figures are the original image and the two regions obtained from standard segmentation model.*

image domain Ω into several regions in each of which the intensity is homogenous, and thus obtain the associated level set function and intensity constant for each region. Then, as in [11], we use these results as the initial guesses for solving (15) and (26), (27). Of course, there may be other and potentially better ways to choose initial conditions.

Moreover, as usual, during the gradient descent evolutions given above it becomes necessary to periodically reinitialize the level set functions so that they are close to being signed distance functions of the contours they represent (for reinitialization, see [29]). It is not necessary to do this very often, and in fact we observed that reinitializations, if done too frequently and not accurately enough, can interfere with the fourth order evolution.

5. Numerical experiments. In this section, we present numerical results of applying our technique on some synthetic images, including the typical example (Bar and Fork) that appears in [11].

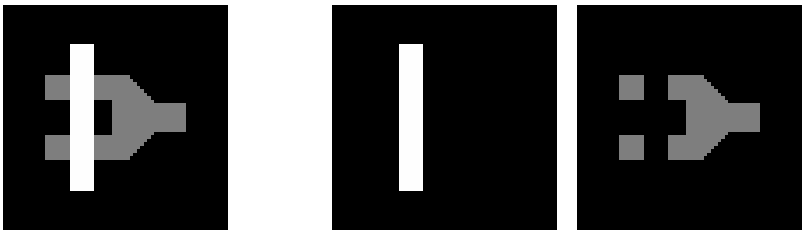
5.1. Two regions and a background. We here show two examples: Bar and Fork, and Ellipse and Disk. For each example, the intensities for the two regions and the background are 1.0, 0.5, and 0, respectively.

The first example is an image with a bar and a broken fork inside a background (Figure 2). For this case, there are only two possible orderings: the bar is in front of the fork, or vice versa. We minimize the NMS functional under both of these order hypothesis separately. The way that we build the hypothesis into the computation is by the assignment of initial conditions for the regions, i.e., in one of the computations, we take the bar to be the initial guess for the region that is in front and the broken fork to be the initial guess for the region that is behind, and we switch them for the other case. The values are listed as follows:

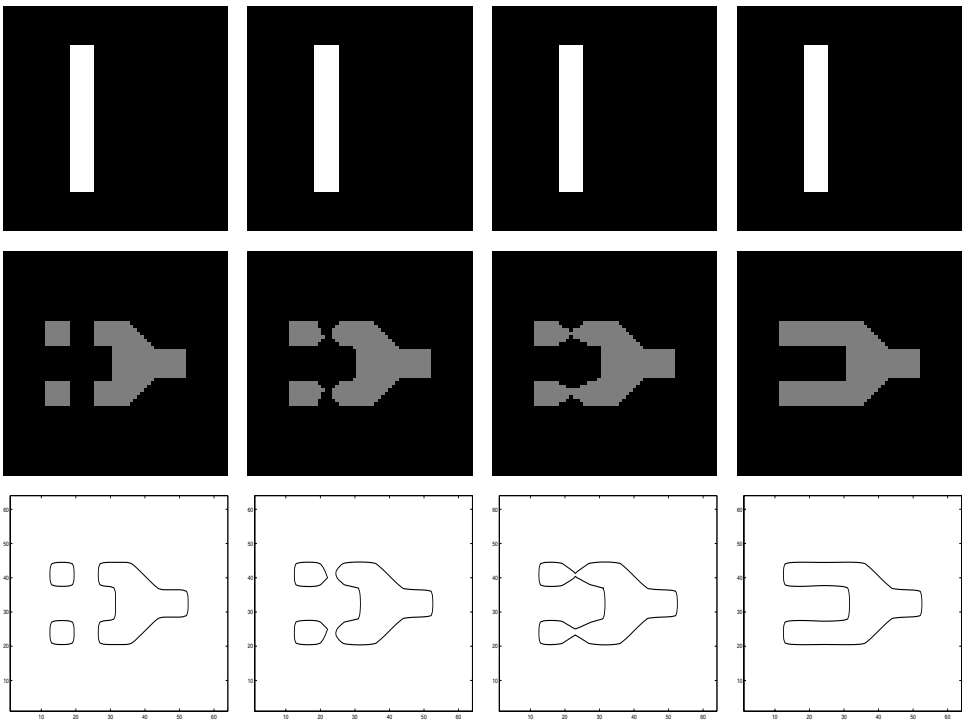
- Bar front, Fork behind, the NMS functional = 0.1762.
- Fork front, Bar behind, the NMS functional = 0.2718.

This shows that the assumption that the bar is in front of the fork yields a smaller value of the NMS functional, the scene is made up of a fork that is behind and occluded by a vertical bar.

In Figure 3, we list the evolution process of the two regions under the assumption of the first ordering. From these groups of figures, one can see how the broken fork is connecting while the bar is kept stable; note especially the behavior of its corners. It



Original image Two regions by segmentation procedure



Initial Intermediate Intermediate Final

FIG. 3. *The evolution process under the assumption that the Bar is in front of the Fork. In the second and third row, the evolutions of the two regions by solving* (15) *are listed. The fourth row shows how the boundary of the Fork is growing. During the process, the front region occupied by the Bar is keeping stable while the region by the Fork is connecting behind the Bar. In this experiment, the parameters are* $\alpha = 0.00005$, $\beta = 0.005$, *and* $\lambda = 0.1$ *in Smereka's semi-implicit method.*

is the choice of $\phi(x) = |x|$ when $|x|$ big in (3) that preserves the corners. The choice $\phi(x) = x^2$ would inevitably smear those corners.

We omit the evolution process under the second assumption since the two regions are kept stable.

The restriction on the time step size has been obtained numerically, by checking whether energy is decreased or not. When Smereka's semi-implicit method is not used, the result is consistent with what one would expect, based on the fact that the gradient descent is given essentially by a fourth order (degenerate) parabolic PDE, namely, the step size has to be proportional to dx^4.

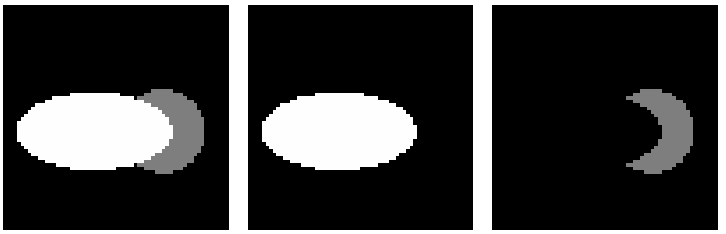
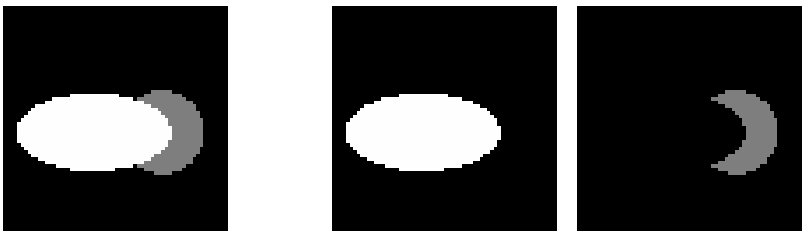


FIG. 4. *Ellipse and Crescent. From left to right, these figures are the original image, and the two regions obtained from standard segmentation model.*



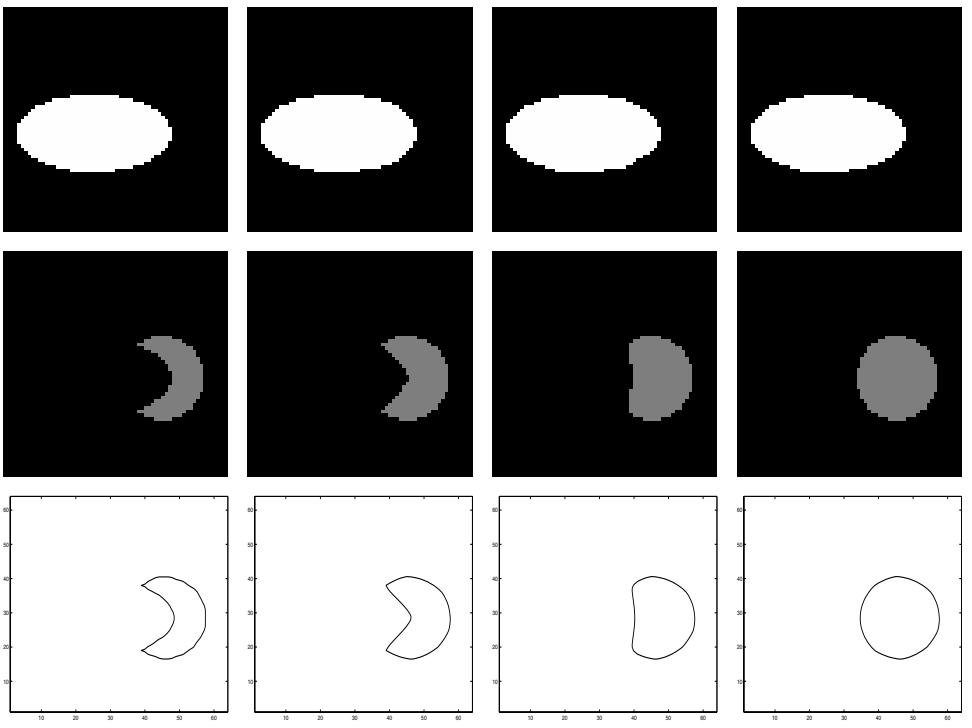


FIG. 5. *The evolution process under the assumption that the Ellipse is in front of the Crescent. In the second and third row, the evolutions of the two regions by solving* (15) *are listed. The fourth row shows how the boundary of the Crescent is growing. During the process, the front region occupied by the Ellipse is keeping stable while the region by the Crescent is connecting behind the Ellipse. In this experiment, the parameters are* $\alpha = 0.00005$, $\beta = 0.01$, *and* $\lambda = 0.1$ *in Smereka's semi-implicit method.*

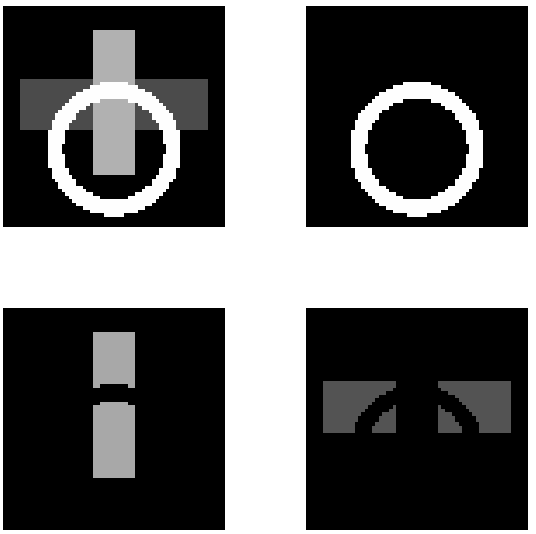


FIG. 6. *An annulus and two bars. The top left is the original image, and the other three figures show the regions obtained from the original image by standard segmentation model.*

The second example is an image with an ellipse and a crescent on top of a dark background (Figure 4). We calculate the values of the NMS functional for the two ordering as follows:

- Ellipse front, Crescent behind, the NMS functional = 0.1404.
- Crescent front, Ellipse behind, the NMS functional = 0.2812.

This shows that the ellipse is in front of the disk (instead of crescent!). In Figure 5, we list the evolution process of the two regions under the first assumption of ordering and we still omit the process under the second assumption. From these groups of figures, one can see how the crescent is growing to be a full moon while the front ellipse is kept stable.

5.2. Three regions and a background. We present here an example with an annulus and two bars inside a background (Figure 6). The intensities for them are 1.0, 0.9, 0.8, and 0, respectively.

Since there are three objects inside the image, $3! = 6$ cases of the ordering should be considered. Here we list the evolution process under the ordering assumption that the Annulus is in front of the vertical Bar while the horizontal Bar is behind the vertical Bar, which yields the smallest value of the NMS functional (Figure 7).

Of course, if there are more than just a few objects (n objects) in the given image, the number of minimizations ($n!$ cases of the ordering) that need to be done quickly becomes unrealistic. However, as pointed out in [11], it would be reasonable to localize the computation by restricting attention to smaller sections of the image at a time. If these sections are small enough, one would not expect to find more than a few objects present in them. But then it would be necessary to combine the results obtained from different sections in a suitable way.

6. Conclusion. In this paper, we develop a level set based technique to numerically minimize the NMS functional that appears in the variational formulation of the segmentation with depth problem. In contrast to [11], we can handle more general dependence of the functionals on curvature, which allows us to remain faithful to

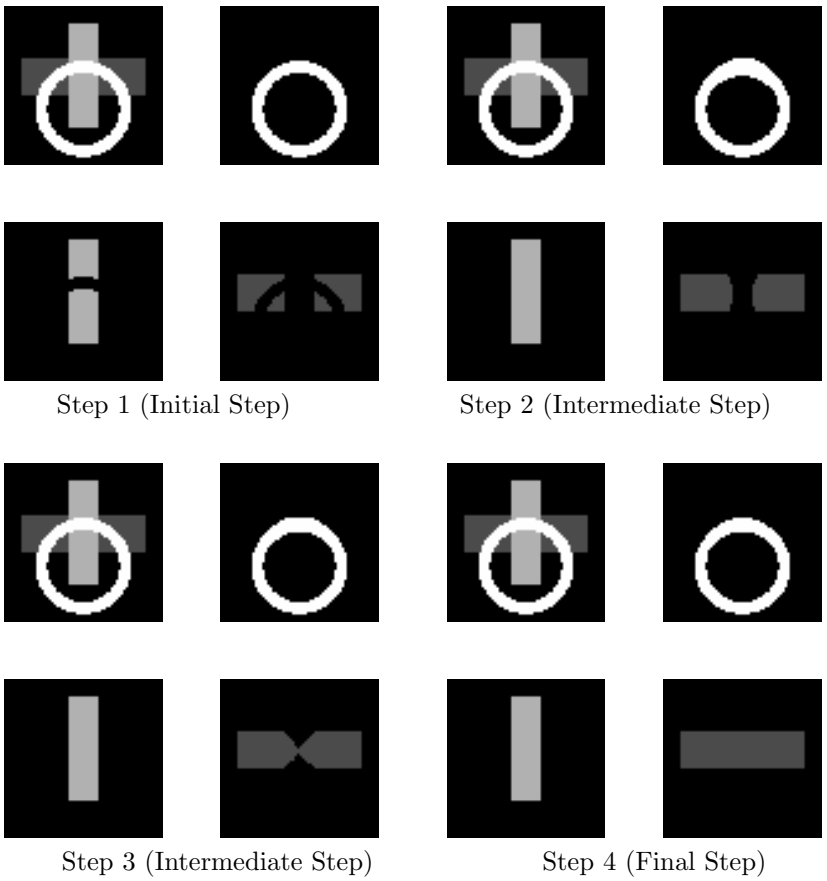


FIG. 7. *The evolution process under the ordering assumption that the Annulus, the vertical Bar, and the horizontal Bar are listed from the nearest to the farthest. In each group of figures, the left top is the original image while the others show the evolution of each region. During this process, the Annulus is almost keeping stable while the vertical Bar is connecting and then the horizontal Bar. In this experiment, the parameters are* $\alpha = 0.00005$, $\beta = 0.03$, *and* $\lambda = 0.1$ *in Smereka's semi-implicit method.*

the original functional of NMS, and therefore avoid the smearing and stability issues. Moreover, level set methods seem to provide better contour localization at the same spatial resolution than the phase-field method used in [11]. One disadvantage of our technique developed in this paper is that the resulting equations are more nonlinear, more degenerate, and therefore more difficult to solve than the ones that result from the phase-field approach based on De Giorgi's conjecture in [11]. However, there is progress in addressing the computational expense of level set methods in such higher order problems, such as Smereka's semi-implicit method, which we have utilized.

We study only some synthetic images in this paper. Our results on these images constitute a promising step toward effective minimization of those challenging functionals such as the NMS functional. In particular, based on our results, we would expect our algorithm to be successful with scenes that have a few large, distinct objects in them, such as the ones considered by Nitzberg, Mumford, and Shiota in [25, 26].

Acknowledgment. The authors thank Stanley Osher, who suggested to us the idea of a level set implementation of the NMS model.

REFERENCES

[1] L. Ambrosio and V. M. Tortorelli, *Approximation of functionals depending on jumps by elliptic functions via Gamma convergence,* Comm. Pure Appl. Math., 43 (1990), pp. 999–1036.

[2] G. Bellettini, *Variational approximation of functionals with curvatures and related properties,* J. Convex Anal., 4 (1997), pp. 91–108.

[3] G. Bellettini and R. March, *Variational properties of a model for image segmentation with overlapping regions,* in Variational Methods for Discontinuous Structures, G. Dal Maso and F. Tomarelli, eds., Prog. Nonlinear Differential Equations Appl. 51, Birkhäuser Boston, Boston, 2002, pp. 9–17.

[4] G. Bellettini and R. March, *An image segmentation variational model with free discontinuities and contour curvature,* Math. Models Methods Appl. Sci., 14 (2004), pp. 1–45.

[5] A. Braides and R. March, *Approximation by Γ-convergence of a curvature-depending functional in visual reconstruction,* Comm. Pure Appl. Math., 59 (2006), pp. 71–121.

[6] V. Caselles, R. Kimmel, and G. Sapiro, *Geodesic active contour,* Internat. J. Comput. Vision, 22 (1997), pp. 61–79.

[7] V. Caselles, F. Catté, T. Coll, and F. Dibos, *A geometric model for active contours in image processing,* Numer. Math., 66 (1993), pp. 1–32.

[8] T. F. Chan, S. H. Kang, and J.-H. Shen, *Euler's elastica and curvature-based inpainting,* SIAM J. Appl. Math., 63 (2002), pp. 564–592.

[9] T. Chan and L. A. Vese, *Active contours without edges,* IEEE Trans. Image Process. 10 (2001), pp. 266–277.

[10] T. Chan and L. A. Vese, *A level set algorithm for minimizing the Mumford–Shah functional in image processing,* in the IEEE Workshop on Variational and Level Set Methods, Vancouver, Canada, 2001, pp. 161–168.

[11] S. Esedoḡlu and R. March, *Segmentation with depth but without detecting junctions,* J. Math. Imaging Vision, 18 (2003), pp. 7–15.

[12] S. Esedoḡlu and J. Shen, *Digital inpainting based on the Mumford–Shah–Euler image model,* European J. Appl. Math., 13 (2002), pp. 353–370.

[13] Y. Giga and K. Ito, *Loss of convexity of simple closed curves moved by surface diffusion,* in Topics in Nonlinear Analysis, The Herbert Amann Anniversary volume, J. Escher and G. Simonett, eds., Prog. Nonlinear Differential Equations Appl. 35, Birkhäuser, Basel, 1999, pp. 305–320.

[14] E. De Giorgi, *Some remarks on convergence and least squares methods,* in Composite Media and Homogenization Theory, G. Dal Maso and G. F. Dell's Antonio, eds., Birkhäuser Boston, Boston, 1991, pp. 135–142.

[15] B. Gustafsson, H. O. Kreiss, and J. Oliger, *Time Dependent Problems and Difference Methods,* Wiley-Interscience, New York, 1995.

[16] M. Kass, A. Witkin, and D. Terzopoulos, *Snakes: Active contour models,* Internat. J. Comput. Vision, 1 (1988), pp. 321–331.

[17] R. J. Leveque, *Numerical Methods for Conservation Laws,* Birkhäuser, Basel, 1992.

[18] G. Dal Maso, *An Introduction to Gamma-Convergence,* Birkhäuser Boston, Boston, 1993.

[19] R. March, *Visual reconstruction with discontinuities using variational methods,* Image and Vision Computing, 10 (1992), pp. 30–38.

[20] R. March and M. Dozio, *A variational method for the recovery of smooth boundaries,* Image and Vision Computing, 15 (1997), pp. 705–712.

[21] R. Malladi, J. A. Sethian, and B. C. Vemuri, *Shape modeling with front propagation: A level set approach,* IEEE Trans. Pattern Analysis and Machine Intelligence, 17 (1995), pp. 158–175.

[22] A. Marquina and S. Osher, *Explicit algorithms for a new time dependent model based on level set motion for nonlinear deblurring and noise removal,* SIAM J. Sci. Comput., 22 (2000), pp. 387–405.

[23] D. Mumford and J. Shah, *Optimal approximation by piecewise smooth functions and associated variational problems,* Comm. Pure Appl. Math., 42 (1989), pp. 577–685.

[24] J. M. Morel and S. Solimini, *Variational Methods in Image Segmentation,* Birkhäuser Boston, Boston, 1995.

[25] M. Nitzberg and D. Mumford, *The* 2.1*D sketch,* in Proceedings of the Third International Conference on Computer Vision, Osaka, Japan, 1990, pp. 138–144.

[26] M. Nitzberg, D. Mumford, and T. Shiota, *Filtering, Segmentation, and Depth,* Lecture Notes in Comput. Sci. 662, Springer-Verlag, Berlin, 1993.

[27] S. Osher and J. A. Sethian, *Fronts propagating with curvature-dependent speed-algorithm based on Hamilton–Jacobi formulations,* J. Comput. Phys., 79 (1988), pp. 12–49.

[28] S. Osher and R. Fedkiw, *Level set methods: An overview and some recent results,* J. Comput. Phys., 169 (2001), pp. 463–502.

[29] S. Osher and R. Fedkiw, *Level Set Methods and Dynamic Implicit Surfaces,* Springer-Verlag, New York, 2002.

[30] P. Smereka, *Semi-implicit level set methods for curvature and surface diffusion motion,* J. Sci. Comput., 19 (2003), pp. 439–456.

[31] H. K. Zhao, T. Chan, B. Merriman, and S. Osher, *A variational level set approach to multiphase motion,* J. Comput. Phys., 127 (1996), pp. 179–195.

[32] W. Zhu and T. Chan, *Capturing Illusory Contours: A Level Set Approach,* UCLA CAM Report 03-65, University of California, Los Angeles, 2003.

SIAM J. SCI. COMPUT.
Vol. 28, No. 5, pp. 1974–1999
© 2006 Society for Industrial and Applied Mathematics

ON THE ACOUSTIC SINGLE LAYER POTENTIAL: STABILIZATION AND FOURIER ANALYSIS*

A. BUFFA† AND S. SAUTER‡

Abstract. In this paper, we propose a general approach for stabilizing the single layer potential for the Helmholtz boundary integral equation and prove its stability. We consider Galerkin boundary element discretizations and analyze their convergence. Furthermore, we derive quantitative error bounds for the Galerkin discretization which are explicit with respect to the mesh width *and* the wave number for the special case that the surface is the unit sphere in $\mathbb{R}^3$. We perform then a qualitative analysis which allows us to choose the stabilization such that the (negative) influence of the wave number in the stability and convergence estimates attains its minumum.

Key words. acoustic scattering, Galerkin boundary element method, stabilization of Helmholtz equation, high wave number

AMS subject classifications. 1A10, 35S30, 35P25, 65R20, 65R30

DOI. 10.1137/040615110

1. Introduction. In this paper, we will address problems related to the discretization of boundary integral equations for the Helmholtz problem outside of a reflecting obstacle Ω^-, where $\Omega^- \subset \mathbb{R}^3$ is a bounded Lipschitz domain. Let $\Omega^+ = \mathbb{R}^3 \setminus \bar{\Omega}^-$ and $\mathcal{L}_k := -\Delta - k^2$. We consider the following problem: Find $u^+ \in H^1_{\mathrm{loc}}(\Omega^+)$ such that the Helmholtz problem

$$\begin{aligned} \mathcal{L}_k u^+ &= 0 && \text{in } \Omega^+, \\ u^+ &= g && \text{on } \Gamma := \partial\Omega^-, \\ \left| \frac{\partial u}{\partial r} - \mathrm{i}\, k u \right| &\le C \left\| \mathbf{x} \right\|^{-2} && \left\| \mathbf{x} \right\| \to \infty \end{aligned} \tag{1.1}$$

is satisfied in a weak sense (cf. [26]). Here, $\partial/\partial r$ denotes the derivative in radial direction $\mathbf{x}/\left\|\mathbf{x}\right\|$.

Our goal is to solve these equations by the method of integral equations. A potential ansatz leads to a boundary integral equation on Γ for the unknown density φ which is of the form $R_k \varphi = g$. Here, R_k is the trace V_k of the single layer potential associated to $\mathcal{L}_k$ on Γ or a stabilized version of it. We will consider the Galerkin boundary element method for its discretization. It is well known that the V_k is not invertible on a countable set of frequencies k (see, e.g., [11]) and we will introduce a class of stabilizations such that the boundary integral equation is well posed for all frequencies $k > 0$.

Alternatively, the Helmholtz equation (1.1) can be solved numerically by finite element discretizations where the problem related to the unbounded domain Ω^+ is treated either by [illegible] elements or by introducing an artificial outer boundary far away from the scatterer. It is well known (see, e.g., [3]) that finite element discretiza-

*Received by the editors September 15, 2004; accepted for publication (in revised form) May 24, 2006; published electronically November 24, 2006.
http://www.siam.org/journals/sisc/28-5/61511.html
†IMATI-CNR, Via Ferrata 1, 27100 Pavia, Italy (annalisa@imati.cnr.it).
‡Institut für Mathematik, Universität Zürich, Winterthurerstr. 190 CH-8057 Zürich, Switzerland (stas@amath.unizh.ch).

tions for the Helmholtz problem suffer from the [illegible]; i.e., the constants in the Galerkin error estimates deteriorate to infinity with increasing wave number $k > 0$. Hence, the question arises whether this pollution effect is possibly reduced by solving the boundary integral equation for the Helmholtz problem via the Galerkin boundary element method. In order to address this question, it is mandatory to first remove the forbidden frequencies of the single layer potential through a suitable stabilization.

In the literature, various approaches exist for stabilizing these integral equations (cf. [31], [4], [10], [2], [14], [20], [9]). Among them the so-called Brakhage–Werner formulation for the stabilization of the acoustic double layer potential is one of the most popular.

Here we introduce a class of stabilizations for the single layer potential for which the well-posedness of the resulting continuous and discrete equations (for mesh size sufficiently small) can be proved (a related work is [8]). The results can be summarized as follows:

a. The stabilized acoustic single layer potential, on the continuous level, admits a unique solution which depends continuously on the data for general Lipschitz surfaces. This is a strong advantage compared to the Brakhage–Werner stabilization, where the question of existence and uniqueness is open for general Lipschitz surfaces and even for piecewise smooth surfaces.
b. The Galerkin method converges for "sufficiently small" step size on general triangulated surfaces with optimal rate.

Indeed, as in all stabilization approaches, well-posedness and quasi-optimality can be proved provided the step size is "sufficiently small." More precisely, the threshold for the maximal step size such that the Galerkin discretization is stable depends on the wave number, and the "constant" in the quasi-optimality error estimate typically deteriorates to infinity as the wave number increases.

Consequently, in order to compare different approaches from the viewpoint of numerical efficiency the following questions have to be addressed:

1. How does the threshold for the stability of the Galerkin discretization quantitatively depend on the wave number?
2. How does the Galerkin error quantitatively depend on the mesh width [illegible] the wave number?
3. Can the stabilization approach be implemented efficiently in a boundary element code? What is the computational complexity?

These questions have been discussed for the Brakhage–Werner stabilization in [17] (see also [24]). Here, we analyze quantitatively the dependence of the constants entering the stability and convergence estimates for our class of stabilized single layer potentials, in the case that the surface is the unit sphere in $\mathbb{R}^3$. For this case, we obtain the following:

c. The condition for the stability of Galerkin method (related to the condition "the step size has to be sufficiently small") is slightly more restrictive as for the stabilization in the Brakhage–Werner approach.
d. The constant of quasi-optimality in the Galerkin error estimates which amplifies the error of the best approximation is $k^{1/3}$ for both the stabilized acoustic single layer potential and the Brakhage–Werner stabilization.

In this light, the Fourier analysis in this paper shows that, for the surface of the unit sphere, the stabilized acoustic single layer potential has similar convergence properties as the Brakhage–Werner stabilization.

However, we consider the result that the stabilized acoustic single layer potential is stable also on surfaces of general Lipschitz polyhedra as the essential advantage compared to the Brakhage–Werner formulation, where the stability on general Lipschitz polyhedra is still an open question.

The paper is organized as follows. In section 2, we will formulate the boundary integral equation for the Helmholtz problem and introduce our abstract stabilization approach. Concrete stabilization operators which satisfy the abstract assumptions will be presented as examples. As a side result, we will prove, for some sesquilinear forms associated to integral operators of general fractional order, continuity and ellipticity in appropriate Sobolev spaces.

In section 3, we will introduce the Galerkin boundary element method with piecewise constant boundary elements for the stabilized single layer integral equation.

In section 4, the stability and convergence of the Galerkin boundary element method will be analyzed. It will be proved that the discretization is stable on general Lipschitz polyhedrons and the Galerkin solution converges at an optimal rate, provided the step size is sufficiently small.

In section 5, we will employ Fourier analysis to explicitly analyze the dependence of the stability and convergence of the stabilized Galerkin method with respect to both the mesh size and the wave number. We will discuss three parameter constellations by asymptotic analysis in a rigorous way. The intermediate ranges of the parameters are studied by systematic computer experiments and show that the asymptotic cases are relevant for the estimates of the constants of interest. In this light, the asymptotic analysis [illegible] that the estimates of the constants of interest cannot be improved while the computer experiments [illegible] that these constants do not behave worse in the intermediate ranges of the parameters.

Section 6 briefly discusses the computational complexity of the proposed stabilization approaches.

2. Boundary integral formulation.

2.1. Sobolev spaces on Lipschitz manifolds and trace operators. In this section, we will introduce some notation related to Sobolev spaces and recall some of their properties. As before, Ω^- denotes a bounded Lipschitz domain in $\mathbb{R}^3$, Ω^+ its complement, and $\mathbf{n}$ the unit normal vector pointing from Ω^- to Ω^+.

We make use of standard complex Sobolev spaces in the whole space $H^s(\mathbb{R}^3)$, in the domains $H^s(\Omega^\pm)$, $s \in \mathbb{R}$, and on the boundary Γ, $H^\ell(\Gamma)$, $\ell \in [-1,1]$. On the boundary, $(\cdot,\cdot)_0$ is the $L^2(\Gamma)$ scalar product, i.e., $(u,v)_0 = \int_\Gamma u\bar{v}$, which is identified with its continuous extension to the duality pairing between $H^s(\Gamma)$ and $H^{-s}(\Gamma)$, $s \in (0,1]$.

Moreover, for any positive s, we denote by $H^s_{\mathrm{loc}}(\mathbb{R}^3)$, $H^s_{\mathrm{loc}}(\Omega^+)$ the space of functions which are locally in H^s and by $H^{-s}_{\mathrm{comp}}(\mathbb{R}^3)$, $H^{-s}_{\mathrm{comp}}(\Omega^+)$ their dual spaces. Furthermore, we introduce

$$H^{s+1}_\Delta(\Omega^\pm) := \{u \in H^{s+1}(\Omega^\pm) \ : \ \Delta u \in H^s(\Omega^\pm)\}. \tag{2.1}$$

The standard one-sided trace operators are denoted by γ_0^+, γ_0^- and they map $\gamma_0^\pm : H^s(\Omega^\pm) \to H^{s-1/2}(\Gamma)$ continuously for all $s \in (1/2, 3/2)$. The one-sided normal derivative trace operators associated to the mapping $u \mapsto \partial_{\mathbf{n}} u$ are denoted by γ_1^+, γ_1^- and they are continuous operators from $H^{s+1}_\Delta(\Omega^\pm)$ to $H^{s-1/2}(\Gamma)$ for $s \in [0, 1/2)$. Note that the ranges of $\gamma_0^\pm$ and $\gamma_1^\pm$ do not change if the spaces $H^s(\Omega^+)$ are replaced by $H^s_{\mathrm{loc}}(\Omega^+)$ everywhere.

For later use, we will define Sobolev spaces of order $H^{1+s}(\Gamma)$ for $s > 0$ as the ranges of the trace operator γ_0 applied to functions in $H^{3/2+s}(\Omega^-)$. More precisely, for $s \in (0, \frac{1}{2}]$, we define

$$H^{1+s}(\Gamma) := \gamma_0(H^{3/2+s}(\Omega^-)),$$

$$\|v\|_{H^{1+s}(\Gamma)} := \inf_{u \in H^{3/2+s}(\Omega^-), \gamma_0(u)=v} \|u\|_{H^{3/2+s}(\Omega^-)},$$

and we denote by $H^{-(1+s)}(\Gamma)$, $s \in (0, 1/2]$, the dual space of $H^{1+s}(\Gamma)$ with $L^2(\Gamma)$ as pivot space. The corresponding duality pairing is denoted again by $(\cdot, \cdot)_0$. We will often use the shorthand notation $\|\cdot\|_s$ for $\|\cdot\|_{H^s(\Gamma)}$.

Since we shall deal with the numerical discretization of boundary integral operators via the boundary element method, it is reasonable to assume that the Lipschitz surface Γ is piecewise smooth.

NOTATION 2.1. Γ Γ $\Gamma_j \subset \Gamma$ $1 \le j \le J < \infty$ $\Gamma = \bigcup_{j=1}^{J} \overline{\Gamma}_j$

PROPOSITION 2.2. Γ $H^{1+s}(\Gamma_j)$ $s \in (0, \frac{1}{2}]$ j

$$H^{1+s}(\Gamma) \equiv \{v \in H^1(\Gamma) \ : \ v_{|\Gamma_j} \in H^{1+s}(\Gamma_j)\}, \quad s \in (0, \tfrac{1}{2}]. \tag{2.2}$$

The proof can be found in, e.g., [13] or [7].

2.2. Boundary integral operators. For $z \in \mathbb{R}^3 \setminus \{0\}$, let $G_k(z) := \frac{e^{\mathrm{i}\, k \|z\|}}{4\pi \|z\|}$ and define the associated single layer and double potential by

$$(S_k \varphi)(x) := \int_\Gamma G_k(x-y)\, \varphi(y)\, ds_y, \qquad x \in \mathbb{R}^3 \setminus \Gamma,$$

$$(D_k \varphi)(x) := \int_\Gamma \left(\frac{\partial}{\partial n_y} G_k(x-y) \right) \varphi(y)\, ds_y, \qquad x \in \mathbb{R}^3 \setminus \Gamma.$$

The restrictions of these operators to Ω^+ (resp., Ω^-) are denoted by S_k^+, D_k^+ (resp., S_k^-, D_k^-).

The boundary integral operators associated with the single and double layer potentials are given by $V_k := \gamma_0 S_k$ and $K_k := \{\gamma_0\} D_k := \frac{1}{2}\left(\gamma_0^+ D_k + \gamma_0^- D_k\right)$.

It is well known that every solution $\varphi \in H^{-1/2}(\Gamma)$ of

$$V_k \varphi = g \tag{2.3}$$

has the property that $u^- = S_k^- \varphi$, respectively, $u^+ = S_k^+ \varphi$, satisfies the homogeneous interior, respectively, exterior, Helmholtz problem.

However, for countably many wave numbers k, the boundary integral equation (2.3) is not injective and, hence, does not admit a solution for all right-hand sides g (although, e.g., the exterior Helmholtz problem (1.1) admits a unique solution for all boundary data $g \in H^{1/2}(\Gamma)$).

Our goal is to modify the boundary integral equation so that the problem admits a unique solution for all wave numbers. Our approach follows the framework of [31].

We start with the formal ansatz

$$R_k = V_k + \mathrm{i}\,\eta(\tfrac{1}{2}I + K_k)\gamma_0 B, \tag{2.4}$$

where $B : H^{-1/2}(\Gamma) \to H^{1/2}(\mathbb{R}^3)$ is, for the moment, any linear and continuous operator. We consider the equation

$$R_k\varphi = g. \tag{2.5}$$

In Proposition 2.3, we will prove that, under suitable conditions on B, (2.5) admits a unique solution for all wave numbers, and the functions

$$u^+ = \left(S_k^+ + \mathrm{i}\,\eta D_k^+\gamma_0 B\right)\varphi, \quad \text{resp.,} \quad u^- = \left(S_k^- + \mathrm{i}\,\eta D_k^-\gamma_0 B\right)\varphi \tag{2.6}$$

satisfy the homogeneous exterior, respectively, interior, Helmholtz problem.

PROPOSITION 2.3. [illegible] $\gamma_0 B : H^{-1/2}(\Gamma) \to H^{1/2}(\Gamma)$ [illegible] $(\cdot, \gamma_0 B\cdot)_0 : H^{-1/2}(\Gamma) \times H^{-1/2}(\Gamma) \to \mathbb{C}$ [illegible] $\varphi \in H^{-1/2}(\Gamma)$ [illegible] $(\varphi, \gamma_0 B\varphi)_0 > 0 \Leftrightarrow \varphi \neq 0$ [illegible] (2.5) [illegible] $k \in \mathbb{R}_+$ [illegible] $\eta \in \mathbb{R} \setminus \{0\}$ [illegible] u^+ [illegible] u^- [illegible] (2.6) [illegible] Ω^+ [illegible] Ω^-

For the proof, see [11] or [8].

There are various ways of choosing an operator B in (2.4) satisfying the assumptions in Proposition 2.3. However, to obtain an efficient numerical scheme it is essential that the complexity of the numerical realization of B is moderate and the implementation does not cause too much extra work. We will present in section 2.3 some choices of B and will comment on the numerical complexity in section 6. All these choices will satisfy the following assumption.

[illegible] 2.4. There exists $0 < \varepsilon \leq 1$, such that the following hold:

1. The operator $\gamma_0 B : H^{-1/2-\varepsilon}(\Gamma) \to H^{1/2+\varepsilon}(\Gamma)$ is continuous.
2. The sesquilinear form $(\cdot, \gamma_0 B\cdot)_0 : H^{-1/2-\varepsilon}(\Gamma) \times H^{-1/2-\varepsilon}(\Gamma) \to \mathbb{C}$ is hermitian and $H^{-1/2-\varepsilon}(\Gamma)$-elliptic: There exists $\alpha_B > 0$ such that

$$(\varphi, \gamma_0 B\varphi)_0 \geq \alpha_B \|\varphi\|^2_{-1/2-\varepsilon} \qquad \forall \varphi \in H^{-1/2-\varepsilon}(\Gamma).$$

Note that Assumption 2.4 implies that the assumptions in Proposition 2.3 are satisfied.

2.3. Choices of B. In this section, we will present various choices for the operator B.

2.3.1. $B = S_0 V_0$. The single layer potential for the operator $-\Delta$ is given by

$$(S_0\varphi)(x) := \int_\Gamma \frac{\varphi(y)}{4\pi\|x-y\|} ds_y, \qquad x \in \mathbb{R}^3.$$

It is well known (see [12], [21]) that $V_0 := \gamma_0 S : H^{-1/2+s}(\Gamma) \to H^{1/2+s}(\Gamma)$ is a linear and continuous isomorphism for all $s \in \left[-\frac{1}{2}, \frac{1}{2}\right]$ and, for $s = 0$, the associated sesquilinear form $(\cdot, V_0\cdot)_0 : H^{-1/2}(\Gamma) \times H^{-1/2}(\Gamma) \to \mathbb{C}$ is elliptic. Hence,

$$V_0^2 : H^{-1}(\Gamma) \to H^1(\Gamma)$$

is continuous and

$$\left(\varphi, V_0^2\varphi\right)_0 = \|V_0\varphi\|_0^2 \geq c\|\varphi\|^2_{H^{-1}(\Gamma)},$$

where we have used that $V_0 : L^2(\Gamma) \to H^1(\Gamma)$ is an isomorphism.

Hence, the choice $B = S_0 V_0$ leads to $\gamma_0 B = V_0^2$ which satisfies Assumption 2.4 with $\varepsilon = 1/2$.

2.3.2. Single layer potential for $(I-\Delta)^{1+\varepsilon}$. The fundamental solution of the pseudodifferential operator $(I-\Delta)^{1+\varepsilon}$, $0 < \varepsilon \leq 1$, is given by (cf. [30, Example 2.2])

$$\mathcal{G}_\varepsilon(z) := \frac{2^{-\varepsilon}}{(2\pi)^{3/2}\,\Gamma(1+\varepsilon)}\,\|z\|^{\varepsilon-1/2}\,\mathrm{K}_{\varepsilon-1/2}(\|z\|)\,, \tag{2.7}$$

where K_ν is the modified Bessel function (cf. [1, sec. 9.6]) and $\Gamma(\cdot)$ denotes the Gamma function. The corresponding potential is given by

$$(B_\varepsilon \varphi)(x) := \int_\Gamma \mathcal{G}_\varepsilon(x-y)\,\varphi(y)\,ds_y, \qquad x \in \mathbb{R}^3. \tag{2.8}$$

Next, we will prove that the operator $\gamma_0 B_\varepsilon : H^{-1/2-\varepsilon}(\Gamma) \to H^{1/2+\varepsilon}(\Gamma)$ is continuous and the associated sesquilinear form $(\cdot, \gamma_0 B_\varepsilon \cdot)_0 : H^{-1/2-\varepsilon}(\Gamma) \times H^{-1/2-\varepsilon}(\Gamma) \to \mathbb{C}$ is hermitian and $H^{-1/2-\varepsilon}(\Gamma)$-elliptic, i.e., satisfies Assumption 2.4 for any chosen value of $\varepsilon \in]0,1] \setminus \{1/2\}$. (The case $\varepsilon = 1/2$ is exceptional only for nonsmooth surfaces and this problem is related to the mapping properties of γ_0 applied to functions in $H^{3/2}(\mathbb{R}^3)$.)

THEOREM 2.5 (mapping properties). [illegible] $1/2 < \ell \leq 2$ $\ell \neq 3/2$ [illegible] $\varepsilon \in]0,1]$ [illegible] $B_\varepsilon : H^{1/2-\ell}(\Gamma) \to H^{2+2\varepsilon-\ell}(\mathbb{R}^3)$ [illegible] $\varepsilon \in]0,1] \setminus \{1/2\}$ [illegible] $\gamma_0 B_\varepsilon : H^{-1/2-\varepsilon}(\Gamma) \to H^{1/2+\varepsilon}(\Gamma)$ [illegible] $\varepsilon = 1/2$

[illegible]. We introduce the Bessel potential for the operator $(I-\Delta)^{1+\varepsilon}$:

$$N_\varepsilon u(x) := \int_{\mathbb{R}^3} \mathcal{G}_\varepsilon(x-y)\,u(y)\,dy, \qquad x \in \mathbb{R}^3, \tag{2.9}$$

with the fundamental solution $\mathcal{G}_\varepsilon$ as in (2.7). Its symbol is given by $\sigma(\xi) = (\|\xi\|^2 + 1)^{m/2}$ with $m = -(2+2\varepsilon)$. Hence, Theorem 1.4' in [15] implies the mapping property $N_\varepsilon : H^s(\mathbb{R}^3) \to H^{s+2+2\varepsilon}(\mathbb{R}^3)$ for all $s \in \mathbb{R}$. Since, for all $1/2 < \ell \leq 2$, $\ell \neq 3/2$, the trace operator $\gamma_0 : H^\ell(\mathbb{R}^3) \to H^{-1/2+\ell}(\Gamma)$ is continuous, its dual $\gamma_0' := H^{1/2-\ell}(\Gamma) \to H^{-\ell}(\mathbb{R}^3)$ is continuous in the same range of ℓ. Thus, the representation $B_\varepsilon = N_\varepsilon \gamma_0'$ implies the continuity of

$$B_\varepsilon : H^{1/2-\ell}(\Gamma) \to H^{2+2\varepsilon-\ell}(\mathbb{R}^3)\,. \tag{2.10}$$

The mapping property of $\gamma_0 B_\varepsilon$ follows from this and the mapping properties of the trace operator γ_0. □

THEOREM 2.6. [illegible] $\varepsilon \in]0,1] \setminus \{1/2\}$ [illegible] $(\cdot, \gamma_0 B_\varepsilon \cdot)_0 : H^{-1/2-\varepsilon}(\Gamma) \times H^{-1/2-\varepsilon}(\Gamma) \to \mathbb{C}$ [illegible] $H^{-1/2-\varepsilon}(\Gamma)$ [illegible] $\varepsilon = 1/2$

[illegible]. Let $\varepsilon \in]0,1]$ with the exclusion of the case $\varepsilon = 1/2$ for nonsmooth surfaces. Let $(\cdot,\cdot)_{0,\mathbb{R}^3}$ denote the continuous extension of the $L^2(\mathbb{R}^3)$-scalar product to the duality pairing $H^{-1-\varepsilon}(\mathbb{R}^3) \times H^{1+\varepsilon}(\mathbb{R}^3)$. The relation $B_\varepsilon = N_\varepsilon \gamma_0'$ and the mapping properties of B_ε imply

$$(\varphi, \gamma_0 B_\varepsilon \varphi)_0 = (\varphi, \gamma_0 N_\varepsilon \gamma_0' \varphi)_0 = (\gamma_0' \varphi, N_\varepsilon \gamma_0' \varphi)_{0,\mathbb{R}^3} \qquad \forall \varphi \in H^{-1/2-\varepsilon}(\Gamma)\,. \tag{2.11}$$

Simple properties of the Fourier transform lead to

(2.12)
$$(\gamma_0'\varphi, N_\varepsilon\gamma_0'\varphi)_{0,\mathbb{R}^3} = \left(\widehat{\gamma_0'\varphi}, \frac{\widehat{\gamma_0'\varphi}}{\left(1+\|\cdot\|^2\right)^{1+\varepsilon}}\right)_{0,\mathbb{R}^3}$$
$$= \left(\frac{\widehat{\gamma_0'\varphi}}{\left(1+\|\cdot\|^2\right)^{\frac{1+\varepsilon}{2}}}, \frac{\widehat{\gamma_0'\varphi}}{\left(1+\|\cdot\|^2\right)^{\frac{1+\varepsilon}{2}}}\right)_{0,\mathbb{R}^3} \geq c\,\|\gamma_0'\varphi\|^2_{H^{-1-\varepsilon}(\mathbb{R}^3)}.$$

Let $R := \left\{\gamma_0'(\varphi) : \varphi \in H^{-1/2-\varepsilon}(\Gamma)\right\}$ denote the range of γ_0'. Next we will show that $\gamma_0' : H^{-1/2-\varepsilon}(\Gamma) \to R$ is an isomorphism.

It is well known that $\gamma_0 : H^{t+1/2}(\mathbb{R}^3) \to H^t(\Gamma)$ is surjective for $t \in (0, 3/2] \setminus \{1\}$, and hence, trivially, γ_0 has closed range. From the surjectivity of γ_0 we conclude the injectivity of γ_0' and the closed range theorem [34, sec. VII.5] implies that R is closed in $H^{-1-\varepsilon}\left(\mathbb{R}^3\right)$. Thus, $\gamma_0' : H^{-1/2-\varepsilon}(\Gamma) \to R$ is bijective and has closed range in $H^{-1-\varepsilon}\left(\mathbb{R}^3\right)$. The open mapping theorem implies that $\gamma_0' : H^{-1/2-\varepsilon}(\Gamma) \to R$ is an isomorphism. Thus,

$$\|\gamma_0'\varphi\|^2_{H^{-1-\varepsilon}(\mathbb{R}^3)} \geq c\,\|\varphi\|^2_{-1/2-\varepsilon} \qquad \forall\varphi \in H^{-1/2-\varepsilon}(\Gamma),$$

and this, in combination with (2.11) and (2.12), yields the $H^{-1/2-\varepsilon}(\Gamma)$-ellipticity of $(\cdot, \gamma_0 B_\varepsilon \cdot)_{0,\Gamma}$.

The sesquilinear form $(\cdot, \gamma_0 B_\varepsilon \cdot)_{0,\Gamma}$ is hermitian because the kernel function $\mathcal{G}_\varepsilon$ in (2.7) is real valued (cf. [1, sec. 9.6.1]) and symmetric; i.e., $\mathcal{G}_\varepsilon(x-y) = \mathcal{G}_\varepsilon(y-x)$. □

2.7. The choice $\varepsilon = 1$ in (2.3.2) yields that B_1 is the single layer potential for the biharmonic operator $\mathcal{L}_{\mathrm{i}}^2 = (-\Delta + I)(-\Delta + I)$, and in this case the fundamental solution becomes

$$\mathcal{G}_1(z) = \frac{e^{-\|z\|}}{8\pi}. \tag{2.13}$$

2.8. A further choice for the operator $\gamma_0 B$ in (2.4) is the inverse of the Laplace–Beltrami operator. This operator satisfies, as V_0^2, Assumption 2.4 with $\varepsilon = 1/2$ (cf. [9] and [8]).

3. Galerkin boundary element method. The continuous problem which we are going to solve numerically is given by seeking $\varphi \in H^{-1/2}(\Gamma)$ such that

$$(R_k\varphi, \psi)_0 = (g, \psi)_0 \qquad \forall\psi \in H^{-1/2}(\Gamma), \tag{3.1}$$

where R_k is as in (2.4) and $g \in H^{1/2}(\Gamma)$ is a given right-hand side.

Our goal is to solve the problem (3.1) by the Galerkin boundary element method and we start by defining the relevant boundary element space.

Let Γ be a polyhedral surface. Let $\mathcal{T} = \{\tau_1, \tau_2, \ldots, \tau_N\}$ denote a shape-regular triangulation of Γ, and let X_h be the boundary element space

$$X_h := \operatorname{span}\{b_\tau : \tau \in \mathcal{T}\} \tag{3.2}$$

with the indicator function $b_\tau : \Gamma \to \mathbb{R}$ for the triangle τ. The mesh width is denoted by

$$h := \max_{\tau \in \mathcal{T}} h_\tau \quad \text{with} \quad h_\tau := \operatorname{diam} \tau.$$

The discrete problem is given by seeking $\varphi_h \in X_h$ such that

$$(R_k \varphi_h, \psi)_0 = (g, \psi)_0 \qquad \forall \psi \in X_h. \tag{3.3}$$

4. Stability and error analysis for the Galerkin boundary element method. We start with the well-known approximation property of piecewise constant boundary elements on a shape-regular triangulation. In this light, we assume from now on that Γ is a polyhedral Lipschitz surface.

THEOREM 4.1. $\mathcal{T}$ Γ h C_A $\mathcal{T}$ $-1/2 \le s \le 0$ $s \le t \le 1$

$$\inf_{\psi \in X_h} \|\varphi - \psi\|_s \le C_A h^{t-s} \|\varphi\|_t \qquad \forall \varphi \in H^t(\Gamma).$$

We need now to recall the well-known mapping properties for the operators V_k and $\gamma_0^+ D_k$.

For polyhedral surfaces, we denote by s_Γ the regularity exponent associated with Γ such that for all s, $|s| < s_\Gamma$, the operators

$$V_k : H^{-1/2+s}(\Gamma) \to H^{+1/2+s}(\Gamma) \quad \text{and} \quad \gamma_0^+ D_k = \tfrac{1}{2} I + K_k : H^{1/2+s}(\Gamma) \to H^{1/2+s}(\Gamma) \tag{4.1}$$

are continuous. Note that, for polyhedral surfaces, we may choose $s_\Gamma = 1/2$. For smooth surfaces of class C^∞, the choice $s_\Gamma = \infty$ is allowed and, moreover, the operator K_k is of order -1, namely,

$$K_k : H^{1/2+s}(\Gamma) \to H^{3/2+s}(\Gamma) \quad \forall\, s \in \mathbb{R},\ \Gamma \text{ in } C^\infty. \tag{4.2}$$

As a consequence of these facts and of Proposition 2.3, the operator

$$R_k : H^{-1/2+s}(\Gamma) \to H^{1/2+s}(\Gamma) \text{ is an isomorphism } \forall |s| < s_\Gamma. \tag{4.3}$$

The stability and convergence analysis will be based on a splitting of R_k into its principal part $V_0 + \frac{\mathrm{i}\eta}{2}\gamma_0 B_\varepsilon$ and the compact perturbation $\tilde{R}_k := V_0 + \frac{\mathrm{i}\eta}{2}\gamma_0 B_\varepsilon - R_k$. It is well known that the boundary integral operator V_0 for the single layer potential of the Laplacian is coercive, i.e., there is a constant $\alpha_0 > 0$ such that

$$(V_0 \varphi, \varphi)_0 \ge \alpha_0 \|\varphi\|^2_{-1/2} \qquad \forall \varphi \in H^{-1/2}(\Gamma). \tag{4.4}$$

Assumption 2.4 implies that the sesquilinear form $\left(\cdot, \frac{1}{2}\gamma_0 B_\varepsilon \cdot\right)_0$ is hermitian and

$$\left(\varphi, \frac{1}{2}\gamma_0 B_\varepsilon \varphi\right)_0 \ge \alpha_B \|\varphi\|^2_{-1/2-\varepsilon} \ge 0 \qquad \forall \varphi \in H^{-1/2}(\Gamma).$$

We conclude that $V_0 + \frac{1}{2}\gamma_0 B_\varepsilon$ is $H^{-1/2}$-coercive

$$\operatorname{Re}\left(\left(V_0 + \frac{\mathrm{i}\eta}{2}\gamma_0 B_\varepsilon\right)\varphi, \varphi\right)_0 \ge \alpha_0 \|\varphi\|^2_{-1/2} \qquad \forall \varphi \in H^{-1/2}(\Gamma), \tag{4.5}$$

where α_0 is as in (4.4) and, in particular, independent of ε.

In order to describe the mapping properties of the compact perturbation $\tilde{R}_k$ we introduce the interval I_{shift} which depends on the smoothness of the surface by

$$I_{\text{shift}} := \begin{cases} [0, 2\varepsilon] \cap [0, s_\Gamma[& \text{if } \Gamma \text{ is a general polyhedral surface,} \\ \left[0, \min\left\{\tfrac{3}{2}, 1+2\varepsilon\right\}\right] & \text{if } \Gamma \text{ is of class } C^\infty. \end{cases} \tag{4.6}$$

PROPOSITION 4.2. [illegible] $\tilde{R}_k : H^{-1/2}(\Gamma) \to H^{1/2+\mu}(\Gamma)$ [illegible] $\mu \in I_{\text{shift}}$.

[illegible]. For polyhedral surfaces, the combination of (2.10) and (4.1) yields

$$K_k \gamma_0 B : H^{-1/2}(\Gamma) \to H^{1/2+\mu}\left(\mathbb{R}^3\right) \qquad \forall \mu \in I_{\text{shift}}, \tag{4.7}$$

whereas, for surfaces of class C^∞, the combination of (2.10) and (4.2) yields

$$K_k \gamma_0 B : H^{-1/2}(\Gamma) \to H^{1/2+\mu}\left(\mathbb{R}^3\right) \qquad \forall \mu \in [0, 1+2\varepsilon]. \tag{4.8}$$

The difference $V_0 - V_k$ can be written in the form $\gamma_0 \mathcal{N}_{0,k} \gamma_0'$, where $\mathcal{N}_{0,k} : H^s_{\text{comp}}\left(\mathbb{R}^3\right) \to H^{s+4}_{\text{loc}}\left(\mathbb{R}^3\right)$ is a pseudodifferential operator of order -4 (cf. [32, Bemerkung 3.1.3]). Hence, the mapping properties of the trace operator and its dual imply the continuity of

$$V_0 - V_k : H^{-1/2}(\Gamma) \to H^{1/2+s}(\Gamma) \qquad \forall s \ : \ s \le 2, s < s_\Gamma. \qquad \square \tag{4.9}$$

Note that the proof of Proposition 4.2 allows us to replace $\frac{3}{2}$ by 2 in (4.6) for smooth surfaces. However, the proof of Theorem 4.3 will further restrict the set of admissible shifts to $\mu \le 3/2$ and we have taken this fact into account already in (4.6).

We denote by C_X the continuity constant of R_k; i.e.,

$$C_X := \sup\left\{ |(\varphi, R_k \psi)_0| : \varphi, \psi \in H^{-1/2}(\Gamma) : \|\varphi\|_{-1/2} = \|\psi\|_{-1/2} = 1 \right\}. \tag{4.10}$$

The combination of (4.3) with Proposition 4.2 yields that

$$C_\mu := \left\| (R_k^\star)^{-1} \tilde{R}_k \right\|_{H^{-1/2+\mu}(\Gamma) \leftarrow H^{-1/2}(\Gamma)} \tag{4.11}$$

is bounded for all $\mu \in I_{\text{shift}}$. The operator R_k satisfies a Gårding inequality but is not coercive. Hence, we may expect the existence of a discrete solution of (3.3) only for sufficiently small mesh width h. In this light, we define h_0 depending on $\mu \in I_{\text{shift}}$ by

$$h_0 := \left(\frac{\alpha_0}{2C_A} \frac{1}{C_X C_\mu} \right)^{1/\mu}. \tag{4.12}$$

THEOREM 4.3. [illegible] 2.4 [illegible] $0 < h < h_0$ [illegible] h_0 [illegible] (4.12) [illegible] $\mu \in I_{\text{shift}}$ [illegible] (3.3) [illegible]

$$\|\varphi - \varphi_h\|_{-1/2} \le \frac{2C_X}{\alpha_0} \inf_{\psi \in X_h} \|\varphi - \psi\|_{-1/2}. \tag{4.13}$$

[illegible] $\bigcup X_h \hookrightarrow H^{-1/2}(\Gamma)$ [illegible] $h \to 0$

. This proof is inspired by the proof of the analogue theorem for elliptic partial differential equations (see, e.g., [6, sec. 5.7]).

(a) . Assume that a discrete solution φ_h exists. The error is denoted by $e := \varphi - \varphi_h$ and can be estimated by using the Galerkin orthogonality

$$\begin{aligned}(4.14)\quad \alpha_0 \|e\|_{-1/2}^2 &\le (R_k e, e)_0 + \left(\tilde{R}_k e, e\right)_0 \le |(R_k e, \varphi - \phi_h)_0| + \left|\left(\tilde{R}_k e, e\right)_0\right| \\ &\le C_X \|e\|_{-1/2} \|\varphi - \phi_h\|_{-1/2} + \left|\left(\tilde{R}_k e, e\right)_0\right|,\end{aligned}$$

where $\phi_h \in X_h$ is the best approximation of φ with respect to the $\|\cdot\|_{-1/2}$-norm. In order to estimate the modulus of $(\tilde{R}_k e, e)_0$ we use a duality argument. Let ψ_e be the solution of the adjoint problem:

$$R_k^\star \psi_e = \tilde{R}_k e \qquad \text{in } H^{1/2}(\Gamma).$$

By the definition of the constant C_μ, we conclude that

$$\|\psi_e\|_{-1/2+\mu} \le C_\mu \|e\|_{-1/2} \qquad \forall \mu \in I_{\text{shift}}$$

holds.

Let $\psi_h \in X_h$ denote the best approximation of ψ_e with respect to the $\|\cdot\|_{-1/2}$-norm. The approximation property of piecewise constant boundary elements yields

$$\|\psi_e - \psi_h\|_{-1/2} \le C_A C_\mu \|e\|_{-1/2} h^\mu \qquad \forall \mu \in I_{\text{shift}}.$$

Galerkin's orthogonality implies

$$\begin{aligned}\left|\left(\tilde{R}_k e, e\right)_0\right| &= |(R_k^\star \psi_e, e)_0| = |(\psi_e, R_k e)_0| = |(R_k e, \psi_e - \psi_h)_0| \\ &\le C_X \|e\|_{-1/2} \|\psi_e - \psi_h\|_{-1/2} \le C_A C_X C_\mu h^\mu \|e\|_{-1/2}^2 \qquad \forall \mu \in I_{\text{shift}}.\end{aligned}$$

Using this estimate in (4.14) we obtain

$$\alpha_0 \|e\|_{-1/2}^2 \le C_X \|e\|_{-1/2} \|\varphi - \phi_h\|_{-1/2} + C_A C_X C_\mu h^\mu \|e\|_{-1/2}^2.$$

Let h_0 be as in (4.12). Then, for $h < h_0$, the estimate (4.13) holds. Note that, when Γ is a C^∞-surface, the choice $\mu = \min\left\{\frac{3}{2}, 1 + 2\varepsilon\right\}$ is allowed.

(b) . Since the discrete problem is finite dimensional it suffices to prove uniqueness. For $g = 0$, Proposition 2.3 implies that the exact solution is $\varphi = 0$. Equation (4.13) implies $\varphi_h = 0$ and this proves the uniqueness. □

. 4.4. Theorem 4.3 provides convergence of the Galerkin method for polyhedral surfaces Γ. This is a much stronger result as for the Brakhage–Werner stabilization where the stability of the resulting method is still open for polyhedral surfaces.

. 4.5. In section 5 we will investigate the dependence of the maximal mesh width h_0 on the wave number k for the special case that Γ is the unit sphere. It turns out that the operator splitting

$$C_\mu \le \left\|(R_k^\star)^{-1}\right\|_{H^{-1/2+\mu}(\Gamma) \leftarrow H^{1/2+\mu}(\Gamma)} \left\|\tilde{R}_k\right\|_{H^{1/2+\mu}(\Gamma) \leftarrow H^{-1/2}(\Gamma)}$$

and the estimate of the two factors in the right-hand side lead to too pessimistic estimates of the dependence of C_μ on k. Thus, we estimate the constant C_μ directly.

In the following, we will analyze the dependence of the Galerkin error on the wave number k for the case, where we have full regularity. In this light, we introduce the set of functions having the property that the derivatives grow proportionally with respect to the wave number k.

DEFINITION 4.6. [illegible] $\rho > 0$ [illegible] $\mathcal{O}_{\rho,k}$ [illegible] $\varphi \in H^1(\Gamma)$ [illegible]

$$\|\varphi\|_1 \leq \rho k^{3/2} \|\varphi\|_{-1/2}. \tag{4.15}$$

THEOREM 4.7. [illegible] 2.4 [illegible] Γ [illegible] φ [illegible] (2.5) [illegible] $\mathcal{O}_{\rho,k}$ [illegible] $\rho > 0$ [illegible] $0 < h < h_0$ [illegible] h_0 [illegible] (4.12) [illegible] $\mu \in I_{\text{shift}}$ [illegible] $\varphi \neq 0$ [illegible]

$$\frac{\|\varphi - \varphi_h\|_{-1/2}}{\|\varphi\|_{-1/2}} \leq \left(\frac{2C_X}{\alpha_0}\right) C_A \rho (kh)^{3/2}.$$

[illegible]. Using (4.13) together with Theorem 4.1, we obtain

$$\|\varphi - \varphi_h\|_{-1/2} \leq \left(\frac{C_X}{2\alpha_0}\right) C_A h^{3/2} \|\varphi\|_1.$$

Taking into account the oscillation condition (4.15) yields the proof. ∎

In subsection 5.7 we will consider the question of under which conditions the solution of the integral equation (2.5) belongs to the set $\mathcal{O}_{\rho,k}$.

5. The special case of $\Gamma = \mathbb{S}^2$. In this section, we will investigate the dependence of the constant C_X and the minimal mesh width h_0 (see (4.12)) upon η and k for the special case that Ω is the unit ball

$$\Omega := \left\{x \in \mathbb{R}^3 : \|x\| < 1\right\}.$$

It is well known that for $\Gamma = \mathbb{S}^2$, the Sobolev spaces can be defined via the decay properties of the Fourier coefficients.

5.1. Spherical harmonics. For a function $f \in L^2(\mathbb{S}^2)$ the Fourier coefficients are defined by

$$f_n^m := \int_{\mathbb{S}^2} Y_n^m(\hat{x}) \overline{f(\hat{x})} ds_x, \tag{5.1}$$

where Y_n^m are the spherical harmonics. We defer the reader to [1].

DEFINITION 5.1. [illegible] $f \in L^2(\mathbb{S}^2)$ [illegible] $\mathcal{H}^s(\mathbb{S}^2)$ $s \geq 0$ [illegible]

$$\sum_{n=0}^{\infty} \sum_{m=-n}^{n} |f_n^m|^2 (1+n^2)^s < \infty. \tag{5.2}$$

[illegible]

$$\langle f, g \rangle_s := \sum_{n=0}^{\infty} (1+n^2)^s \sum_{m=-n}^{n} f_n^m \overline{g_n^m} \tag{5.3}$$

[illegible] $\mathcal{H}^s(\mathbb{S}^2)$ [illegible] $s < 0$ [illegible] $\mathcal{H}^s(\mathbb{S}^2)$ [illegible] $\mathcal{H}^{-s}(\mathbb{S}^2)$ [illegible] $F \in \mathcal{H}^s(\mathbb{S}^2)$ [illegible] (5.2), [illegible] f_n^m [illegible] $f_n^m := F(Y_n^m)$

The eigenfunctions of all arising boundary integral operator on $\mathbb{S}^2$ are given by the spherical harmonics. The eigenvalues can be expressed by Bessel and related functions. Let j_n, respectively, $h_n^{(1)}$, denote the spherical Bessel functions of first and third kind (cf. [1]).

LEMMA 5.2.

a. $\mathcal{H}^s\left(\mathbb{S}^2\right)$ $s \in \mathbb{R}$

$$\left\langle Y_n^m, Y_{n'}^{m'} \right\rangle_s = \delta_{n,n'}\delta_{m,m'}\left(1+n^2\right)^s$$

$n, n' \in \mathbb{N}$ $|m| \le n$ $|m'| \le n'$.

b. R_k

i.

$$V_k Y_n^m = \lambda_{n,k}^{(V)} Y_n^m \qquad \lambda_{n,k}^{(V)} := 2\,\mathrm{i}\,k h_n^{(1)}(k)\, j_n(k).$$

ii.

$$\left(\frac{1}{2}I + K_k\right) Y_n^m = \lambda_{n,k}^{(K)} Y_n^m \qquad \lambda_{n,k}^{(K)} := \mathrm{i}\,k^2 h_n^{(1)}(k)\, j_n'(k).$$

c. $\mathcal{H}^s\left(\mathbb{S}^2\right)$ $H^s\left(\mathbb{S}^2\right)$ $\varphi \in \mathcal{H}^s\left(\mathbb{S}^2\right)$ $\varphi := (\varphi_n^m)_{\substack{n\in\mathbb{N}_0 \\ -n\le m\le n}}$ $\mathbf{H}_s := \left(\operatorname{diag}\left(1+n^2\right)^s\right)_{n\in\mathbb{N}_0}$

$$\|\varphi\|_{H^s(\mathbb{S})}^2 \sim \left|\!\left|\!\left|(\varphi_n^m)_{n,m}\right|\!\right|\!\right|_s^2 := \sum_{n=0}^{\infty} (\mathbf{H}_s)_n \sum_{m=-n}^{n} |\varphi_n^m|^2. \tag{5.4}$$

. Ad a. The result follows by using the $L^2(\mathbb{S})$-orthogonality of the spherical harmonics (cf., e.g., [29, Theorem 2.4.1]) for $f = Y_{n'}^{m'}$ in (5.1) and the definition of the scalar product (5.3).

Ad b. See [22].

Ad c. See [28, Chapter X, Theorem 6.4]. □

In view of Lemma 5.2 (c), we will use the same notation $H^s(\mathbb{S})$ for both $H^s(\mathbb{S})$ and $\mathcal{H}^s(\mathbb{S})$.

The space of sequences $(\varphi_n^m)_{\substack{n\in\mathbb{N} \\ -n\le m\le n}}$ where the right-hand side in (5.4) is finite is denoted by $\mathbf{h}_s$ and the right-hand side in (5.4) defines (the square of) the norm in $\mathbf{h}_s$.

Assumption 2.4 implies that $\gamma_0 B : H^{-1/2-\varepsilon}(\Gamma) \to H^{1/2+\varepsilon}(\Gamma)$ is continuous and elliptic. In order to develop a spectral analysis of the operator R_k we furthermore assume that, for $\Gamma = \mathbb{S}^2$, the spherical harmonics are the eigenfunctions also for the operator $\gamma_0 B$.

. 5.3. Let $\Gamma = \mathbb{S}^2$ and let Assumption 2.4 be satisfied for some $0 < \varepsilon \le 1$. The spherical harmonics are the eigenfunctions of $\gamma_0 B$:

$$(\gamma_0 B)\, Y_n^m = \lambda_n^{(B)} Y_n^m \qquad \forall n \in \mathbb{N}, -m \le n \le m.$$

There exist constants $0 < c_1 \le C_1 < \infty$ independent of n such that

$$\frac{c_1}{(n+1)^{1+2\varepsilon}} \le \lambda_n^{(B)} \le \frac{C_1}{(n+1)^{1+2\varepsilon}} \qquad \forall n \in \mathbb{N}. \tag{5.5}$$

For the choice $\gamma_0 B = V_0^2$ (cf. subsection 2.3.1) Assumption 5.3 with $\varepsilon = 1/2$ simply follows from the well-known relation $V_0 Y_n^m = (2n+1)^{-1} Y_n^m$ (cf. [29]).

THEOREM 5.4. [illegible] $\varepsilon \in]0,1]$ [illegible] B_ε [illegible] (2.8) [illegible] 5.3.

[illegible]. First, we will prove that

$$\gamma_0 B_\varepsilon Y_n^m = \lambda_n Y_n^m$$

holds. Note that

$$\gamma_0 B_\varepsilon Y_n^m = \gamma_0 N_\varepsilon \left(Y_n^m \delta_\Gamma \right), \tag{5.6}$$

where N_ε denotes the Bessel potential (cf. (2.9)) and δ_Γ is the Dirac function concentrated on Γ. For $\xi \in \mathbb{S}^2$, we write for short $Y_n^m(\xi)$ instead of $Y_n^m(\alpha,\beta)$, where α,β are the spherical angles of ξ. We employ the Fourier transform to evaluate (5.6) and obtain

$$\begin{aligned}
\widehat{N_\varepsilon \left(Y_n^m \delta_\Gamma \right)}(\hat{x}) &= \frac{1}{\left(1+\|\hat{x}\|^2\right)^{1+\varepsilon} (2\pi)^{3/2}} \int_{\mathbb{R}^3} Y_n^m\left(\frac{x}{\|x\|}\right) \delta_\Gamma(x)\, e^{-i\langle \hat{x}, x\rangle} dx \\
&= \frac{1}{\left(1+\|\hat{x}\|^2\right)^{1+\varepsilon} (2\pi)^{3/2}} \int_{\mathbb{S}^2} Y_n^m(\xi)\, e^{-i\langle \hat{x}, \xi\rangle} d\xi.
\end{aligned}$$

We make use of the formula

$$\int_{\mathbb{S}^2} e^{-i\langle x, \hat{x}\rangle} Y_\ell^m(x)\, dx = g_\ell(\|\hat{x}\|)\, Y_\ell^m\left(\frac{\hat{x}}{\|\hat{x}\|}\right) \quad \text{with} \quad g_\ell(r) = (-i)^\ell 4\pi j_\ell(r)$$

which follows by a comparison of [29, section 3.2.4, formula (3.2.44)] and [29, section 3.2.4, formula (3.2.54)] . Let $\mathcal{F}^{-1}$ denote the inverse Fourier transform. Then

$$\begin{aligned}
N_\varepsilon \left(Y_n^m \delta_\Gamma \right)(x) &= \left(\mathcal{F}_{\hat{x}}^{-1} \frac{g_n(\|\hat{x}\|)\, Y_n^m\left(\frac{\hat{x}}{\|\hat{x}\|}\right)}{\left(1+\|\hat{x}\|^2\right)^{1+\varepsilon} (2\pi)^{3/2}} \right)(x) \\
&= \frac{1}{(2\pi)^{3/2}} \int_{\mathbb{R}^3} \frac{g_n(\|\hat{x}\|)\, Y_n^m\left(\frac{\hat{x}}{\|\hat{x}\|}\right)}{\left(1+\|\hat{x}\|^2\right)^{1+\varepsilon}} e^{i\langle x, \hat{x}\rangle} d\hat{x} \\
&\overset{\hat{x} \leftarrow r\zeta}{=} \frac{1}{(2\pi)^{3/2}} \int_0^\infty \frac{r^2 g_n(r)}{(1+r^2)^{1+\varepsilon}} \left(\int_{\mathbb{S}^2} Y_n^m(\zeta)\, e^{i\langle x, r\zeta\rangle} d\zeta \right) dr \\
&= Y_n^m\left(\frac{-x}{\|x\|}\right) \frac{1}{(2\pi)^{3/2}} \int_0^\infty \frac{r^2 g_n(r)\, g_n(-\|x\|\, r)}{(1+r^2)^{1+\varepsilon}} dr.
\end{aligned}$$

Note that $Y_n^m(-\xi) = (-1)^n Y_n^m(\xi)$. Applying the trace operator to this equation yields

$$\gamma_0 B_\varepsilon Y_n^m = \lambda_n Y_n^m \quad \text{with} \quad \lambda_n := \frac{(-1)^n}{(2\pi)^{3/2}} \int_0^\infty \frac{r^2 g_n(r)\, g_n(-r)}{(1+r^2)^{1+\varepsilon}} dr,$$

where the asymptotic behavior of the spherical Bessel functions (see [1, (10.1)]) implies that λ_n is finite.

In summary, we have proved that Y_n^m are the eigenfunctions of the operator $\gamma_0 B_\varepsilon$ with eigenvalues λ_n. We already proved (cf. Theorem 2.5) that $B_\varepsilon : H^{-1/2-\varepsilon}(\Gamma) \to H^{1/2+\varepsilon}(\Gamma)$ is continuous and (cf. Theorem 2.6) that the sesquilinear form associated with $\gamma_0 B_\varepsilon$ is $H^{-1/2-\varepsilon}(\Gamma)$-elliptic and, thus, $\lambda_n > 0$. Hence,

$$\frac{1}{C}\,\|Y_n^m\|^2_{H^{-1/2-\varepsilon}(\Gamma)} \le \underbrace{(\gamma_0 B_\varepsilon Y_n^m, Y_n^m)_{0,\Gamma}}_{=\lambda_n} \le C\,\|Y_n^m\|^2_{H^{-1/2-\varepsilon}(\Gamma)} \quad \forall n \in \mathbb{N},\ -n \le m \le n,$$

where $C > 0$ is independent of n, m. For $\Gamma = \mathbb{S}^2$, the $H^{-1/2-\varepsilon}(\Gamma)$-norm is equivalent to the Fourier norm (cf. Lemma 5.2) and the estimate

$$\tilde{c}\,(1+n)^{-1-2\varepsilon} \le \lambda_n \le \tilde{C}\,(1+n)^{-1-2\varepsilon} \qquad \forall n \in \mathbb{N}$$

directly follows. ∎

5.2. Evaluation of the constants of interest in terms of eigenvalues of the underlying integral operators. The eigenvalues of the operator R_k are given by

$$\lambda_{n,k}^{(R)} := \lambda_{n,k}^{(V)} + \mathrm{i}\,\eta \lambda_n^{(B)} \lambda_{n,k}^{(K)}. \tag{5.7}$$

Remark 5.5. Assumption 5.3 implies that the qualitative dependence of $\lambda_n^{(B)}$ on n can be studied by replacing $\lambda_n^{(B)}$ by $\tilde{\lambda}_n^{(B)} := (n+1)^{-1-2\varepsilon}$. For all numerical experiments, we have employed this simplification and indicated this by a superscript $\tilde{\lambda}$.

Since we are concerned in this section with the smooth surface of the unit ball we may choose the parameter μ in the definition of h_0 (cf. (4.12)) by

$$\mu := \min\left\{\frac{3}{2}, 1+2\varepsilon\right\}. \tag{5.8}$$

In view of Theorem 4.3, we see that the maximal mesh width h_0 which guarantees existence and uniqueness is given by (4.12) and depends on α_0, ε, C_μ, C_X, C_A. Our goal is to analyze the dependence of h_0 on k and h. Since α_0 and C_A (cf. (4.5), Theorem 4.1) are independent of k and h, we are left with the analysis of C_μ (cf. (4.11)), C_X (cf. (4.10)). Note that such estimates directly imply an estimate of the constant $2C_X/\alpha_0$ in the error estimate (4.13).

i. The continuity constant C_X can be estimated in terms of eigenvalues by

$$\begin{aligned} C_X = \|R_k\|_{H^{1/2}(\Gamma)\leftarrow H^{-1/2}(\Gamma)} &\le \sup_{n\in\mathbb{N}} \sqrt{1+n^2}\left|\lambda_{n,k}^{(R)}\right| \\ &\le \sup_{n\in\mathbb{N}} \sqrt{1+n^2}\left(\left|\lambda_{n,k}^{(V)}\right| + \eta\left|\lambda_n^{(B)}\lambda_{n,k}^{(K)}\right|\right). \end{aligned} \tag{5.9}$$

ii. The constant C_μ can be expressed in terms of the eigenvalues as follows. Let $\varphi = \sum_{n=0}^\infty \sum_{m=-n}^n \varphi_n^m Y_n^m$ be such that $\sum_{n=0}^\infty \sum_{m=-n}^n (1+n^2)^{-1/2}|\varphi_n^m|^2 = 1$; then $(R_k^\star)^{-1}\tilde{R}_k\varphi$ reads

$$(R_k^\star)^{-1}\tilde{R}_k\varphi = \sum_{n=0}^\infty \frac{1}{\overline{\lambda}_{n,k}^{(R)}}\left(\lambda_{n,k}^{(R)} - \frac{2}{2n+1} - \frac{\mathrm{i}\,\eta}{2}\lambda_n^{(B)}\right)\sum_{m=-n}^n \varphi_n^m Y_n^m$$

and C_μ can be expressed by

$$C_\mu = \sup_{n\in\mathbb{N}}\left\{\frac{\left(1+n^2\right)^{\frac{\mu+1}{2}}\left|\lambda_{n,k}^{(R)} - \frac{2}{2n+1} - \frac{\mathrm{i}\,\eta}{2}\lambda_n^{(B)}\right|}{|\lambda_{n,k}^{(R)}|\sqrt{1+n^2}}\right\}. \tag{5.10}$$

The rest of the section is structured as follows: In subsection 5.3 we will study the behavior (in k and n) of the eigenvalues in the asymptotic ranges of k and n; the choice of η in subsection 5.4 will be based on these asymptotics; in subsection 5.5 we will study the behavior of the eigenvalues in the nonasymptotic range of k and n via computer experiments. In subsection 5.6 we will derive bounds for the quantities C_X and h_0 which are [illegible] in the wave number k. Finally, in subsection 5.7 we will investigate the oscillation hypothesis made in Theorem 4.7.

5.3. Asymptotic analysis of the eigenvalues. The following lemma concerns the asymptotic behavior of the eigenvalues $\lambda_{n,k}^{(V)}$, $\lambda_n^{(B)}$, and $\lambda_{n,k}^{(K)}$. Let

$$\operatorname{trig}(k) := \begin{cases} \sin k & \text{if } n \text{ is even,} \\ \mathrm{i}\cos k & \text{if } n \text{ if odd,} \end{cases}$$

where we suppress the dependence on n in the notation of trig. Note that

$$\operatorname{trig}\left(k+\frac{\pi}{2}\right) = \operatorname{trig}'(k) = \begin{cases} \cos k & \text{if } n \text{ is even,} \\ -\mathrm{i}\sin k & \text{if } n \text{ if odd.} \end{cases}$$

LEMMA 5.6. [illegible] $\lambda_{n,k}^{(V)}$ $\lambda_{n,k}^{(K)}$ [illegible]

1. [illegible] n [illegible] $k \to \infty$ [illegible]

$$\lambda_{n,k}^{(V)} = 2\frac{e^{\mathrm{i}k}}{k}\operatorname{trig}(k) + O\left(\frac{1}{k^2}\right),$$

$$\lambda_{n,k}^{(K)} = e^{\mathrm{i}k}\operatorname{trig}'(k) + O\left(\frac{1}{k}\right).$$

2. [illegible] $k = n + 1/2$ [illegible]

$$\lambda_{n,k}^{(V)} = \pi\left(\frac{2^{1/3}}{3^{2/3}\Gamma\left(\frac{2}{3}\right)}\right)^2 k^{-2/3}\left(\sqrt{3}+\mathrm{i}\right)(1+o(1)),$$

$$\lambda_{n,k}^{(K)} = \frac{\mathrm{i}+\sqrt{3}}{2\sqrt{3}}(1+o(1)).$$

3. [illegible] k [illegible] $n \to \infty$ [illegible]

$$\lambda_{n,k}^{(V)} = \frac{2}{2n+1}\left(1+O\left(\frac{1}{n^2}\right)\right),$$

$$\lambda_{n,k}^{(K)} = \frac{1}{2} + O\left(\frac{1}{n}\right).$$

[illegible] Ad 1. The [illegible] Bessel functions j_n, h_n can be expressed via the Bessel functions of first and third kind J_n and H_n (cf. [1, (10.1.1)]):

$$j_n(z) = \sqrt{\frac{\pi}{2z}}J_{n+1/2}(z) \quad \text{and} \quad h_n(z) = \sqrt{\frac{\pi}{2z}}H_{n+1/2}^{(1)}(z). \tag{5.11}$$

We combine (5.11) with the asymptotic expansion (cf. [1, (9.2.1) and (9.2.3)]) to obtain

$$
\begin{aligned}
(5.12)\qquad h_n^{(1)}(k)\, j_n(k) &\overset{[1,\ (10.1.1)]}{=} \frac{\pi}{2k} J_{n+1/2}(k)\, H_{n+1/2}^{(1)}(k) \\
&\overset{[1,\ (9.2.1),\ (9.2.3)]}{\sim} \frac{1}{k^2} e^{\mathrm{i}\left(k-\frac{\pi}{2}(n+1)\right)} \cos\left(k-\frac{\pi}{2}(n+1)\right) + O\left(\frac{1}{k^3}\right) \\
&= \frac{e^{\mathrm{i}\,k}}{\mathrm{i}\,k^2}\,\mathrm{trig}(k) + O\left(\frac{1}{k^3}\right)
\end{aligned}
$$

and

$$
\begin{aligned}
h_n^{(1)}(k)\, j_n'(k) &\overset{[1,\ (10.1.20)]}{=} \sqrt{\frac{\pi}{2k}} H_{n+1/2}^{(1)}(k)\, \frac{\left(n j_{n-1}(k) - (n+1)\, j_{n+1}(k)\right)}{2n+1} \\
(5.13)\qquad &\overset{[1,\ (10.1.1)]}{=} \frac{\pi}{2k} H_{n+1/2}^{(1)}(k)\, \frac{\left(n J_{n-1/2}(k) - (n+1)\, J_{n+3/2}(k)\right)}{2n+1} \\
&\overset{[1,\ (9.2.1),\ (9.2.3)]}{\sim} \frac{e^{\mathrm{i}\,k}}{k^2} e^{-\mathrm{i}\,\frac{\pi}{2}(n+1)} \frac{n\cos\left(k-\frac{\pi n}{2}\right) - (n+1)\cos\left(k-\frac{\pi(n+2)}{2}\right)}{2n+1} \\
&\quad + O\left(\frac{1}{k^3}\right) \\
&= \frac{e^{\mathrm{i}\,k}}{\mathrm{i}\,k^2}\,\mathrm{trig}'(k) + O\left(\frac{1}{k^3}\right).
\end{aligned}
$$

Hence

$$
(5.14)\qquad \lambda_{n,k}^{(V)} = 2\,\mathrm{i}\,k h_n^{(1)}(k)\, j_n(k) = 2\frac{e^{\mathrm{i}\,k}}{k}\,\mathrm{trig}(k) + O\left(\frac{1}{k^2}\right),
$$

$$
(5.15)\qquad \lambda_{n,k}^{(K)} = \mathrm{i}\,k^2 h_n^{(1)}(k)\, j_n'(k) = e^{\mathrm{i}\,k}\,\mathrm{trig}'(k) + O\left(\frac{1}{k}\right).
$$

Ad 2. Let $k = n + 1/2$. Then,

$$
j_n\left(n+\frac{1}{2}\right) \overset{(5.11)}{=} \sqrt{\frac{\pi}{2k}} J_k(k) \quad \text{and} \quad h_n^{(1)}(k) = \sqrt{\frac{\pi}{2k}} H_k^{(1)}(k).
$$

We employ the asymptotics for $J_\nu(\nu)$ and $H_\nu(\nu)$ to obtain

$$
\begin{aligned}
\lambda_{n,k}^{(V)} &= 2\,\mathrm{i}\,k j_n\left(n+\frac{1}{2}\right) h_n^{(1)}\left(n+\frac{1}{2}\right) \\
&\overset{[1,\ (9.3.31),\ (9.3.32)]}{=} \pi\left(\frac{2^{1/3}}{3^{2/3}\Gamma\left(\frac{2}{3}\right)}\right)^2 k^{-2/3}\left(\sqrt{3}+\mathrm{i}\right)(1+o(1))
\end{aligned}
$$

and

$$
\begin{aligned}
\lambda_{n,k}^{(K)} &= \mathrm{i}\,k^2 h_n^{(1)}(k)\, j_n'(k) \overset{(5.13)}{=} \mathrm{i}\,k^2 \frac{\pi}{2k} H_k^{(1)}(k)\, \frac{\left(\left(k-\frac{1}{2}\right) J_{k-1}(k) - \left(k+\frac{1}{2}\right) J_{k+1}(k)\right)}{2k} \\
&\overset{[1,\ (9.1.27)(1)\text{-}(2)]}{=} \mathrm{i}\,k\frac{\pi}{2} H_k^{(1)}(k)\left(J_k'(k) - \frac{1}{2k} J_k(k)\right) \\
&\overset{[1,\ (9.3.31\text{-}34)]}{\sim} \frac{\mathrm{i}+\sqrt{3}}{2\sqrt{3}}(1+o(1)).
\end{aligned}
$$

Ad 3. Let k be fixed and $n \to \infty$.

The definition of the spherical Bessel and Hankel functions implies in that case

$$\lambda_{n,k}^{(V)} = \frac{2\,\mathrm{i}\,k \cdot k^n \left\{1 - \frac{k^2/2}{2n+3} + O\left(n^{-2}\right)\right\}}{1 \cdot 3 \cdot \cdots \cdot (2n+1)} \times \frac{1 \cdot 3 \cdot \cdots \cdot (2n-1)\left\{1 - \frac{k^2/2}{1-2n} + O\left(n^{-2}\right)\right\}}{\mathrm{i}\,k^{n+1}}$$
$$= \frac{2}{2n+1}\left(1 + O\left(n^{-2}\right)\right)$$

and

$$\lambda_{n,k}^{(K)} = \mathrm{i}\,k^2 h_n^{(1)}(k)\, j_n'(k) = \frac{n}{2n+1}\left(1 + O\left(\frac{1}{n}\right)\right) = \frac{1}{2} + O\left(\frac{1}{n}\right). \qquad \square$$

Lemma 5.6 leads to the following asymptotic behavior of the real and imaginary parts of the eigenvalues $\lambda_{n,k}^{(R)}$. We restrict ourselves to the case $k \geq k_0 > 0$. The "$\sim$"-notation indicates that we neglect the higher order terms in Lemma 5.6. The numbers $a_0, \ldots, a_4$ below are positive and may depend on k_0 but not on k.

- $n = 0$, $k \to \infty$:

  (5.16a)
  $$\operatorname{Re}\lambda_{0,k}^{(R)} \sim \sin k \cos k \left(\frac{2}{k} - a_0\eta\right) \quad \text{and} \quad \operatorname{Im}\lambda_{0,k}^{(R)} \sim 2\frac{\sin^2 k}{k} + a_0\eta\cos^2 k.$$

- $n + \frac{1}{2} = k$:

  (5.16b)
  $$\sqrt{1+n^2}\operatorname{Re}\lambda_{n,k}^{(R)} \sim a_1 k^{1/3} - \frac{a_2\eta}{k^{2\varepsilon}} \quad \text{and} \quad \sqrt{1+n^2}\operatorname{Im}\lambda_{n,k}^{(R)} \sim a_3 k^{1/3} + \frac{a_4\eta}{k^{2\varepsilon}}.$$

- k fixed and $n \to \infty$:

  (5.16c)
  $$\sqrt{1+n^2}\operatorname{Re}\lambda_{n,k}^{(R)} \sim 1 \quad \text{and} \quad \operatorname{Im}\lambda_{n,k}^{(R)} \to 0.$$

5.4. Choice of η. Our goal is to stabilize the single layer potential V_k such that the constant governing the convergence behavior, i.e., $2C_X/\alpha_0$, and the bound for the maximal step width h_0 are as close as possible to those of the pure single layer potential away from the forbidden frequencies. The asymptotic behavior of the eigenvalues in (5.16b) indicates that the continuity constant C_X in (5.9) cannot behave better than $O\left(k^{1/3}\right)$. This leads to the heuristics to choose the stabilization parameter η maximal under the side condition that C_X still is bounded by $Ck^{1/3}$. The asymptotics (5.16a)–(5.16c) show that the choice $\eta = k^{1/3}$ leads to the bound

$$\sqrt{1+n^2}\left|\lambda_{n,k}^{(R)}\right| \leq Ck^{1/3}$$

for the three asymptotic cases, while the choice $\eta = k^{1/3+\delta}$ for any $\delta > 0$ would lead to an increased constant $C_X = O\left(k^{1/3+\delta}\right)$; cf. (5.16a).

We have performed computer experiments to study the behavior of $\sqrt{1+n^2}|\lambda_{n,k}^{(R)}|$ in the intermediate ranges.

a. Figure 5.1 indicates that the case $n + \frac{1}{2} = k$ (Lemma 5.6(2)) is relevant for the upper bound of $\sqrt{1+n^2}|\lambda_{n,k}^{(V)}|$ and this choice leads to

(5.17)
$$\sqrt{1 + \left(k - \frac{1}{2}\right)^2}\left|\lambda_{k-1/2,k}^{(V)}\right| \overset{\text{Lemma 5.6(2)}}{\leq} C' k^{1/3}.$$

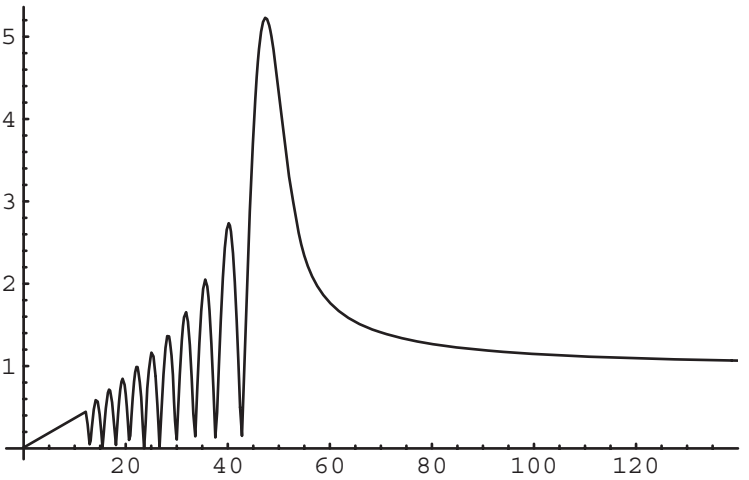


FIG. 5.1. *Plot of* $\sqrt{1+n^2}\left|\lambda_{n,k}^{(V)}\right|$ *for* $k = 50$ *in the range of* $1 \le n \le 300$. *The maximum is achieved at about* $n = k - 1/2$. *Pictures for different values of* k *show the same behavior.*

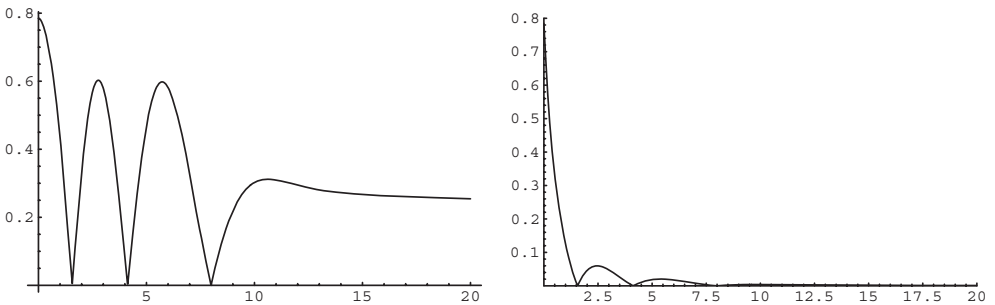


FIG. 5.2. *Plot of* $\sqrt{1+n^2}|\tilde{\lambda}_n^{(B)}\lambda_{n,k}^{(K)}|$ *for* $k = 10$ *and* $\varepsilon = \frac{1}{10}$ *(left picture),* $\varepsilon = 1$ *(right picture) in the range of* $0 \le n \le 20$. *The function decreases as* $n \to \infty$. *The maximum is achieved at* $n = 0$. *Pictures for different values of* k *show qualitatively the same behavior.*

b. We have plotted the upper bound of $|\sqrt{1+n^2}\tilde{\lambda}_n^{(B)}\lambda_{n,k}^{(K)}|$ for a small value $\varepsilon = 0.1$ and the maximal value $\varepsilon = 1$ to study its qualitative behavior. Figure 5.2 indicates that the case $n = 0$ is relevant for the upper bound and this choice leads to the (rough) estimate (cf. Lemma 5.2 (b.ii) for $n = 0$)

$$\left|\lambda_0^{(B)}\lambda_{0,k}^{(K)}\right| \le C\left|\cos k - \frac{\sin k}{k}\right| \le 2C. \tag{5.18}$$

Hence the estimates (5.17), (5.18) along with the asymptotics (5.16a)–(5.16c) suggest $\eta \le k^{1/3}$ in order not to destroy the upper bound for the continuity constant C_X.

DEFINITION 5.7. [illegible] η [illegible] (2.4) [illegible]

$$\eta := k^{1/3}.$$

5.5. Computer based analysis of the constant C_μ. Recall the choice of the shift parameter μ as in (5.8) and the formula (5.10) for the constant $C_\mu = \sup_{n\in\mathbb{N}} \frac{\beta_{n,k}}{\gamma_{n,k}}$,

where

$$\beta_{n,k} := \left(1+n^2\right)^{\frac{\mu+1}{2}} \left|\lambda_{n,k}^{(R)} - \frac{2}{2n+1} - \frac{\mathrm{i}\,\eta}{2}\lambda_n^{(B)}\right| \quad \text{and} \quad \gamma_{n,k} := \sqrt{1+n^2}\left|\lambda_{n,k}^{(R)}\right|.$$

First, we will estimate the behavior of $\beta_{n,k}$. The results of the asymptotic analysis (see Lemma 5.6 and (5.16)) together with (5.5) yield the following:

1. $n = 0$ and $k \to \infty$,

$$\left(1+n^2\right)^{\frac{\mu+1}{2}} \left|\lambda_{n,k}^{(R)} - \frac{2}{2n+1} - \frac{\mathrm{i}\,\eta}{2}\lambda_n^{(B)}\right| \le a_5 k^{1/3}; \tag{5.19a}$$

2. for $n + \frac{1}{2} = k$,

$$\left(1+n^2\right)^{\frac{\mu+1}{2}} \left|\lambda_{n,k}^{(R)} - \frac{2}{2n+1} - \frac{\mathrm{i}\,\eta}{2}\lambda_n^{(B)}\right| \le a_6 k^{\mu+1/3}; \tag{5.19b}$$

3. for k fixed and $n \to \infty$,

$$\left(1+n^2\right)^{\frac{\mu+1}{2}} \left|\lambda_{n,k}^{(R)} - \frac{2}{2n+1} - \frac{\mathrm{i}\,\eta}{2}\lambda_n^{(B)}\right| \le a_7 k^{1/3}. \tag{5.19c}$$

In Figure 5.3, the function $\left(1+n^2\right)^{\frac{\mu+1}{2}} |\lambda_{n,k}^{(R)} - \frac{2}{2n+1} - \frac{\mathrm{i}\,\eta}{2}\lambda_n^{(B)}|$ is depicted for $k = 20$ indicating that the bounds derived from asymptotic analysis are valid in neighborhoods of $n = 0$, $n = k - 1/2$ as well.

Next, we will investigate the behavior of $\gamma_{n,k}$. The asymptotic behavior of $\gamma_{n,k}$ as $n \to \infty$ is derived by using (5.16c)

$$\left|\gamma_{n,k}\right| \ge \sqrt{1+n^2}\left|\operatorname{Re}\lambda_{n,k}^{(R)}\right| \sim 1 \quad \text{for } k \text{ fixed and } n \to \infty.$$

By choosing $\eta = k^{1/3}$, the function $\operatorname{Re}\lambda_{n,k}^{(R)}$ is oscillating as a function of $n < k$ about zero and bounded from below properly away from zero for $n \ge k$ (cf. Figure 5.4). This supports the heuristics

$$\inf_{n\ge k} \gamma_{n,k} \ge \inf_{n\ge k} \left\{\sqrt{1+n^2}\operatorname{Re}\lambda_{n,k}^{(R)}\right\} \ge 1. \tag{5.20a}$$

The behavior of $\gamma_{n,k}$ in the range $0 \le n < k$ is more complicated.

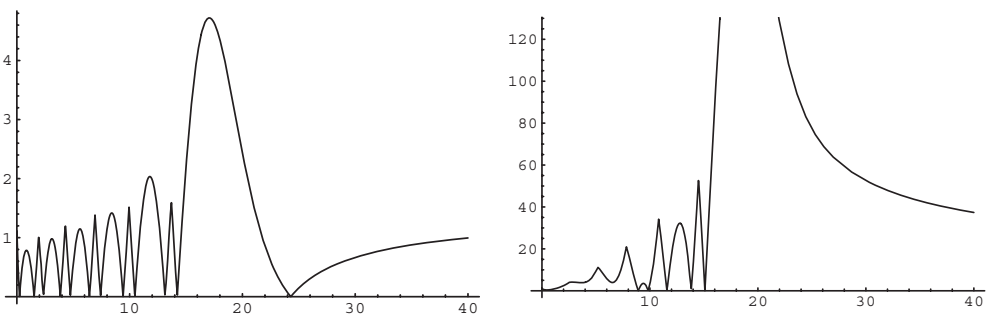


FIG. 5.3. *Plot of* $\left(1+n^2\right)^{\frac{s+1}{2}} |\lambda_{n,k}^{(R)} - \frac{2}{2n+1} - \frac{\mathrm{i}\,\eta}{2}\tilde{\lambda}_n^{(B)}|$ *for* $k = 20$ *and* $\varepsilon = \frac{1}{10}$ *(left),* $\varepsilon = 1$ *(right) in the range of* $n \in [0, 40]$.

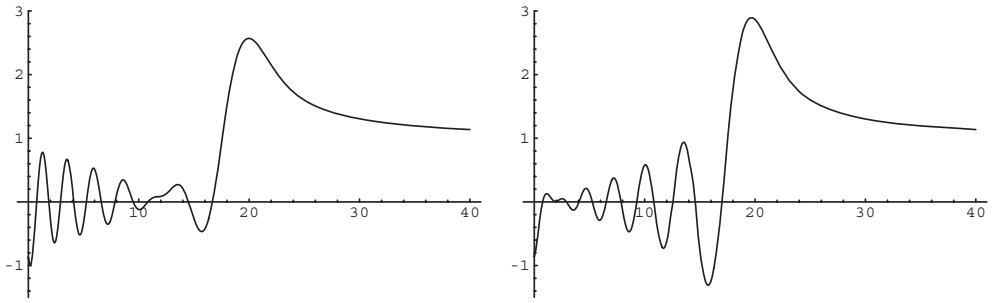


FIG. 5.4. *Function* $\sqrt{1+n^2}\,\mathrm{Re}\,\tilde{\lambda}_{n,k}^{(R)}$ *for fixed* $k = 20$ *and* $\varepsilon = \frac{1}{10}$ *(left picture),* $\varepsilon = 1$ *(right picture) in the range* $n \in [0, 40]$. *The function values are oscillating for* $n \in [0, k[$ *about zero and bounded from below away from zero in the range* $n \geq k$.

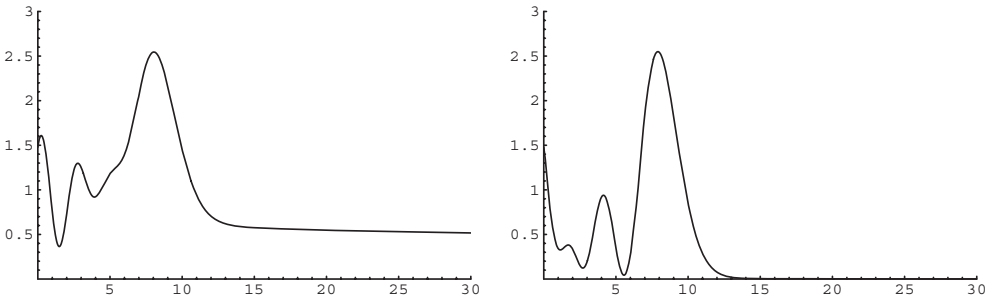


FIG. 5.5. *Plot of* $\sqrt{1+n^2}\,\mathrm{Im}\,\tilde{\lambda}_{n,k}^{(R)}$ *for* $k = 10$ *and* $\varepsilon = \frac{1}{10}$ *(left picture),* $\varepsilon = 1$ *(right picture) in the range* $0 \leq n \leq 30$. *The function values are all positive and decreasing as* $n \to \infty$.

By choosing $\eta = k^{1/3}$, the function $\mathrm{Im}\,\lambda_{n,k}^{(R)}$ is positive for all $n \geq 0$ and $k \geq 0$ (cf. Figure 5.5).

Since the quantities $\mathrm{Re}\,\lambda_{n,k}^{(R)}$ are oscillating in the range $0 \leq n \leq k$, we employ the imaginary part to bound $\gamma_{n,k}$ from below

$$\gamma_{n,k} \geq \sqrt{1+n^2}\,\mathrm{Im}\,\lambda_{n,k}^{(R)} = f_1(n,k) + f_2(n,k,\varepsilon)\,, \tag{5.21}$$

where

$$f_1(n,k) := \sqrt{1+n^2}\,\mathrm{Im}\,\lambda_{n,k}^{V} \quad \text{and} \quad f_2(n,k,\varepsilon) := \sqrt{1+n^2}\,\mathrm{Im}\left(\mathrm{i}\,\eta\lambda_n^{(B)}\lambda_{n,k}^{(K)}\right).$$

We have plotted both summands in the right-hand side of (5.21) for $\eta = k^{1/3}$ in the range $0 \leq n \leq k$ separately (cf. Figure 5.6).

In the range $0 \leq n \leq k$, the function f_1 is large when f_2 is zero or close to zero while, vice versa, f_2 is large when f_1 is zero or close to zero. The maxima of f_1 are monotonously increasing while the maxima of f_2 are monotonously decreasing. These observations lead to the following heuristics.

- If n is small, then

$$\gamma_{n,k} \geq \sqrt{1+n^2}\,\mathrm{Im}\,\lambda_{n,k}^{(R)} \geq a_8 f_1(n_0,k) \overset{(5.16a)}{\geq} a_9 k^{-1}, \tag{5.20b}$$

  where n_0 denotes the first local maximum of f_1 and a_8, a_9 depend only on k_0.

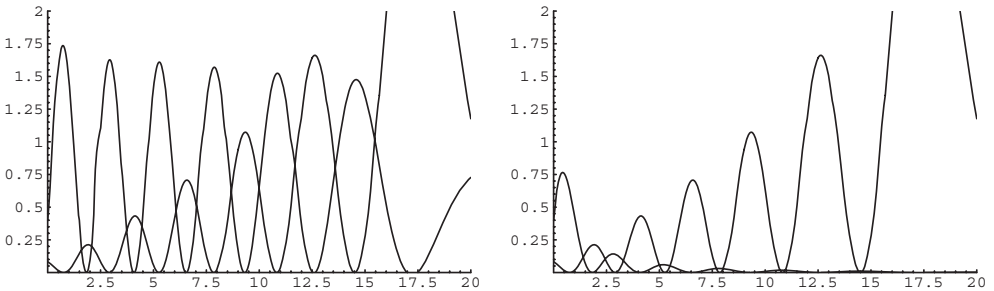


FIG. 5.6. *Functions* $\sqrt{1+n^2}\,\mathrm{Im}\,\lambda_{n,k}^V$ *and* $\sqrt{1+n^2}\,\mathrm{Im}(\mathrm{i}\,\eta\tilde{\lambda}_n^{(B)}\lambda_{n,k}^{(K)})$ *for* $k=20$ *and* $\varepsilon=\frac{1}{10}$ *(left picture),* $\varepsilon=1$ *(right picture) as a function of* n *in the range* $0\le n\le 20$. *The zeros and extrema are properly separated.*

- If n is close to k but smaller than k, then

  (5.20c)
  $$\gamma_{n,k}\ge\sqrt{1+n^2}\,\mathrm{Im}\,\lambda_{n,k}^{(R)}\ge a_{10}f_2\left(k-\frac{1}{2},k,\varepsilon\right)\overset{(5.5),\ \text{Lemma 5.6(2)},}{\ge}a_{11}k^{1/3-2\varepsilon},$$

  where a_{10}, a_{11} depend only on k_0.

We combine the estimates (5.19) and (5.20) and obtain the following heuristics.

- If n is small, then
  $$\beta_{n,k}/\gamma_{n,k}\le a_{12}k^{4/3}.$$
- If n is close to k but smaller than k, then
  $$\beta_{n,k}/\gamma_{n,k}\le a_{13}k^{\mu+2\varepsilon}.$$
- If k is fixed and n becomes large, then
  $$\beta_{n,k}/\gamma_{n,k}\le a_{14}k^{1/3}.$$

This leads to the heuristics

$$C_\mu=\sup_{n\ge0}\frac{\beta_{n,k}}{\gamma_{n,k}}\le a_{15}k^{\max\{4/3,\mu+2\varepsilon\}}. \tag{5.22}$$

To verify the qualitative dependence of C_μ on k numerically in the whole range of n we have evaluated the quantity $k^{-\max\{4/3,\mu+2\varepsilon\}}\beta_{n,k}/\gamma_{n,k}$ as a function of n for different values of k and ε by the software [illegible]. In Table 5.1, the results are depicted which strongly support the heuristics that the asymptotic behavior as in (5.22) is valid in the whole range of n.

5.6. Estimates on the constants of interest. We combine the general error estimate (cf. Theorem 4.3) with the estimate for the continuity constant C_X (cf. subsection 5.4) to obtain an estimate of the Galerkin error which is explicit with respect to the wave number k.

THEOREM 5.8. [illegible] h [illegible] $h<h_0$ h_0 [illegible] (4.12). [illegible] $\hat{C}$ $k_0>0$ [illegible] $k\ge k_0$

$$\frac{\|\varphi-\varphi_h\|_{H^{-1/2}(\Gamma)}}{\|\varphi\|_{H^{-1/2}(\Gamma)}}\le\hat{C}k^{1/3}\,(hk)^{3/2}. \tag{5.23}$$

TABLE 5.1
Numerical verification of the heuristics (5.22) *for the constant* C_μ. *The table lists the numerical determined value of* $k^{-\max\{4/3,\mu+2\varepsilon\}} \sup_{n\geq 0} \beta_{n,k}/\gamma_{n,k}$ *which we expect to be bounded by a constant independent of* k.

ε	$k=10$	$k=20$	$k=40$	$k=80$	$k=160$
$\frac{1}{20}$	0.307	0.301	0.284	0.258	0.240
$\frac{1}{5}$	0.251	0.278	0.268	0.238	0.201
1	0.0917	0.142	0.164	0.162	0.148

Finally, we will determine the dependence of the maximal step size h_0 on the wave number k such that existence and uniqueness of the Galerkin solution is guaranteed. The combination of (4.12) with the results of subsection 5.4 and (5.22) leads to the condition

$$h_0^\mu k^{1/3} k^{\max\{4/3,\mu+2\varepsilon\}} \leq C, \tag{5.24}$$

where the constant C is independent of k and ε. Recall that $\mu = \min\{3/2, 1+2\varepsilon\}$. We distinguish the following cases.

1. $0 \leq \varepsilon \leq \frac{1}{12}$. Condition (5.24) takes the form
$$h^{1+2\varepsilon} k^{5/3} \leq C.$$
2. $\frac{1}{12} \leq \varepsilon \leq \frac{1}{4}$. Condition (5.24) takes the form
$$h^{1+2\varepsilon} k^{4/3+4\varepsilon} \leq C.$$
3. $\frac{1}{4} \leq \varepsilon \leq 1$. Condition (5.24) takes the form
$$h^{3/2} k^{11/6+2\varepsilon} \leq C.$$

By inspecting these cases we derive that the choice $\varepsilon = 1/12$ is optimal and the condition

$$(hk)\, k^{\frac{3}{7}} \leq C$$

ensures existence and uniqueness. In Table 5.2, the Galerkin error estimate and the stability condition are listed for different choices of $\gamma_0 B$.

We finish this section by comparing these results with the quantitative analysis of the Brakhage–Werner stabilization which we briefly recall. For the solution of the Helmholtz problem, the ansatz

$$u_+ = D_k \psi - i\eta S_k \psi$$

TABLE 5.2
Stability condition and estimate of the relative Galerkin error for different choices of $\gamma_0 B$.

$\gamma_0 B$	ε	Stability condition	Rel. Galerkin error is smaller than
$(V_0)^2$	$\frac{1}{2}$	$hk^{1+8/9} \leq C$	$\hat{C} k^{1/3} (hk)^{3/2}$
B_1	1	$hk^{1+14/19} \leq C$	$\hat{C} k^{1/3} (hk)^{3/2}$
$B_{1/12}$	$\frac{1}{12}$	$hk^{1+3/7} \leq C$	$\hat{C} k^{1/3} (hk)^{3/2}$

is employed and the density ψ is determined by solving the following boundary integral equation:

Find $\psi \in L^2(\Gamma)$ such that

$$\left(\left(\frac{1}{2} I + K_k - i\eta V_k \right) \psi, \sigma \right)_{0,\Gamma} = (g, \sigma)_{0,\Gamma} \qquad \forall \sigma \in L^2(\Gamma).$$

Let ψ_h denote the corresponding Galerkin solution where $L^2(\Gamma)$ is replaced by the piecewise constant boundary element space S as in (3.2). By choosing $\eta \sim k/4$, it was proved in [17]—for Γ being the surface of the unit sphere—that the relative error can be estimated by

$$\frac{\|\psi - \psi_h\|_{L^2(\Gamma)}}{\|\psi\|_{L^2(\Gamma)}} \leq \hat{C} k^{1/3} (hk).$$

By comparing this result with Table 5.2 it is obvious that the pollution factor which amplifies the error of the best approximation is $k^{1/3}$ for both cases, the Brakhage–Werner approach and the stabilized acoustic single layer potential. On the other hand, the convergence rate $(hk)^{3/2}$ compared to (kh) is higher for the stabilized single layer ansatz. In addition, we emphasize that the stability behavior of the acoustic single layer potential can be proved for general Lipschitz polyhedra while the proof of the well-posedness of the Brakhage–Werner approach is restricted to smooth surfaces. For the special case of a sphere, the stability condition $hk \lesssim 1$ for the Brakhage–Werner formulation is better compared to the stability condition $hk^{1+3/7} \lesssim 1$ for the stabilized single layer potential.

5.7. Oscillation condition. In this subsection, we will investigate the oscillation condition (4.15) in the case of the unit sphere.

Any right-hand side $g \in H^{1/2}(\Gamma)$ in (2.5) has a Fourier representation

$$g = \sum_{n=0}^{\infty} \sum_{m=-n}^{n} g_n^m Y_n^m.$$

Due to the orthogonality of the spherical harmonics, the solution of (2.5) has the representation

$$\varphi = \sum_{n=0}^{\infty} \frac{1}{\lambda_{n,k}^{(R)}} \sum_{m=-n}^{n} g_n^m Y_n^m.$$

The ratio of the $H^1(\mathbb{S}^2)$- and the $H^{-1/2}(\mathbb{S}^2)$-norms takes the form

$$Q(g) := \frac{\displaystyle\sum_{n=0}^{\infty} \frac{1+n^2}{\left|\lambda_{n,k}^{(R)}\right|} |g_n|^2}{\displaystyle\sum_{n=0}^{\infty} \frac{|g_n|^2}{\sqrt{1+n^2}\left|\lambda_{n,k}^{(R)}\right|}} = \frac{\|\varphi\|_{H^1(\Gamma)}^2}{\|\varphi\|_{H^{-1/2}(\Gamma)}^2} \quad \text{with} \quad |g_n|^2 := \sum_{n=-m}^{m} |g_n^m|^2.$$

By the substitution $w_n := \frac{g_n}{(1+n^2)^{1/4} |\lambda_{n,k}^{(R)}|^{1/2}}$ we obtain

$$Q(g) = \left(\sum_{n=0}^{\infty} \left(1+n^2\right)^{3/2} |w_n|^2 \right) / \|w_n\|_0^2.$$

It is not our goal to derive sharp decay conditions for the Fourier coefficients of the right-hand side $g \in H^{1/2}(\Gamma)$ so that $Q(g)$ is bounded by $C\left(1+k^2\right)^{3/2}$ and the oscillation property (4.15) is valid with properly chosen $\rho = O(1)$. Instead we will discuss the characteristic case that all oscillations in g vanish starting from $n \geq k$; i.e., all oscillations in the solution stem from the boundary integral equation itself and not from the right-hand side.

PROPOSITION 5.9. [illegible] $\Gamma = \mathbb{S}^2$ [illegible] g_n^m [illegible] $g \in H^{1/2}(\Gamma)$ [illegible] (3.1) [illegible]

$$g_n^m = 0 \qquad \forall n > k \quad -n \leq m \leq n$$

[illegible]

$$Q(g) \leq \left(1+k^2\right)^{3/2}$$

[illegible] (4.15) [illegible] $\rho = O(1)$

6. Computational complexity. In this section, we will remark on the computational complexity related to the different choices of the operator B.

The computation of the Galerkin system matrix $\mathbf{R}_k$ for (3.3) requires the evaluation of the integrals $(R_k b_\tau, b_t)_{\tau,t\in\mathcal{T}}$. Besides the additive term $K_k\gamma_0 B$ in (2.4), the discretization of the arising integral operators is standard. The numerical treatment of the composition $K_k\gamma_0 B$ is discussed in the following remark.

[illegible] 6.1. In the context of the Galerkin boundary element method, the exact computation of the system matrix entries is not possible in general and, e.g., numerical quadrature and panel clustering have to be employed. This leads to a perturbed Galerkin method and the accuracy requirements are determined via Strang's lemma [18]. In this light, we recommend replacing the matrix entries $\mathbf{W}_{\tau,t} := (K_k\gamma_0 B b_\tau, b_t)_0$ by $\tilde{\mathbf{W}}_{\tau,t} := (K_k P\gamma_0 B b_\tau, b_t)_0$, where P denotes the L^2-orthogonal projection onto $\mathcal{S}$. The advantage is that $\tilde{\mathbf{W}}$ has the representation

$$\tilde{\mathbf{W}} = \mathbf{K}_k\mathbf{M}^{-1}\mathbf{B},$$

where $\mathbf{K}_k := \left((K_k b_\tau, b_t)_0\right)_{\tau,t\in\mathcal{T}}$, respectively, $\mathbf{B} := \left((\gamma_0 B b_\tau, b_t)_0\right)_{\tau,t\in\mathcal{T}}$, are the standard Galerkin system matrices of K_k, respectively, $\gamma_0 B$, or approximations to it. The mass matrix $\mathbf{M} = \left((b_\tau, b_t)_0\right)_{\tau,t\in\mathcal{T}}$ is diagonal and its inversion is trivial.

In other words, the Galerkin system matrix $\mathbf{R}_k$ for the operator R_k can be approximately computed from the system matrices $\mathbf{V}_k$, $\mathbf{K}_k$, $\mathbf{B}$ (or approximations thereof) via

$$\widehat{\mathbf{R}}_k := \mathbf{V}_k + \mathrm{i}\,\eta\left(\frac{1}{2}\mathbf{B} + \mathbf{K}_k\mathbf{M}^{-1}\mathbf{B}\right).$$

Some comments concerning the complexity are listed below.

- Since for large problems the linear system should be solved iteratively, the matrix entries of $\widehat{\mathbf{R}}_k$ need not be computed explicitly; only a procedure for a matrix-vector multiplication has to be provided. Hence, the matrix-matrix multiplication $\mathbf{K}_k\mathbf{M}^{-1}\mathbf{B}$ does not have to be carried out, but $\mathbf{K}_k\mathbf{M}^{-1}\mathbf{B}\mathbf{v}$ can be realized by a matrix-vector multiplication with $\mathbf{K}_k$, one diagonal scaling, and the evaluation of $\mathbf{B}\mathbf{v}$. Table 6.1 lists the different strategies for this evaluation.

TABLE 6.1
Evaluation of $\mathbf{Bv}$.

$\gamma_0 B$	Storage	Matrix-vector multiplication
$(V_0)^2$	$\mathbf{V}_0 = \big((V_0 b_\tau, b_t)_0\big)_{\tau,t\in\mathcal{T}}$	$\mathbf{Bv} = \mathbf{V}_0\mathbf{M}^{-1}\mathbf{V}_0\mathbf{v}$
B_1	$\mathbf{B}_1 := \big((B_1 b_\tau, b_t)_0\big)_{\tau,t\in\mathcal{T}}$	$\mathbf{Bv} = \mathbf{B}_1\mathbf{v}$
$B_{1/12}$	$\mathbf{B}_{1/12} := \Big(\big(B_{1/12} b_\tau, b_t\big)_0\Big)_{\tau,t\in\mathcal{T}}$	$\mathbf{Bv} = \mathbf{B}_{1/12}\mathbf{v}$

- Since $\mathbf{B}$ is independent of the possibly high wave number k, the accuracy requirements for numerical quadrature, panel-clustering, and iterative solution are reduced compared to $\mathbf{V}_k$, $\mathbf{K}_k$. Hence, we expect that the extra cost for the stabilization is moderate.
- The choices $\mathbf{V}_0$, $\mathbf{B}_1$, $\mathbf{B}_{1/12}$ correspond to the discretization of a boundary integral operator (such as $\mathbf{V}_k$ and $\mathbf{K}_k$). Hence, the numerical implementation does not require any new data structures. By using the blackbox quadrature methods (cf. [16]) and/or the panel-clustering based on interpolation (cf. [5], [25]) the discretization of $\gamma_0 B$ requires only a subroutine for the evaluation of the integral kernel in (2.13).

Acknowledgments. Thanks are due to Prof. Herbert Amann for fruitful discussions. Some of the results were achieved during the authors' stay at the Mathematische Forschungsinstitut Oberwolfach. This support is greatly acknowledged.

REFERENCES

[1] M. ABRAMOWITZ AND I. A. STEGUN, EDS., *Handbook of Mathematical Functions, with Formulas, Graphs, and Mathematical Tables*, National Bureau of Standards Applied Mathematics Series 55, National Bureau of Standards, Washington, D.C., 1972.
[2] S. AMINI, *On the choice of the coupling parameter in boundary integral formulations of the exterior acoustic problem*, Appl. Anal., 35 (1990), pp. 75–92.
[3] I. BABUŠKA AND S. SAUTER, *Is the pollution effect of the FEM avoidable for the Helmholtz equation considering high wave numbers?* SIAM Rev., 42 (2000), pp. 451–484; reprint of SIAM J. Numer. Anal., 34 (1997), pp. 2392–2423.
[4] H. BRAKHAGE AND P. WERNER, *Über das Dirichletsche Außenraumproblem für die Helmholtzsche Schwingungsgleichung*, Arch. Math., 16 (1965), pp. 325–329.
[5] A. BRANDT AND A. A. LUBRECHT, *Multilevel matrix multiplication and fast solutions of integral equations*, J. Comput. Phys., 90 (1990), pp. 348–370.
[6] S. C. BRENNER AND L. R. SCOTT, *The Mathematical Theory of Finite Element Methods*, Springer-Verlag, New York, 2002.
[7] A. BUFFA AND P. CIARLET, JR., *On traces for functional spaces related to Maxwell's equations. Part* I: *An integration by parts formula in Lipschitz polyhedra*, Math. Methods Appl. Sci., 21 (2001), pp. 9–30.
[8] A. BUFFA AND R. HIPTMAIR, *A coercive combined field integral equation for electromagnetic scattering*, SIAM J. Numer. Anal., 42 (2004), pp. 621–640.
[9] A. BUFFA AND R. HIPTMAIR, *Coercive combined field integral equations*, Numer. Math., 100 (2005), pp. 1–19.
[10] A. BURTON AND G. MILLER, *The application of integral methods for the numerical solution of boundary value problems*, Proc. Roy. Soc. London Sect. A, 323 (1971), pp. 201–210.
[11] D. COLTON AND R. KRESS, *Inverse Acoustic and Electromagnetic Scattering Theory*, Appl. Math. Sci. 93, Springer-Verlag, Berlin, 1992.
[12] M. COSTABEL, *Boundary integral operators on Lipschitz domains: Elementary results*, SIAM J. Math. Anal., 19 (1988), pp. 613–626.
[13] M. DAUGE, *Elliptic Boundary Value Problems on Corner Domains*, Lecture Notes in Math. 1341, Springer-Verlag, Berlin, 1988.

[14] L. Demkowicz, *Asymptotic convergence in finite and boundary element methods:* II. *The LBB constant for rigid and elastic scattering problems*, Comput. Math. Appl., 28 (1994), pp. 93–109.
[15] Y. V. Egorov, A. I. Komech, and M. A. Shubin, *Elements of the Modern Theory of Partial Differential Equations*, Springer-Verlag, New York, 1999.
[16] S. Erichsen and S. A. Sauter, *Efficient automatic quadrature in* 3*-d Galerkin BEM*, Comp. Methods Appl. Mech. Engrg., 157 (1998), pp. 215–224.
[17] K. Giebermann, *Schnelle Summationsverfahren zur numerischen Lösung von Integralgleichungen für Streuprobleme im* $\mathbb{R}^3$, Ph.D. thesis, Universität Karlsruhe, Karlsruhe, Germany, 1997.
[18] I. G. Graham, W. Hackbusch, and S. A. Sauter, *Discrete boundary element methods on general meshes in* 3*D*, Numer. Math., 86 (2000), pp. 103–137.
[19] W. Hackbusch, *Elliptic Differential Equations*, Springer-Verlag, New York, 1992.
[20] F. Ihlenburg, *Finite Element Analysis of Acousting Scattering*, Springer-Verlag, New York, 1998.
[21] D. Jerison and C. E. Kenig, *The inhomogeneous Dirichlet problem in Lipschitz domains*, J. Funct. Anal., 130 (1995), pp. 161–219.
[22] R. Kress, *Minimizing the condition number of boundary integral operators in acoustics and electromagnetic scattering*, Quart. J. Mech. Appl. Math., 38 (1985), pp. 323–341.
[23] R. Kress, *Linear Integral Equations*, Springer-Verlag, Berlin, 1989.
[24] R. Kress and W. T. Spassov, *On the condition number of boundary integral operators for the exterior Dirichlet problem for the Helmholtz equation*, Numer. Math., 42 (1983), pp. 77–95.
[25] C. Lage, G. Schmidlin, and C. Schwab, *Rapid solution of first kind boundary integral equations in* $\mathbb{R}^3$, Engineering Analysis with Boundary Elements, 27 (2003), pp. 469–490.
[26] R. Leis, *Initial Boundary Value Problems in Mathematical Physics*, Stuttgart, John Wiley & Sons, Chichester, UK, 1986.
[27] J.-L. Lions and E. Magenes, *Non-Homogeneous Boundary Value Problems and Applications*, Springer-Verlag, New York, 1970.
[28] S. Mikhlin and S. Prössdorf, *Singular Integral Operators*, Springer-Verlag, Heidelberg, 1986.
[29] J. C. Nédélec, *Acoustic and Electromagnetic Equations*, Springer-Verlag, New York, 2001.
[30] N. Ortner and P. Wagner, *A survey on explicit representation formulae for fundamental solutions of linear partial differential operators*, Acta Appl. Math., 47 (1997), pp. 101–124.
[31] O. Panich, *On the question of the solvability of the exterior boundary-value problems for the wave equation and Maxwell's equation*, Russian Math. Surveys, 20 (1965), pp. 221–226.
[32] S. Sauter and C. Schwab, *Randelementmethoden*, Teubner, Leipzig, 2004.
[33] G. Schmidt and B. N. Khoromskij, *Boundary integral equations for the biharmonic Dirichlet problem on nonsmooth domains*, J. Integral Equations Appl., 11 (1999), pp. 217–253.
[34] K. Yosida, *Functional analysis*, Classics in Mathematics, Springer-Verlag, Berlin, 1995; reprint of the 6th (1980) edition.

SIAM J. SCI. COMPUT.
Vol. 28, No. 6, pp. 2001–2022
© 2006 Society for Industrial and Applied Mathematics

A FULLY MASS AND VOLUME CONSERVING IMPLEMENTATION OF A CHARACTERISTIC METHOD FOR TRANSPORT PROBLEMS*

TODD ARBOGAST† AND CHIEH-SEN HUANG‡

Abstract. The characteristics-mixed method considers the transport not of a single point or fluid particle, but rather the mass in an entire region of fluid. This mass is transported along the characteristic curves of the hyperbolic part of the transport equation, and the scheme thereby produces very little numerical dispersion, conserves mass locally, and can use long time steps. However, since the shape of a characteristic trace-back region must be approximated in numerical implementation, its volume may be incorrect, resulting in inaccurate concentration densities and, further, inaccurate reaction dynamics. We present a simple modification to the characteristics-mixed method that conserves both mass and volume of the transported fluid regions. Our algorithm also handles boundary conditions through a space-time change of variables in the trace-back routines, which allows the boundary to be treated as if it were interior to the domain. Nearly point sources, such as wells, present special difficulties, since characteristic trace-back curves converge in their vicinity. We also present techniques that allow one to conservatively implement wells. The techniques are illustrated in five numerical examples.

Key words. advection-diffusion, characteristics-mixed method, ELLAM, mixed finite element, local conservation, porous media

AMS subject classifications. 65M25, 76M25, 76S05

DOI. 10.1137/040621077

1. Introduction.

We consider in this paper the problem of tracer transport in a flow field, as might arise in a porous medium or a shallow water or atmospheric system. We concentrate on the problem as it arises in porous media simulation, though many of the ideas carry over to other contexts.

Let $\Omega \subset \mathbb{R}^2$ be our bounded domain, and consider an incompressible bulk fluid of velocity $\mathbf{u}(\mathbf{x}, t)$ satisfying the incompressibility condition

$$\nabla \cdot \mathbf{u} = q, \quad \Omega \times J, \tag{1.1}$$

where $q(\mathbf{x}, t)$ is a given external source or sink function, assumed smooth enough for our purposes, and $J = (0, \infty)$ is the time interval. Our interest lies in the transport of some dilute tracer or other solute species of concentration $c(\mathbf{x}, t)$ within the bulk fluid. We assume that it does not change the overall velocity $\mathbf{u}$. The concentration will generally satisfy an advection-diffusion equation of the form

$$(R\phi c)_t + \nabla \cdot (c\mathbf{u} - D\nabla c) = c_I q_+ + c q_- \equiv q_c(c), \quad \Omega \times J, \tag{1.2}$$

where $R(\mathbf{x})$ is the retardation factor, $\phi(\mathbf{x})$ is the storage factor of the medium called porosity, subscript t is time partial differentiation, $D(\mathbf{x}, t)$ is the diffusion/dispersion

*Received by the editors December 17, 2004; accepted for publication (in revised form) May 15, 2006; published electronically December 5, 2006.
http://www.siam.org/journals/sisc/28-6/62107.html

†Department of Mathematics, The University of Texas at Austin, 1 University Station C1200, Austin, TX 78712, and Institute for Computational Engineering and Sciences, The University of Texas at Austin, 1 University Station C0200, Austin, TX 78712 (arbogast@ices.utexas.edu). This author was supported in part by U.S. National Science Foundation grant DMS-0417431.

‡Corresponding author. Department of Applied Mathematics and National Center for Theoretical Sciences, National Sun Yat-sen University, Kaohsiung 804, Taiwan, R.O.C. (huangcs@math.nsysu.edu.tw).

coefficient (that may also depend on $\mathbf{u}$), $q_+(\mathbf{x},t) \geq 0$ is q when $q > 0$ and 0 otherwise, $q_-(\mathbf{x},t) = q - q_+ \leq 0$, and $c_I(\mathbf{x},t)$ is the given concentration of injected fluid.

To apply characteristic methods to (1.2), one generally uses an operator splitting technique to isolate the hyperbolic and parabolic parts of the equation. That is, over a time step, one approximates the hyperbolic part of the operator,

$$(R\phi c)_t + \nabla \cdot (c\mathbf{u}) = q_c(c), \quad \Omega \times J, \tag{1.3}$$

and the parabolic part,

$$(R\phi c)_t - \nabla \cdot (D\nabla c) = 0, \quad \Omega \times J, \tag{1.4}$$

in some order. We discuss neither the operator splitting nor the approximation of the parabolic part in this paper.

The hyperbolic part of (1.3) is the most delicate to approximate well. It models pure transport of the fluid particles, while the rest of the operator, $-\nabla \cdot (D\nabla c)$, is diffusive. In the absence of shocks and diffusion, fluid particles simply travel along the characteristics of (1.3) (see (2.2) below). The fluid must obey two principles: [illegible] and incompressibility, or, loosely speaking, [illegible] (i.e., mass conservation of the incompressible bulk fluid). The first principle is well known; we make the second precise in the next section, but basically, for an incompressible fluid, the volume occupied by a given amount of mass is fixed.

Numerical methods based on fixed grids, such as Godunov's method [15], are locally mass conservative by design. They are also automatically volume conserving, since the volumes of the fixed grid elements do not change in time.

Moving mesh and characteristic methods, however, do not automatically conserve volume. Characteristic methods became viable when Douglas and Russell described their [illegible] in 1982 [12]. In their method, one approximates the characteristic derivative by a finite difference in the characteristic direction; that is,

$$c_\tau(\mathbf{x},t) \equiv c_t(\mathbf{x},t) + \frac{\mathbf{u}(\mathbf{x},t)}{R\phi} \cdot \nabla c(\mathbf{x},t) \approx \frac{c(\mathbf{x},t+\Delta t) - c(\hat{\mathbf{x}},t)}{\Delta t} ,$$

where $\mathbf{x}$ is a grid point, $\Delta t > 0$, and $\hat{\mathbf{x}}$ is defined by tracing the point $(\mathbf{x}, t+\Delta t)$ backwards in time along the characteristics to $(\hat{\mathbf{x}}, t)$ (see section 2). This results in the approximation

$$R\phi \frac{c(\mathbf{x},t+\Delta t) - c(\hat{\mathbf{x}},t)}{\Delta t} = (c_I - c)q_+$$

of (1.3). Because the method is based on points, it violates [illegible] local mass and volume constraints. A subsequent modification produced a global mass balance [10], but not a local mass balance.

Various ELLAM [6] schemes have been developed based on the local mass constraint, including the characteristics-mixed method [2, 3] and its two-phase variant [11]. The basic idea is to trace back along the characteristics each entire grid element E to $\hat{E}$. In this way, all mass can be accounted for locally; that is, all the mass in $\hat{E}$ is numerically transported forward into E. In the absence of sources, sinks, and external boundaries, the volumes of E and $\hat{E}$ agree. However, to trace $\hat{E}$ back in time requires tracing each boundary point back, which can only be done in one space dimension (unless perhaps the velocity is particularly simple). So, in practice, one

must approximate $\hat{E}$ by some simpler shape $\tilde{E}$ by, say, tracing back only the vertices of the element. Almost assuredly the volumes of $\tilde{E}$ and $\hat{E}$ will disagree, violating the volume conservation principle.

Although mass is conserved locally, incorrect local volumes lead to incorrect concentrations, which measure mass per volume. That is, the density is incorrectly approximated and can lead to overshoot or undershoot and seriously degrade the quality of the solution over time, especially when reaction dynamics, based on densities, are also considered.

We present in this paper a simple and relatively computationally efficient method for adjusting the trace-back regions $\hat{E}$ to $\tilde{E}$. The difficulty, of course, is that the set of $\tilde{E}$ must tessellate the region (i.e., they must have no overlaps or gaps). For simplicity of exposition, we consider in this paper only a two-dimensional (2-D) domain and a rectangular grid of elements. In principle, our ideas should generalize to other grids and even to three dimensions, but we expect the difficulty of implementation to be considerable, and so we have no numerical experience to report for these cases at this time.

A more computationally expensive method was proposed by Chilakapati [7, 8]. He proposed a procedure that modifies the way in which the Darcy velocity itself is computed and then applies this to the advection equation.

An outline of this paper follows. We define the characteristics in the next section, and point out that our techniques cannot handle multiple characteristics as might arise in multiphase flow. In section 3, we then carefully derive the local mass and volume constraints. Since sources and sinks in porous media simulation arise from wells, which are very small, the source terms must be approximated carefully. Similar to sources are inflow boundaries, where $\mathbf{u}\cdot\nu < 0$ and points track out of the domain as time reverses. When the flow is not too strong, we have simple techniques for handling injection wells and inflow boundaries, as described in sections 4 and 5. Our trace-back point volume adjustment algorithm is described in section 6. For stronger flows, it is more effective to use a trace-forwarding technique to treat injection wells (and perhaps inflow boundaries), as presented in section 7. The final section is devoted to five numerical examples illustrating the need for volume adjustment and showing the efficacy of the proposed scheme.

2. Characteristic trace-back. In the absence of diffusion and dispersion (i.e., $D = 0$), fluid particles governed by (1.3) travel along paths called characteristics. In multiphase flow problems, there are multiple characteristics, as each phase flows at its own speed. We allow here only the relatively simple situation of linear adsorption, with a single fluid phase and a single incompressible solid phase. Assuming adsorption of the tracer but not the bulk fluid, the bulk and tracer fluid particles travel at different speeds. This can be seen more easily if the medium is considered as being compressible, i.e., ϕ depends also on time. Then (1.1) should be written as

$$\phi_t + \nabla \cdot \mathbf{u} = q, \quad \Omega \times J,$$

and we see that bulk fluid particles flow with velocity $\mathbf{u}/\phi$. However, tracer particles flow with velocity $\mathbf{u}/R\phi$, which is not the same if $R \neq 1$. Multiple characteristics cause difficulties, so we have assumed incompressibility of the medium (ϕ and R depend only on $\mathbf{x}$). Then (1.1) is equivalent to

$$(R\phi)_t + \nabla \cdot \mathbf{u} = q, \quad \Omega \times J, \tag{2.1}$$

with characteristic speed $\mathbf{u}/R\phi$. We remark that we could have instead assumed that $R = 1$ and allowed ϕ to depend on t. Our technique extends to this case.

We consider only flow during a time step, so let $\Delta t > 0$ be given and $t^n = n\Delta t$ for $n = 0, 1, \ldots$. As is usual, for a generic function φ, we denote $\varphi(t^n)$ by φ^n. The time interval of interest is $[t^n, t^{n+1}]$. The characteristic trace-back of the point $\mathbf{x} \in \Omega$ is denoted $\hat{\mathbf{x}} = \hat{\mathbf{x}}(\mathbf{x}; t)$, and it satisfies the (time backward) ordinary differential equation

$$\frac{d\hat{\mathbf{x}}}{dt} = \frac{\mathbf{u}(\hat{\mathbf{x}}, t)}{R(\hat{\mathbf{x}})\phi(\hat{\mathbf{x}})}, \quad t^n \le t < t^{n+1}, \tag{2.2}$$

$$\hat{\mathbf{x}}(t^{n+1}) = \mathbf{x}, \tag{2.3}$$

unless the particle traces to the boundary of the domain or is created at an external source. In either of these two exceptional cases, the particle arises from the outside world and must be treated accordingly.

3. The local mass and volume constraints. Let $\mathcal{T}$ be a finite partition of Ω into closed elements with disjoint interiors; that is, $\bar{\Omega} = \bigcup_{E\in\mathcal{T}} E$ and $\overset{\circ}{E}_i \cap \overset{\circ}{E}_j = \emptyset$ for all $i \ne j$. Often in our discussion $\mathcal{T}$ can be an arbitrary partition of Ω; however, sometimes we need more structure, so we tacitly assume that both Ω and the partition $\mathcal{T}$ are rectangular, with $\mathcal{T}$ having $M \times N$ elements. We use the usual Cartesian labeling of the grid points $\mathbf{x}_{i,j} = (x_i, y_j)$, where $i = 1, 2, \ldots, M+1$ and $j = 1, 2, \ldots, N+1$, and $x_i < x_{i+1}$ and $y_j < y_{j+1}$. Then rectangular element $E_{i,j} \in \mathcal{T}$ has vertices $\{\mathbf{x}_{i,j}, \mathbf{x}_{i+1,j}, \mathbf{x}_{i,j+1}, \mathbf{x}_{i+1,j+1}\}$. We also denote the midpoints of the element edges as $\mathbf{x}_{i+1/2,j} = (x_{i+1/2}, y_j)$ and $\mathbf{x}_{i,j+1/2} = (x_i, y_{j+1/2})$, where $x_{i+1/2} = (x_i + x_{i+1})/2$ and $y_{j+1/2} = (y_j + y_{j+1})/2$.

The key idea used to obtain mass conserving characteristic methods is to trace regions rather than points, an idea due to Arbogast and Wheeler [2, 3]. The particles in $E \in \mathcal{T}$ trace back to a region we call $\hat{E}$, defined by

$$\hat{E} = \{\hat{\mathbf{x}} \in \Omega : \hat{\mathbf{x}} = \hat{\mathbf{x}}(\mathbf{x}; t^n) \text{ for some } \mathbf{x} \in E\}.$$

In space and time, we actually trace a region $\mathcal{E} = \mathcal{E}(E)$ given by

$$\mathcal{E} = \{(\hat{\mathbf{x}}, t) \in \Omega \times [t^n, t^{n+1}] : \hat{\mathbf{x}} = \hat{\mathbf{x}}(\mathbf{x}; t) \text{ for some } \mathbf{x} \in E\}.$$

We decompose $\partial\mathcal{E}$ into three regions, viewing time as going "up." Then the "top" is E, where $t = t^{n+1}$; the "bottom" is $\hat{E}$, where $t = t^n$; and the "sides" we call $\mathcal{S}$. The sides come in two flavors. If particles trace back to the boundary of the domain, we have $\mathcal{S}_B = \partial\mathcal{E} \cap \partial\Omega \times (t^n, t^{n+1})$. Otherwise, we have the more usual interior side $\mathcal{S}_I = \partial\mathcal{E} \cap \Omega \times (t^n, t^{n+1})$. That is, we have the disjoint union

$$\partial\mathcal{E} = E \cup \hat{E} \cup \mathcal{S} = E \cup \hat{E} \cup \mathcal{S}_I \cup \mathcal{S}_B.$$

Local mass conservation of the tracer can be obtained over $E \in \mathcal{T}$ by integrating (1.3) in space-time over $\mathcal{E}$ and using the divergence theorem [3]. First, if $d\sigma$ is the space-time boundary differential on $\partial\mathcal{E}$, we compute

$$\begin{aligned}
\iint_{\mathcal{E}} q_c \, dx\, dt &= \iint_{\mathcal{E}} \nabla_{x,t} \cdot \begin{pmatrix} c\mathbf{u} \\ R\phi c \end{pmatrix} dx\, dt \\
&= \iint_{\partial\mathcal{E}} c \begin{pmatrix} \mathbf{u} \\ R\phi \end{pmatrix} \cdot \nu_{x,t} \, d\sigma \\
&= \int_E R\phi c^{n+1} \, dx - \int_{\hat{E}} R\phi c^n \, dx + \int_{\mathcal{S}} c \begin{pmatrix} \mathbf{u} \\ R\phi \end{pmatrix} \cdot \nu_{x,t} \, d\sigma,
\end{aligned}$$

since on the top and bottom of the domain $\nu_{x,t} = (0, \pm 1)$. The last term is integration on the space-time sides $\mathcal{S}$ of $\mathcal{E}$. Because the region $\mathcal{E}$ is defined by following characteristics, $\binom{\mathbf{u}}{R\phi}$ is orthogonal to $\nu_{x,t}$ on $\mathcal{S}_I$, that is, on $\mathcal{S}$ unless we have traced to the boundary of the domain. Thus the last term is

$$\int_{\mathcal{S}} c \begin{pmatrix} \mathbf{u} \\ R\phi \end{pmatrix} \cdot \nu_{x,t} \, d\sigma = \int_{\mathcal{S}_B} c\mathbf{u} \cdot \nu \, d\sigma,$$

since $\nu_{x,t} = \nu$ on $\mathcal{S}_B \subset \partial\Omega \times (t^n, t^{n+1})$.

The local mass constraint. Combining, we have the [illegible]

$$\int_E R\phi c^{n+1} \, dx - \int_{\hat{E}} R\phi c^n \, dx + \int_{\mathcal{S}_B} c\mathbf{u} \cdot \nu \, d\sigma = \iint_{\mathcal{E}} q_c \, dx \, dt. \tag{3.1}$$

This equation is the basis for approximation of (1.3) in the the characteristics-mixed method [2, 3], which uses piecewise discontinuous constant test functions defined over the space-time regions $\mathcal{E}$ (and the lowest order Raviart–Thomas mixed finite element [16] method for the parabolic part of the problem (1.4)).

The local volume constraint. A similar local mass constraint for the bulk fluid can also be derived analogously starting from the flow equation (1.1) or, equivalently, (2.1); that is, compute as above with $c \equiv 1$. Since we are dealing with incompressible fluids, we call this constraint the [illegible] and it is

$$\int_E R\phi \, dx - \int_{\hat{E}} R\phi \, dx + \int_{\mathcal{S}_B} \mathbf{u} \cdot \nu \, d\sigma = \iint_{\mathcal{E}} q \, dx \, dt. \tag{3.2}$$

As mentioned in the introduction, the reason why volume is not conserved locally is that the trace-back region $\hat{E}$ has an irregular shape, which must therefore be approximated by a simpler region $\tilde{E}$ so that the integrals in (3.1) can be evaluated. We propose to define $\tilde{E}$ as a simple octagonal shape that is approximately the original trace-back region $\hat{E}$. We do this by tracing back the four vertices and four midpoints of the edges of E and then adjusting the positions of these trace-back points locally until the volume constraint is satisfied (to some acceptably small tolerance). Recall that the characteristics are the solutions of a nonlinear ordinary differential equation (2.2)–(2.3), so these are approximated numerically. Thus the trace-back points $\hat{\mathbf{x}}$ are only approximately in the correct position to begin with, so some adjustment of them to maintain a system constraint seems quite reasonable.

4. Approximation of the integrals representing wells. We must approximate the integral of the source function in the volume balance equation (3.2). In porous medium applications, this function models wells, which have small scales that may require special consideration.

There is no need to treat the integral representing sink (i.e., production or extraction) wells in a special way, since they simply grow in size. If $q < 0$ on element E, the trace-back region $\hat{E}$ contains E. Thus

$$\iint_{\mathcal{E}} q \, dx \, dt = \int_{t^n}^{t^{n+1}} \int_E q \, dx \, dt \quad \text{and} \quad \iint_{\mathcal{E}} q_c \, dx \, dt = \int_{t^n}^{t^{n+1}} \int_E q_c \, dx \, dt,$$

and $\hat{E}$ can be adjusted as below in section 6 to satisfy (3.2). The only caveat is that we might choose to trace more than eight points on the boundary of E to obtain reasonable resolution of $\hat{E}$, since the trace-back points are diverging.

At an injection well ($q > 0$), fluid particles trace back into the well-bore where they arise. If we simply use the trapezoidal rule in time,

$$\iint_{\mathcal{E}} q\,dx\,dt \approx \frac{\Delta t}{2}\Big(\int_{\hat{E}} q^n\,dx + \int_E q^{n+1}\,dx\Big),$$

then if $\mathcal{S}_B$ is empty, the discrete volume constraint (3.2) becomes

$$\int_{\hat{E}} R\phi\,dx + \frac{\Delta t}{2}\int_{\hat{E}} q^n\,dx = \int_E R\phi\,dx - \frac{\Delta t}{2}\int_E q^{n+1}\,dx.$$

At injection, where $q > 0$, the left-hand side is positive, but the right-hand side could be [illegible] if Δt and/or q is large. Thus there is no adjustment of $\hat{E}$ that will result in local volume conservation. Since we desire to use large time steps with characteristic methods, simply restricting Δt is inappropriate.

Instead, we use a weighted trapezoidal rule with weight w between 0 and 1, as in

$$\iint_{\mathcal{E}} q\,dx\,dt \approx w\Delta t\int_{\hat{E}} q^n\,dx + (1-w)\Delta t\int_E q^{n+1}\,dx, \tag{4.1}$$

so that the volume constraint becomes

$$\int_{\hat{E}} R\phi\,dx + w\Delta t\int_{\hat{E}} q^n\,dx = \int_E R\phi\,dx - (1-w)\Delta t\int_E q^{n+1}\,dx. \tag{4.2}$$

We want to define w so that adjustment of $\hat{E} \subset E$ to $\tilde{E}$ results in volume conservation. We must adjust $\hat{E}$ to $\tilde{E}$ somewhere between the extreme cases of $\tilde{E} = E$ and $\tilde{E}$ being a point of size zero. That is, we need

$$\int_E R\phi\,dx + w\Delta t\int_E q^n\,dx \ge \int_E R\phi\,dx - (1-w)\Delta t\int_E q^{n+1}\,dx \ge 0, \tag{4.3}$$

so that we can define $\tilde{E}$ such that

$$\int_{\tilde{E}} R\phi\,dx + w\Delta t\int_{\tilde{E}} q^n\,dx = \int_E R\phi\,dx - (1-w)\Delta t\int_E q^{n+1}\,dx. \tag{4.4}$$

To determine an appropriate w, we argue as follows. Suppose that there is an injection well in element E. Since fluid is being injected, $q > 0$ and we expect trace-back[illegible] the element. Assume that $E = [-h/2, h/2]^2$ is square, q and $R\phi$ are constant on E, the solution is symmetric near the well, and the velocity is, say, approximated by a lowest order Raviart–Thomas mixed finite element [16]. With these assumptions, the solution is found as follows. First

$$qh^2 = \int_E q\,dx\,dy = \int_E \nabla\cdot\mathbf{u}\,dx\,dy = \int_{\partial E}\mathbf{u}\cdot\nu\,d\sigma = 4\mathbf{u}\cdot\nu h,$$

since $\mathbf{u}\cdot\nu$ is assumed constant on ∂E. Thus $\mathbf{u}\cdot\nu = qh/4$ and the corresponding Raviart–Thomas function in E is

$$\mathbf{u}(x,y) = \frac{q}{2}\begin{pmatrix} x \\ y \end{pmatrix}.$$

It is now easy to follow the characteristics (2.2)–(2.3). For $\mathbf{x} \in E$, we have that

$$\hat{\mathbf{x}}(x) = \mathbf{x}\exp\big(q(t-t^{n+1})/2R\phi\big),$$

and so $\hat{E} = [-e^{-\alpha/2}h/2, e^{-\alpha/2}h/2]^2$, where $\alpha = q\Delta t/R\phi \in (0,\infty)$. Thus the volume balance equation (3.2) is

$$R\phi h^2 - e^{-\alpha}R\phi h^2 = \int_{\mathcal{E}} q\,dx\,dt \approx [e^{-\alpha}w + (1-w)]q\Delta t\,h^2,$$

applying the weighted trapezoidal rule to the time integration. We choose the weight w so that this volume balance approximation is exact; that is,

$$w = w(\alpha) = \frac{1}{1-e^{-\alpha}} - \frac{1}{\alpha}. \tag{4.5}$$

Note that $w(0^+) = 1/2$, $w(+\infty) = 1$, and

$$w'(\alpha) = \frac{-e^{-\alpha}}{(1-e^{-\alpha})^2} + \frac{1}{\alpha^2} = \frac{1}{4\sinh^2(\alpha/2)}\left[\left(\frac{\sinh(\alpha/2)}{\alpha/2}\right)^2 - 1\right] \geq 0,$$

since $\sinh(\beta) \geq \beta$ for $\beta \geq 0$. It follows that

$$1/2 \leq w(\alpha) < 1 \quad \text{for all } \alpha > 0.$$

Now in the general case we should define w by (4.5) using

$$\alpha = \frac{\int_E q^{n+1}\,dx\,\Delta t}{\int_E R\phi\,dx}. \tag{4.6}$$

Since $w < 1$, we have the first inequality in (4.3), and the second follows easily. Adjustment of $\hat{E}$ in section 6 can then result in satisfaction of the volume constraint (4.2). Once this is done, the mass constraint (3.1) is then computed analogously from

$$\begin{aligned}\int_E R\phi c^{n+1}\,dx - \int_{\hat{E}} R\phi c^n\,dx + \int_{\mathcal{S}_B} c\mathbf{u}\cdot\nu\,d\sigma \\ = w\Delta t\int_{\hat{E}} q_c^n\,dx + (1-w)\Delta t\int_E q_c^{n+1}\,dx.\end{aligned} \tag{4.7}$$

We remark that the weighted trapezoidal rule is accurate to order Δt, which is the same rate of convergence in time expected of the characteristics-mixed method (see [3]). The trapezoidal rule itself is accurate to order Δt^2 but gives worse results over time because it leads to violation of local volume balance.

5. Approximation of inflow boundaries. Inflow boundaries are similar to internal sources in that particles trace to the boundary where they enter the system. Again, some care is needed to approximate this process. We use a technique that allows us to "trace out of the domain" in some sense. The term of concern in the volume balance equation (3.2) is

$$\int_{\mathcal{S}_B} \mathbf{u}\cdot\nu\,d\sigma,$$

which has an irregular shape on the space-time boundary $\partial\Omega\times(t^n, t^{n+1})$. The idea is to "fold" the time axis down to the xy-plane to create a "ghost region" (see Figure 5.1) by extending the spatial domain by an amount $t^{n+1} - t^n$ outside Ω across the inflow boundary. (The figure shows a relatively weak inflow for which no element is traced

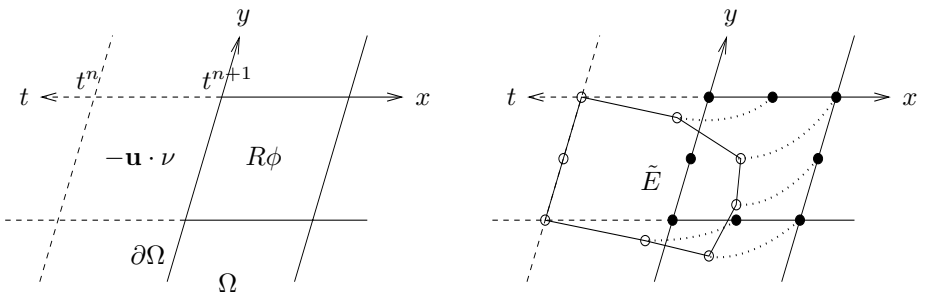


FIG. 5.1. *The inflow space-time boundary "folded" into the xy-plane, and the space-time trace-back region* $\tilde{E}$.

entirely into the folded region; however, it is possible to trace several layers of elements back into this region.)

Given an element E, one traces a vertex or midpoint from time t^{n+1} as usual. The only question is what velocity to use outside the domain Ω. One uses unit speed in the direction of ν, so that the points lying on the inflow boundary of Ω trace to the $t = t^n$ boundary of the folded domain.

This procedure defines the generalized space-time trace-back region $\tilde{E}$, which is decomposed into $\tilde{E} \cap \Omega$, the usual trace-back region, and a piece $\tilde{\mathcal{S}}_B = \tilde{E} \setminus \tilde{E} \cap \Omega$ which approximates $\mathcal{S}_B$. Thus the volume constraint is simply

$$\int_E R\phi \, dx - \int_{\tilde{E}\cap\Omega} R\phi \, dx + \int_{\tilde{\mathcal{S}}_B} \mathbf{u}\cdot\nu \, d\sigma = \int_E R\phi \, dx - \int_{\tilde{E}} R\phi \, dx = \iint_{\mathcal{E}} q \, dx \, dt, \tag{5.1}$$

wherein $R\phi$ is defined to be $\mathbf{u}\cdot\nu$ in the ghost region.

The local mass constraint defining the characteristics method is similarly computed by replacing the quantity $R\phi c^n$ by $c_I^n \mathbf{u}\cdot\nu$ in the ghost region, where c_I^n is the given inflow concentration; that is, we have simply

$$\int_E R\phi c^{n+1} \, dx - \int_{\tilde{E}} R\phi c^n \, dx = \iint_{\mathcal{E}} q_c \, dx \, dt. \tag{5.2}$$

In the way described in this section, we avoid explicit reference to the inflow boundary and reduce its computations to the same ones used in the interior of the domain. There is no need to treat outflow boundaries in a special way, since the trace-back region will remain within the spatial domain Ω. An alternative is to use the trace-forwarding idea of section 7, especially if the flow is quite strong.

6. Trace-back volume adjustment. We begin the process by tracing back the four vertices $\{\mathbf{x}_{i,j},\ \mathbf{x}_{i+1,j},\ \mathbf{x}_{i,j+1},\ \mathbf{x}_{i+1,j+1}\}$ and the four midpoints of the four edges $\{\mathbf{x}_{i+1/2,j},\ \mathbf{x}_{i+1/2,j+1},\ \mathbf{x}_{i,j+1/2},\ \mathbf{x}_{i+1,j+1/2}\}$ of $E_{i,j} \in \mathcal{T}$ to form an octagon $\tilde{E}$ approximating $\hat{E}$ with corner points $\{\hat{x}_{i,j},\ \hat{x}_{i+1,j},\ \hat{x}_{i,j+1},\ \hat{x}_{i+1,j+1},\ \hat{x}_{i+1/2,j},\ \hat{x}_{i+1/2,j+1},\ \hat{x}_{i,j+1/2},\ \hat{x}_{i+1,j+1/2}\}$. We continue to call the first four corner points "vertices," and the last four "midpoints." We adjust only these eight points to enforce the local volume constraint.

We assume in this section that every element traces back without hitting the boundary of the domain. This is always possible using the space-time "folding" technique presented in section 5. That is, we may trace an element E out of Ω as described,

and obtain what for algorithmic purposes looks like an octagon $\tilde{E}$ that can be adjusted as needed.

Our only restriction is related to external boundaries with no-flow, where we should respect this condition and disallow certain adjustments. If $\mathbf{x} \in \partial\Omega$ and the velocity field arises from the Stokes or Navier–Stokes equations with $\mathbf{u}(\mathbf{x}) = 0$, we should not allow any adjustment of the point $\hat{\mathbf{x}} = \mathbf{x}$. If only $\mathbf{u}(\mathbf{x}) \cdot \nu(\mathbf{x}) = 0$, we should not allow adjustment of $\hat{\mathbf{x}}$ in the normal direction (unless perhaps $\mathbf{x}$ traces back along $\partial\Omega$ to a point where the boundary condition changes to a type that allows such movement).

6.1. Trace-back point adjustment along characteristics. When adjusting trace-back points, care must be taken to avoid introducing systematic bias into the transport computation. We use an idea first described by Douglas, Huang, and Pereira [10], who achieved a [illegible] mass balance in the modified method of characteristics [12] by adjusting the trace-back points [illegible]. That is, the trace-back points are adjusted in the direction of the flow field, along the characteristics, so that no bias is introduced into the [illegible] of the flow. The effect is to convert spatial errors into time errors. Spatial errors tend to grow over time, but time errors are better behaved. However, Douglas, Huang, and Pereira applied their idea globally to all points simultaneously each time step. We will apply the idea locally in space.

To adjust a group of trace-back points, simultaneously adjust all of them by solving the characteristic governing equation (2.2) over a time interval $\tau^n \le t < t^{n+1}$ for some $\tau^n \approx t^n$. This changes each position along the characteristic by a small amount proportional to the local speed and the difference $\tau^n - t^n$.

We begin our trace-back point adjustment near external sources of bulk fluid, either injection wells or inflow boundaries, and proceed outward in "layers," as described subsequently.

6.2. Trace-backs near external sources. Using the integral approximation described in section 4, we adjust simultaneously each trace-back point of an injection well boundary until its volume is within tolerance of the target value given by (4.2). Consider now these trace-back points as being fixed in position.

For inflow boundaries, we fix the trace-back of the boundary points into the ghost region, as in section 5.

These techniques work well for milder flows. For stronger flow fields (or, equivalently, longer time steps), the trace-forwarding scheme of section 7 below is a better alternative.

6.3. Trace-back adjustment of layers away from sources. Once an external source has been treated, we proceed to adjust the layer of adjoining elements to obtain volume conservation in two steps, as illustrated in Figure 6.1. These adjustments are computationally quite fast.

First, simultaneously adjust all the trace-back points on the nonfixed boundary of the layer along the characteristics in time. That is, solve (2.2) over a time interval $\tau^n \le t < t^{n+1}$ for some $\tau^n \approx t^n$, so that the sum of the volumes of the elements in the layer meets the volume constraint (3.2), which in the absence of other sources, inflow boundaries, and sinks is simply

$$\sum_{E \text{ in the layer}} \int_E R\phi \, dx = \sum_{E \text{ in the layer}} \int_{\tilde{E}} R\phi \, dx.$$

We meet this constraint to some small tolerance using a type of "bisection" algorithm

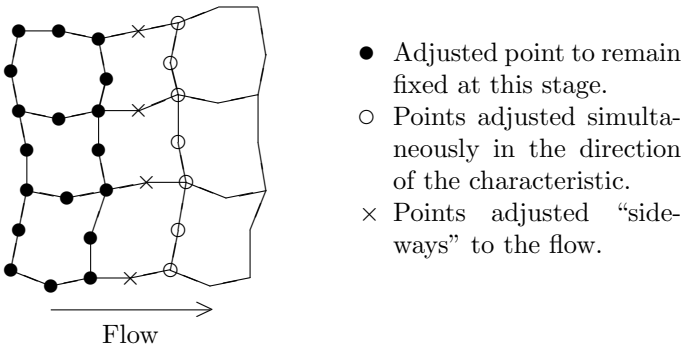


FIG. 6.1. *The adjustment of the trace-back regions. At each adjustment stage, the points marked with a solid dot remain fixed. In the first step, the points marked with a circle are simultaneously adjusted to obtain volume balance of the entire layer. In the second step, the points marked with a cross are adjusted to obtain volume balance for each element.*

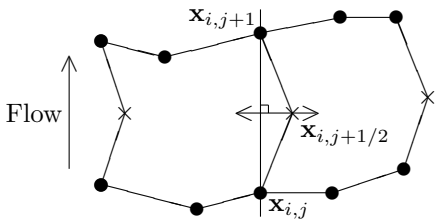


FIG. 6.2. *The adjustment of the interior midpoints. The points marked with a solid dot remain fixed, and the points marked with a cross are adjusted to obtain volume balance for each element.*

converging to the correct τ^n. This method works well since points trace toward the sources backward in time, and so $\tau^n < t^n$ adjusts the region to have less volume while $\tau^n > t^n$ gives more volume. At the conclusion of this step, we have volume balance for the layer, but not for each element.

The second step is to adjust the interior midpoints of the layer to obtain volume balance of each element. We choose to adjust in the direction of maximal change to the volume (so that points move minimally). As depicted in Figure 6.2, we adjust, e.g., $\mathbf{x}_{i,j+1/2} = (x_i, y_{j+1/2})$ along the line through its unadjusted position and perpendicular to the line adjoining vertices $\mathbf{x}_{i,j}$ and $\mathbf{x}_{i,j+1}$. On an exterior no-flow boundary where $\mathbf{u}(\mathbf{x}) \cdot \nu(\mathbf{x}) = 0$, points should not be adjusted, since they can move only tangentially along the boundary, and so do not change volumes.

The trace-back layers can be of several types. Around an injection well, we may have a closed ring, in which case we may fix one of the midpoints and adjust the rest in sequence around the ring. The volume of the entire ring is correct, so adjustment of the last point will correct the volumes of the final two elements. We may also have a simple layer which meets the external boundary on two ends. Even if we are not allowed to change the end trace-back midpoints (because of no-flow boundary conditions), again we can meet the volume constraint for each element since the entire layer has the proper volume.

Things get more complex with multiple injection wells and/or inflow boundaries, since these can produce multiple basins of attraction. In that case, care must be taken when multiple trace-back adjustment layers intersect. In principle this problem can be resolved, either through human intervention or automatically. Automatic handling

involves analyzing intricate topological complexities that are beyond the scope of this work and are left for future research.

For small times, the trace-back regions of elements are well defined. However, if the time step is too large, an approximate trace-back region $\tilde{E}$ might intersect itself. This is unacceptable, and it indicates that the boundary of the true trace-back region $\hat{E}$ is convoluted (we should conclude this also if $|\tau^n - t^n|/\Delta t$ is relatively large). In this case one should either increase the number of points traced per element edge or, more practically, reduce the time step. This is in principle the only limitation on the time step Δt. This problem is most prevalent near injection wells, and we present an alternative strategy in the next section.

7. Trace-forwarding of strong injection wells (and inflow boundaries). Near injection wells, we have observed that before adjustment, relative volume errors of the trace-back regions $\tilde{E}$ can easily exceed 100%; thus, we must handle these regions carefully. The basic problem is that, when tracing backward in time, regions trace [illegible] the very small well-bore, significantly distorting the boundary of the true trace-back regions $\hat{E}$ near the well. The natural solution is to trace forward in time instead of backward. This idea was first advocated by Healy and Russell [13].

We trace points forward in time by solving the time-forward equation for the characteristics

$$\frac{d\check{\mathbf{x}}}{dt} = \frac{\mathbf{u}(\check{\mathbf{x}}, t)}{R(\check{\mathbf{x}})\phi(\check{\mathbf{x}})}, \quad t^n < t \le t^{n+1}, \tag{7.1}$$

$$\check{\mathbf{x}}(t^n) = \mathbf{x}. \tag{7.2}$$

The time-forward local volume constraint for element E, analogous to (3.2), is

$$\int_{\check{E}} R\phi \, dx - \int_E R\phi \, dx + \int_{\mathcal{S}_{f,B}} \mathbf{u} \cdot \nu \, d\sigma = \iint_{\mathcal{E}_f} q \, dx \, dt, \tag{7.3}$$

where now

$$\mathcal{E}_f = \{(\check{\mathbf{x}}, t) \in \Omega \times [t^n, t^{n+1}] : \check{\mathbf{x}} = \check{\mathbf{x}}(\mathbf{x}; t) \text{ for some } \mathbf{x} \in E\}$$

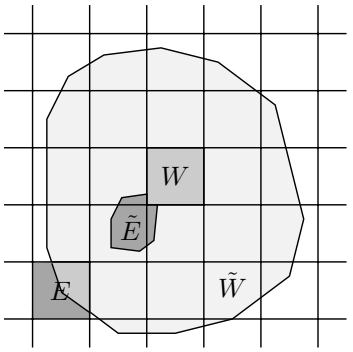
and $\mathcal{S}_{f,B} = \mathcal{E} \cap \partial\Omega \times [t^n, t^{n+1}]$.

We present now a technique for handling strong injection wells that replaces the ideas of section 4. It has four main steps, and is illustrated in Figure 7.1. Let W represent the element containing the well (we tacitly assume the well fills the element).

First, we define the initial region $\tilde{W}$. The region of space filled by fluid from the well, $\check{W}$, is defined by tracing each point on ∂W forward in time. This region is approximated by a polygon $\tilde{W}$, given by joining the traces of a finite number of boundary points. As mentioned in section 4, we might choose to trace more than our customary eight boundary points to obtain reasonable resolution of $\check{W}$, since now the trace-forward points diverge.

The second step is to adjust $\tilde{W}$ so that it satisfies the volume constraint. We simultaneously adjust the boundary points in time by solving (7.1) over the perturbed time interval $t^n < t \le \tau^{n+1} \approx t^{n+1}$ until the time-forward local volume constraint (7.3) is satisfied by the well, which is

$$\int_{\tilde{W}} R\phi \, dx = \int_W R\phi \, dx + \iint_{\mathcal{E}_f} q \, dx \, dt, \tag{7.4}$$



W is the injection well element.
$\tilde{W}$ is the approximation to the trace-forward of ∂W.
E is an element cut by $\partial\tilde{W}$.
$\tilde{E}$ is the approximation to the trace-back of $E \setminus \tilde{W}$.

FIG. 7.1. *Trace-forwarding around an injection well. Over the time step, part of the fluid that flows into E comes from $\tilde{E}$, and the rest comes from the well W.*

assuming no flow on $\mathcal{S}_{f,B}$, wherein, assuming q vanishes outside W, $\mathcal{E}_f$ can be replaced by $W \times [t^n, t^{n+1}]$.

Consider now an element E affected by the well, i.e., such that $E \cap \tilde{W} \neq \emptyset$. Our third step is to define $\tilde{E}$. Points within $\tilde{W}$ need not be traced back, since we know that they trace into the well. Thus, if $E \subset \tilde{W}$, $\tilde{E} = \emptyset$. So consider an E cut by $\partial\tilde{W}$ (i.e., $E \cap \partial\tilde{W} \neq \emptyset$). We trace back any of our usual eight points on ∂E that lie outside $\tilde{W}$. We also trace back the (most likely two) points $\partial E \cap \partial\tilde{W}$, which must trace to ∂W (if not, simply adjust the time interval so as to hit the well). Polygonal approximation defines a region $\tilde{E}$ outside W.

The fourth and final step is to adjust $\tilde{E}$ to satisfy the local volume constraint. We simultaneously adjust the points of $\partial\tilde{E}$ not on ∂W backward in time using (2.2) over a time interval $t^n \approx \tau^n \leq t < t^{n+1}$ to satisfy the local volume constraint outside the well, which is

$$\int_E R\phi \, dx = \int_{\tilde{E}} R\phi \, dx + \frac{|E \cap \tilde{W}|}{|\tilde{W}|} \iint_{W \times [t^n, t^{n+1}]} q \, dx \, dt,$$

since the region $E \cap \tilde{W}$ receives a proportional amount of fluid from the well.

We note that the transport itself, given above in the local mass constraint (3.1), is correspondingly modified as

$$\int_E R\phi c^{n+1} \, dx = \int_{\tilde{E}} R\phi c^n \, dx + \iint_{W \times [t^n, t^{n+1}]} q_c \, dx \, dt.$$

A similar construction can be applied near strong inflow boundaries. In that case, the construction of section 5 is not needed, since points no longer trace out of the domain.

8. Numerical experiments. In this section we illustrate the volume balance techniques in five examples. The last example has an analytic solution, but in the others, we determine the Darcy velocity $\mathbf{u}$ of the fluid in terms of the pressure p according to Darcy's law:

$$\mathbf{u} = -\frac{k}{\mu} \nabla p, \tag{8.1}$$

where μ is the viscosity and k is the permeability. We assume that the wells q are independent of time t, so that it suffices to solve for $\mathbf{u}$ once from the divergence constraint (1.1), subject to the boundary conditions, which vary with the example.

We assume that μ is constant (i.e., the concentrations of the components are too small to affect the viscosity of the fluid), and has the value 0.01 poise (i.e., water) in our examples. However, k varies with the example, and it is heterogeneous. A measure of the variability of k is given by the dimensionless coefficient of variation

$$C_v = \frac{1}{\mu_k}\left(\frac{1}{|\Omega|}\int_\Omega (k(x)-\mu_k)^2\,dx\right)^{1/2}, \tag{8.2}$$

which is the ratio of the standard deviation of k and its mean value

$$\mu_k = \frac{1}{|\Omega|}\int_\Omega k(x)\,dx. \tag{8.3}$$

Since the local volume constraint is really local conservation of the bulk fluid, we believe that it is important to use a locally conservative method to approximate the solution of (8.1). We use a standard lowest order Raviart–Thomas mixed method [16] implemented as cell-centered finite differences [17, 4].

We will be solving (1.2), wherein the diffusion/dispersion tensor D is given by the formula

$$D = \phi\, d_{\mathrm{mol}} I + |\mathbf{u}|[(d_{\mathrm{long}} E(\mathbf{u}) + d_{\mathrm{trans}} E^{\perp}(\mathbf{u}))], \tag{8.4}$$

where $E(\mathbf{u})$ denotes projection along the vector $\mathbf{u}$, $E^{\perp}(\mathbf{u})$ its orthogonal complement, and where the molecular diffusion coefficient $d_{\mathrm{mol}} = 10^{-7}$ cm^2/sec, the longitudinal dispersion coefficient $d_{\mathrm{long}} = 10^{-3}$ cm, and the transverse dispersion coefficient $d_{\mathrm{trans}} = 10^{-4}$ cm. Again we use Raviart–Thomas spaces to approximate this part of the operator (see [2, 3] for details).

In this section, we contrast our three schemes: the unmodified characteristics-mixed method (CMM) [2, 3], our volume conserving CMM without trace-forwarding of the well (VCCMM), and our volume conserving CMM with trace-forwarding of the well (VCCMM-TF).

8.1. A nuclear contamination problem. In our first example, we describe a nuclear contamination problem in a water saturated porous medium. The domain Ω is a square of size 256 m $\times$ 256 m. The permeability k is depicted in Figure 8.1. The

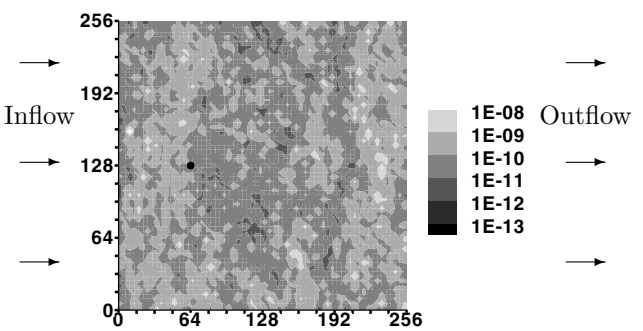


FIG. 8.1. *The nuclear contamination problem domain and permeability field, in cm^2. The location of the injection well is marked with a black dot at* $(64, 128)$.

permeability is log-normal and fractal, with mean 2×10^{-10} cm^2 (about 20 md) and a variation of over five orders of magnitude from around 10^{-13} to 10^{-8} cm^2 (about 0.01 md to 1 darcy). Its coefficient of variation (8.2) is $C_v = 0.522$.

Our assumption that μ is constant merely says that the concentrations of the nuclear components are too small to affect the viscosity of the fluid. We impose the boundary conditions of no normal flow $\mathbf{u}\cdot\nu = 0$ on the "top" and "bottom" faces of the domain ($y = 0$ m and $y = 256$ m, respectively), and a constant pressure at the left face of 5×10^5 g/cm·sec^2 and a constant pressure at the right face of -5×10^5 g/cm·sec^2, giving rise to a background flow from left to right. There is a single injection well covering 4 grid cells and placed at $(64, 128)$ m. We inject one pore volume every 50 years.

We transport a decaying nuclear component of concentration c. It satisfies the advection-diffusion equation (1.2), with a reaction term added in:

$$(R\phi c)_t + \nabla \cdot (c\mathbf{u} - D\nabla c) = -\lambda\phi R c + q_c(c), \quad \Omega \times J. \tag{8.5}$$

We take the porosity $\phi = 0.01$, the retardation factor $R = 12$, and the decay constant (for ^{234}U) $\lambda = 9 \times 10^{-14}$/sec. We inject into an initially clean aquifer a concentration c_I of 1×10^{-5} g/cm^2.

In the first set of runs, we consider a 64×64 rectangular grid and contrast four schemes: the unmodified characteristics-mixed method (CMM) [2, 3], a higher order Godunov method [18, 5, 9, 15, 1], and our volume conserving CMM with and without trace-forwarding near the well (VCCMM and VCCMM-TF).

Explicit methods in one dimension have a time step restriction [15], known as the CFL time-step, given by

$$\Delta t \le \Delta t_{\text{CFL,1-D}} = \max_{\mathbf{x}\in\Omega} \frac{hR\phi(x)}{|\mathbf{u}(\mathbf{x})|},$$

where h is the grid spacing. Since we are working in two dimensions, we should limit Δt to half this value,

$$\Delta t \le \Delta t_{\text{CFL,2-D}} = \max_{\mathbf{x}\in\Omega} \frac{hR\phi(x)}{2|\mathbf{u}(\mathbf{x})|},$$

for the Godunov method. For our 64×64 grid, the two-dimensional CFL limited time-step is 0.586 year, but we use a much larger time step for the VCCMM.

We obtain the trace-back regions by tracing back the corners and midpoints of each grid element, and connecting the points by straight lines. On the left side of Figure 8.2, we plot these trace-back regions near the injection well for the CMM and the corresponding volume (i.e., area) errors, using a time step of only 1.5 years, which is about 3 times the CFL limit. We see up to 292% volume balance error at the well. Most of the regions are beyond $\pm 1\%$, and many beyond $\pm 10\%$. Starting from these trace-back regions, the VCCMM adjustment algorithm (without trace-forwarding around the well) gives the trace-back regions on the right side of Figure 8.2, in which only floating point rounding error is observed (up to only about 10^{-9} volume balance error).

We show the contour plots of the contaminant concentration c at 30 years in Figures 8.3 and 8.4 for the four methods. The shape and sharpness of the front, and the presence of undershoot or overshoot, are to be noted. The CMM, while locally mass conservative, does not conserve volume, so the concentrations exhibit

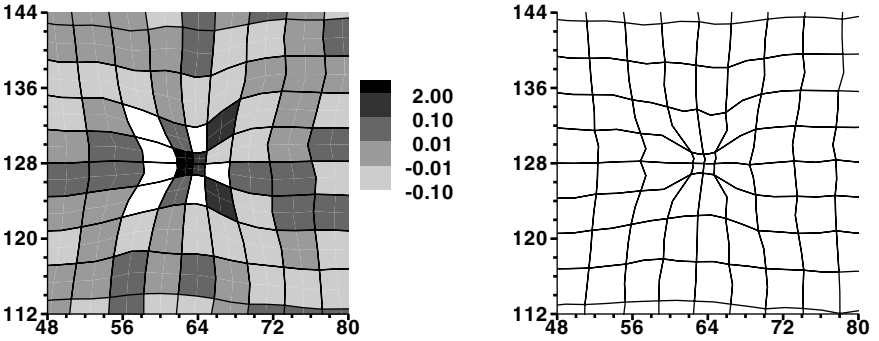


FIG. 8.2. *The nuclear contamination problem trace-back regions near the well, with CMM on the left and VCCMM (without trace-forwarding near the well) on the right. The relative volume errors are shown for CMM. The VCCMM volume errors are on the order of* 10^{-9}.

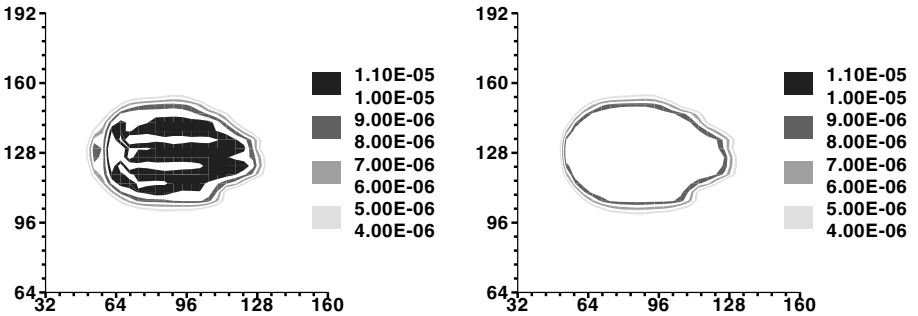


FIG. 8.3. *The nuclear contamination problem CMM (left) and VCCMM (right) concentration at* 30 *years on the* 64×64 *grid, in g/cm*^{2}*, using a time step of* 1.5 *years.*

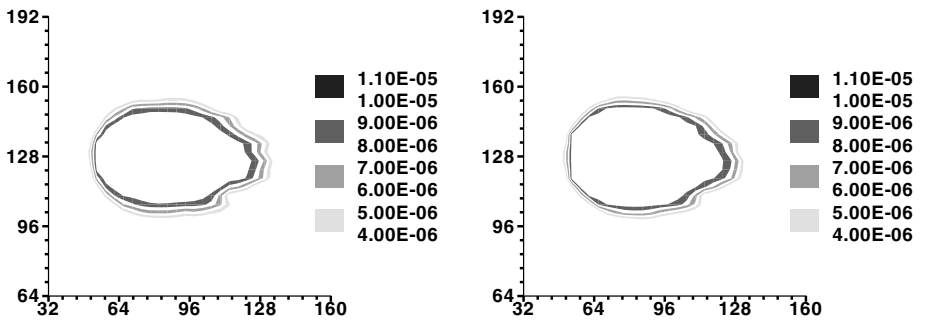


FIG. 8.4. *The nuclear contamination problem higher order Godunov (left) and VCCMM-TF (right) concentration at* 30 *years on the* 64×64 *grid, in g/cm*^{2}*. We used the CFL time step* 0.586 *year for Godunov and* 3 *years for VCCMM-TF.*

significant overshoot, reaching 134% of the maximum possible value near the well using a time step of 1.5 years. Moreover, volume nonconservation introduces many nonphysical local minima and maxima into the solution. The VCCMM corrects these problems and gives a solution very similar to the higher order Godunov method using $\Delta t = 0.586$ year. Moreover, attention to the well in the VCCMM-TF allows us to use a much larger time step of 6 years. These three methods exhibit no undershoot

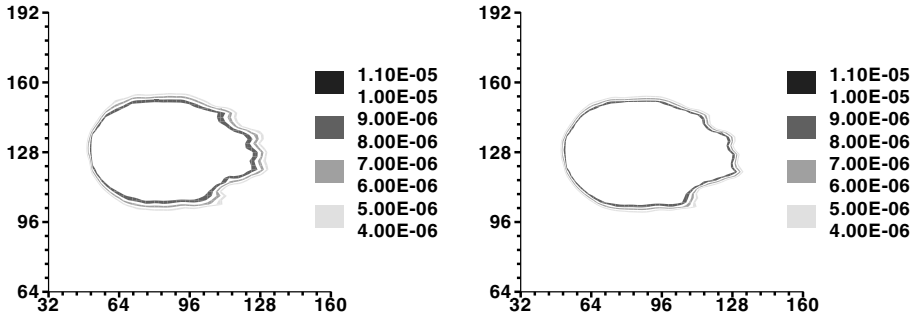


FIG. 8.5. *The nuclear contamination problem higher order Godunov (left) and VCCMM-TF (right) concentration at* 30 *years on the* 128×128 *grid, in g/cm*^{2}. *We used the CFL time step* 0.146 *years for Godunov and* 1 *year for VCCMM-TF.*

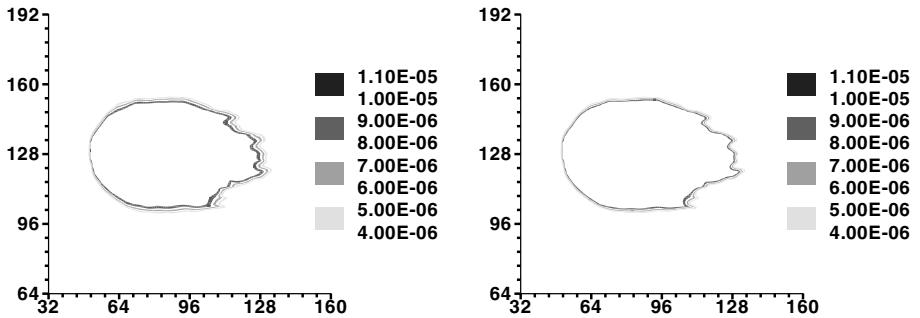


FIG. 8.6. *The nuclear contamination problem higher order Godunov (left) and VCCMM-TF (right) concentration at* 30 *years on the* 256×256 *grid, in g/cm*^{2}. *We used the CFL time step* 0.0366 *years for Godunov and* 0.5 *years for VCCMM-TF.*

or overshoot, and each predicts a similar front shape. However, the VCCMM and VCCMM-TF are less numerically diffuse, in that the front is sharper (due primarily to using a larger time step).

Finally, in Figures 8.5 and 8.6, we show the results of using finer grids: 128×128 and 256×256. The front sharpens under refinement, indicating that most of the spread is indeed due to the numerical methods and not the physical diffusion and dispersion. Again, the higher order Godunov and VCCMM-TF give comparable results, but the VCCMM-TF is somewhat less numerically diffusive and uses a much larger time step. The CFL step for the 128×128 grid is $\Delta t_{\text{CFL,2-D}} = 0.146$ years (205 steps to 30 years), but the VCCMM-TF used $\Delta t = 1$ year (30 steps), which is 6.8 times the CFL limit. For the 256×256 grid, $\Delta t_{\text{CFL,2-D}} = 0.0366$ years (820 steps), but the VCCMM-TF used $\Delta t = 0.5$ years (60 steps), which is 13.7 times the CFL limit.

8.2. A quarter five-spot flood. The second example is related to miscible flooding in a petroleum reservoir. We consider a quarter of a "five-spot" pattern of wells, which is a rectangular domain with an injection well in the lower left corner and a production well in the upper right corner, and no-flow (homogeneous Neumann) boundary conditions. Our domain is 40×40 meters square, and we impose an 80×80 grid over the domain. It is initially clean: $c(\mathbf{x}, 0) = 0$. The injector covers 4 cells near the corner $(0, 0)$ meters and has a rate of 1 pore volume every 5 years, injecting an inert tracer with concentration $c_I = 1$. The 4 cell comprising the producer are near the

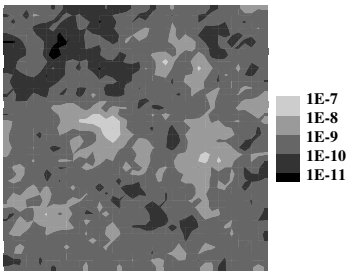


FIG. 8.7. *The quarter five-spot permeability field, in* cm^2.

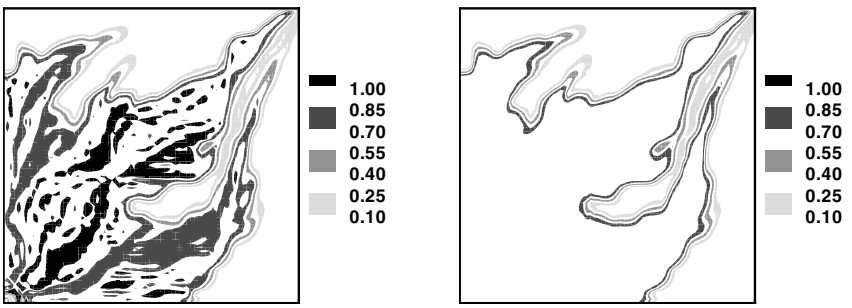


FIG. 8.8. *The quarter five-spot CMM (left) and VCCMM (right) concentration at* 3.36 *years.*

opposite corner $(40, 40)$ meters and its rate is opposite that of the injector. We solve (1.2) with no retardation (i.e., $R = 1$), no reactions, a constant porosity of 0.25, and the heterogeneous permeability k depicted in Figure 8.7, which was geostatistically generated and has mean 100 md (10^{-9} cm^2) and $C_v = 2.58$ and varies by about 4 orders of magnitude.

In Figure 8.8 we plot the results at 3.36 years using 280 steps of 0.012 year/step, which is 10.68 times the 2-D CFL number. The CMM shows both overshoot and undershoot in what should be the swept region, while the VCCMM shows no such artifacts. The poor performance of the CMM can be attributed to the extreme local heterogeneity of the permeability ($C_v = 2.58$), which produces large volume imbalances throughout the domain. In fact the field is so heterogeneous that a slightly larger time step of 0.0136 year/step (12.10 times the CFL number) initially creates degenerate trace-back regions, which cannot be used.

8.3. A fluvial domain. The third example is related to a fluvial geological environment that is correlated over large distances. The problem is taken from White and Horne [19], and the permeability is depicted in Figure 8.9, which clearly shows an ancient river bed with high permeability. We use only three values for the permeability, with mean 4.056 Darcy and $C_v = 1.15$. The domain is 600×600 feet squared and posed on a 30×30 grid. There is an injection well at grid cell $(1, 1)$ and a production well at grid cell $(30, 30)$. The injection concentration is $c_I = 1$, and the rate is 1 pore volume every 3 years. The production rate is opposite, and there are no-flow boundary conditions. The porosity is constant at 0.2, and c is initially set to zero. We solve the problem on a refined 60×60 computational grid to improve resolution (with wells now filling 4 cells). We use a time step of 0.015 years, which is 12.84 times the 2-D CFL limit.

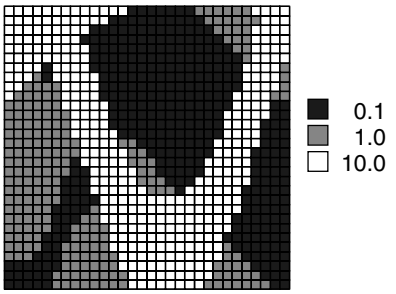


FIG. 8.9. *The fluvial domain example. The permeability field, in darcies, with the injector in the lower left and the producer in the upper right.*

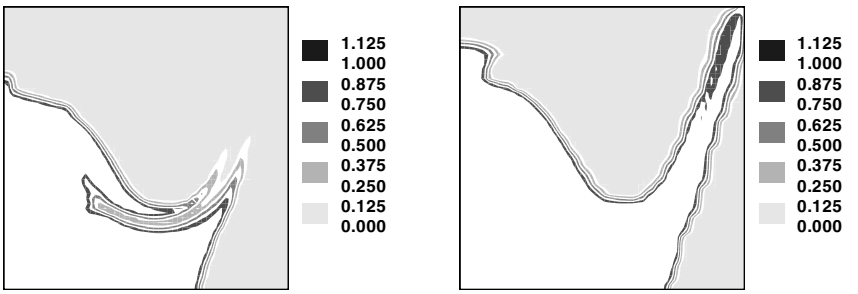


FIG. 8.10. *The fluvial domain VCCMM-TF concentration at* 1.05 *years (left) and* 1.65 *years (right).*

In Figure 8.10, we show results at 1.05 and 1.65 years (70th and 110th steps). Notice how the tracer follows the ancient river channel and avoids the lower permeability regions. The plume advances initially in time on both sides of the very low permeability region containing the well, as seen at 1.05 years, and reconnects further along in time within the very high permeability channel. As is evident at 1.65 years, within the high permeability channel, the top portion of the plume travels a greater distance, and so is retarded in position compared to the bottom half of the plume. Each half of the plume near the production well at 1.65 years is only a few grid elements in width. The plume is connected; however, the plotting routine cannot handle such a thin plume and shows some disconnection. We see a similar illusion at 1.05 years where the plume is very thin.

Serious relative volume errors are observed both at the wells and along the ancient riverbank, where a "stair-step" pattern of grid elements is used to approximate its shape. Using initial trace-back regions, the wells are so poorly approximated that the CMM actually produces negative concentrations on a 30×30 grid and time step of only 0.01 years, indicating a twisting of the trace-back regions so that they self intersect.

To isolate the riverbank errors, we applied only the trace-forwarding well adjustment to the injection well and adjusted volumes near the production well. The result of CMM using this partially processed trace-back mesh is depicted in Figure 8.11. We obtained significant improvement over straight CMM (not pictured), but the front is advanced in the CMM results compared to the VCCMM-TF, since the CMM with well adjustment simulation has average concentrations in the plume below 0.875. Thus the

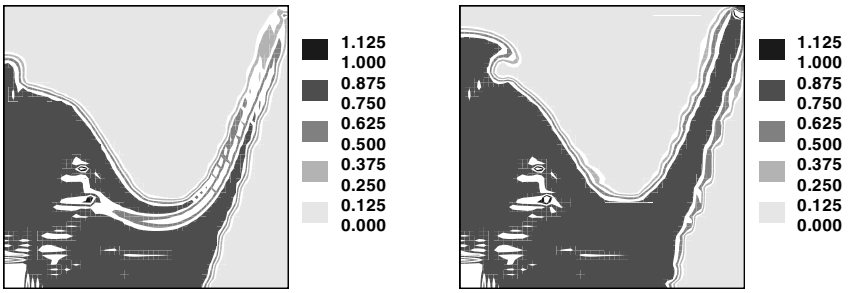


FIG. 8.11. *The fluvial domain CMM with well adjustments only. Concentration at* 1.05 *years (left) and* 1.65 *years (right).*

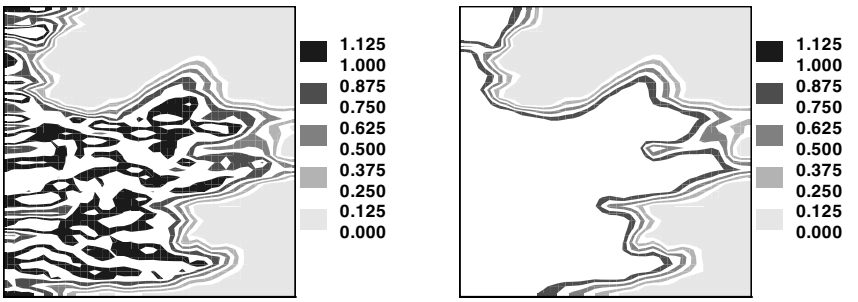


FIG. 8.12. *The linear flood CMM (left) and VCCMM (right) concentration at* 25 *years using time step* 0.18 *years. (The CFL time step is* 0.06 *years.)*

problem is not simply associated with the treatment of wells but also with treatment of trace-back regions throughout the domain. We note that the relative errors along the "stair-step" river bank reduce under grid refinement (as long as the refinement attempts to follow the true, smooth river bank).

8.4. A linear flood. Our fourth example tests the inflow boundary adjustment. We use the permeability field in the second example (see Figure 8.7). However, this time we assume no wells and instead impose a linear pressure drop across the domain in the x-direction (the pressure difference is about a tenth of that in the first example).

The time step is 0.18 years, which is three times the CFL time step. In Figure 8.12, we show results after 140 steps, or about 25 years. The CMM is very disappointing. It exhibits severe overshoots up to a concentration of 1.65 and volume errors exceed 10% in the interior of the domain. However, the VCCMM algorithm (using the ideas in section 5 to approximate the inflow boundary) was able to remove these severe volume errors, and we observe good results with mass and volume being strictly conserved.

In Figure 8.13 we show the corresponding results for the VCCMM-TF (using section 7 to treat the inflow boundary) for time steps 0.18 and 0.36 years (3 and 6 times the CFL step). As can be seen, the inflow is better approximated using VCCMM-TF. We might expect a thinner diffusion layer for the simulation using the larger time step, but evidently the grid is not fine enough to resolve this. We note that for this example, a time step 9 times CFL causes excessive initial relative volume errors inside the domain of around 25%.

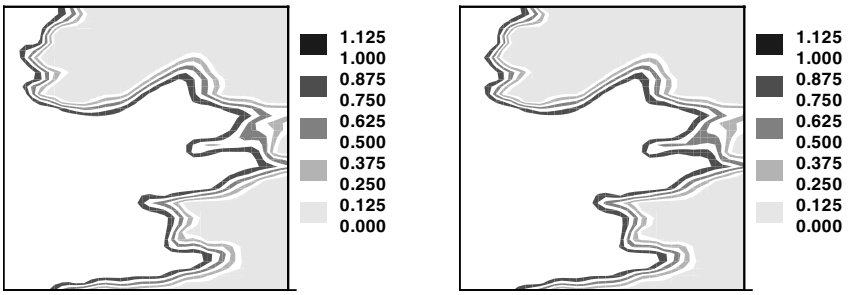


FIG. 8.13. *The linear flood VCCMM-TF concentration at* 25 *years for time steps* 0.18 *years (left) and* 0.36 *years (right). (The CFL time step is* 0.06 *years.)*

8.5. A comparison with an analytic solution. Our fifth and final example is to compare the VCCMM-TF to the results of an analytic solution due to Hsieh [14] involving radial flow out of a well in a horizontally infinite, uniform porous medium (effectively making the problem one dimensional). In radial coordinates, where r is the radial distance from the center line of the recharge well, we have

$$\begin{aligned} &c_t + \frac{A}{r}c_r = d_{\text{long}}\frac{A}{r}c_r r, && r > r_w,\ t > 0,\\ &c(r_w,t) = c_I \quad \text{and} \quad c(+\infty,t) = 0, && t > 0,\\ &c(r,0) = 0, && r > r_w, \end{aligned}$$

where $A = Q/(2\pi b\phi)$, Q is the volumetric rate of tracer injection, b is the vertical thickness of the formation, ϕ is the porosity, d_{long} is the longitudinal dispersivity, r_w is the radius of the injection well, and c_I is the concentration of the injected tracer. In dimensionless form, the equation is

$$G_\tau + \frac{1}{\rho}G_\rho = \frac{1}{\rho}G_{\rho\rho}, \quad \rho > \rho_w,\ \tau > 0,$$

where $G = c/c_I$, $\rho = r/d_{\text{long}}$, $\rho_w = r_w/d_{\text{long}}$, and $\tau = At/d_{\text{long}}^2$. Hsieh gave the analytic solution to this problem.

We mimic the same setting in a 2-D finite domain with the well in the center. The domain is 256 meters square and has a thickness of $b = 1$ m, porosity $\phi = 0.2$, and initial concentration $c = 0$. Fluid with a concentration $c_I = 1$ is injected into the center of the domain at the rate of $Q = 4369.06$ cubic meters per year (which is 1 pore-volume every 3 years). We also set $R = 1$ (i.e., no retardation).

We present two test cases, with either more or less dispersion. For the dispersion tensor (8.4), the less dispersive case has molecular diffusion $d_{\text{mol}} = 0.002$ cm^2/sec and longitudinal dispersion $d_{\text{long}} = 20$ cm. The more dispersive case is five times stronger: $d_{\text{mol}} = 0.01$ cm^2/sec and $d_{\text{long}} = 100$ cm.

In Figure 8.14 we show the comparison in the concentration along the radial direction away from the well. The analytic solution is derived from Hsieh's code [14]. We solve the full 2-D problem using our VCCMM-TF algorithm on a 128×128 grid and a time step of $\Delta t = 0.005$ years, which is 6.82 times the 2-D CFL limit, and report the concentration on the positive x-axis away from the well. On the left is the less diffusive case at time 0.2 years, and on the right is the more diffusive case at 0.1 years. As can be seen, the match is excellent.

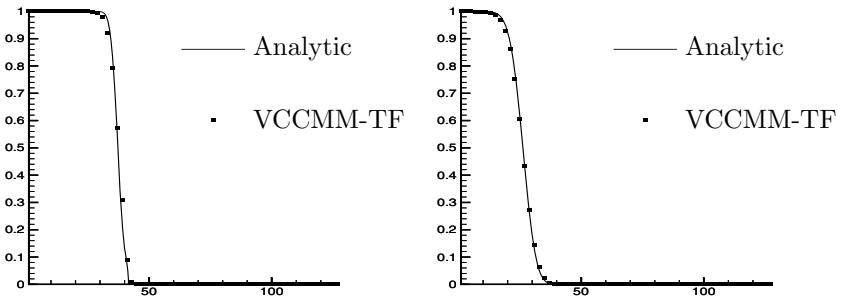


FIG. 8.14. *Comparison between the analytic and the VCCMM-TF concentration along the radial direction away from the well. On the left is the less diffusive case at time* 0.2 *years, and on the right is the more diffusive case at* 0.1 *years.*

9. Conclusions. A critical aspect of approximating hyperbolic transport problems is to conserve the mass of the tracer locally. As we saw in the numerical examples, it is just as critical to the conserve locally the mass of the bulk fluid. We have termed this constraint [illegible]. Incorrect densities, and therefore incorrect reaction dynamics, result from violation of the volume constraint. It can also lead to severe overshoot or undershoot, especially around injection wells, and even to incorrectly predicting the advancement of fronts.

We should remark that one can recover smooth densities by modifying the definition of the trace-back concentration. Simply multiply the concentration obtained for element E by the correction factor $|\tilde{E}|/|E|$. This results in volume conservation, and to the eye, all the plumes appear smooth (i.e., show no overshoot or undershoot) like the VCCMM results. However, this modification destroys mass conservation.

When adjusting trace-back points, care must be used to avoid introducing systematic bias into the transport computation. We presented algorithms that produce good trace-back regions. The key is to adjust the trace-back points "in time," i.e., along the characteristics, where possible. Weaker wells and inflow boundaries can be treated with simple techniques (wells by quadrature and inflow transparently through a space-time "fold-down" strategy). For stronger wells and inflow boundaries, one should use a trace-forwarding strategy.

We do not suggest that ours is the only reasonable adjustment strategy; it is merely one of many that could be implemented. The point of the paper has been to point out how critical this step is, and to present one practical solution to the problem. We saw that our algorithms produce accurate results when compared to Godunov's method and an analytical solution. Our method also allows us to take large time steps such as 3 to 14 times the two-dimensional CFL limited time step, which results in less numerical dispersion than do explicit differencing methods such as Godunov's method.

Trace-back adjustment requires a fair amount of computation. Thus characteristic methods such as ELLAM and CMM are perhaps best suited to steady or near steady flows and long time periods or perhaps when numerical diffusion simply cannot be tolerated. These situations arise, for example, in the modeling of geologic basin formation, long-lived radio-isotope decay, or miscible fingering.

REFERENCES

[1] T. ARBOGAST, *User's Guide to Parssim*1: *The Parallel Subsurface Simulator, Single Phase*, Tech. Rep. TICAM Report 98–13, The Center for Subsurface Modeling, Texas Institute for Computational and Applied Mathematics, The University of Texas at Austin, Austin, TX, May 1998.

[2] T. Arbogast, A. Chilakapati, and M. F. Wheeler, *A characteristic-mixed method for contaminant transport and miscible displacement*, in Computational Methods in Water Resources IX, Vol. 1: Numerical Methods in Water Resources, T. F. Russell et al., eds., Computational Mechanics Publications, Southampton, UK, 1992, pp. 77–84.

[3] T. Arbogast and M. F. Wheeler, *A characteristics-mixed finite element method for advection dominated transport problems*, SIAM J. Numer. Anal., 32 (1995), pp. 404–424.

[4] T. Arbogast, M. F. Wheeler, and I. Yotov, *Mixed finite elements for elliptic problems with tensor coefficients as cell-centered finite differences*, SIAM J. Numer. Anal., 34 (1997), pp. 828–852.

[5] J. B. Bell, C. N. Dawson, and G. R. Shubin, *An unsplit higher-order godunov scheme for scalar conservation laws in two dimensions*, J. Comput. Phys., 74 (1988), pp. 1–24.

[6] M. A. Celia, T. F. Russell, I. Herrera, and R. E. Ewing, *An Eulerian-Lagrangian localized adjoint method for the advection-diffusion equation*, Advances in Water Resources, 13 (1990), pp. 187–206.

[7] A. Chilakapati, *Numerical Simulation of Reactive Flow and Transport through the Subsurface*, Ph.D. thesis, Rice University, Houston, Texas, 1993.

[8] A. Chilakapati, *A characteristic-conservative model for Darcian advection*, Advances in Water Resources, 22 (1999), pp. 597–609.

[9] C. Dawson, *Godunov-mixed methods for advection-diffusion equations in multidimensions*, SIAM J. Numer. Anal., 30 (1993), pp. 1315–1332.

[10] J. Douglas, Jr., C.-S. Huang, and F. Pereira, *The modified method of characteristics with adjusted advection*, Numer. Math., 83 (1999), pp. 353–369.

[11] J. Douglas, Jr., F. Pereira, and L.-M. Yeh, *A locally conservative Eulerian-Lagrangian numerical method and its application to nonlinear transport in porous media*, Comput. Geosci., 4 (2000), pp. 1–40.

[12] J. Douglas, Jr., and T. F. Russell, *Numerical methods for convection-dominated diffusion problems based on combining the method of characteristics with finite element or finite difference procedures*, SIAM J. Numer. Anal., 19 (1982), pp. 871–885.

[13] R. W. Healy and T. F. Russell, *Treatment of internal sources in the finite-volume ELLAM*, in Computational Methods in Water Resources XIII, Vol. 2, L. R. Bentley, J. F. Sykes, C. A. Brebbia, W. G. Gray, and G. F. Pinder, eds., A. A. Balkema, Rotterdam, 2000, pp. 619–622.

[14] P. A. Hsieh, *A new formula for the analytical solution of the radial dispersion problem*, Water Res. Res., 22 (1986), pp. 1597–1605.

[15] R. J. LeVeque, *Numerical Methods for Conservation Laws*, 2nd ed., Birkhäuser, Basel, 1992.

[16] R. A. Raviart and J. M. Thomas, *A mixed finite element method for 2nd order elliptic problems*, in Mathematical Aspects of Finite Element Methods, I. Galligani and E. Magenes, eds., Lecture Notes in Math. 606, Springer-Verlag, New York, 1977, pp. 292–315.

[17] T. F. Russell and M. F. Wheeler, *Finite element and finite difference methods for continuous flows in porous media*, in The Mathematics of Reservoir Simulation, R. E. Ewing, ed., 1 Frontiers Appl. Math., SIAM, Philadelphia, 1983, pp. 35–106.

[18] B. van Leer, *Towards the ultimate conservative difference scheme via second-order sequel to godonov's methods*, J. Comput. Phys., 32 (1979), pp. 101–136.

[19] C. D. White and R. N. Horne, *Computing absolute transmissibility in the presence of fine-scale heterogeneity, Paper SPE* 16011, in Proceedings, Ninth SPE Symposium on Reservoir Simulation, 1987, pp. 209–220.

SIAM J. SCI. COMPUT.
Vol. 28, No. 6, pp. 2023–2053
© 2006 Society for Industrial and Applied Mathematics

ADAPTIVE FINITE ELEMENT METHODS FOR ELLIPTIC PDEs BASED ON CONFORMING CENTROIDAL VORONOI–DELAUNAY TRIANGULATIONS*

LILI JU†, MAX GUNZBURGER‡, AND WEIDONG ZHAO§

Abstract. A new triangular mesh adaptivity algorithm for elliptic PDEs that combines a posteriori error estimation with centroidal Voronoi–Delaunay tessellations of domains in two dimensions is proposed and tested. The ability of the first ingredient to detect local regions of large error and the ability of the second ingredient to generate superior unstructured grids result in a mesh adaptivity algorithm that has several very desirable features, including the following. Errors are very well equidistributed over the triangles; at all levels of refinement, the triangles remain very well shaped, even if the grid size at any particular refinement level, when viewed globally, varies by several orders of magnitude; and the convergence rates achieved are the best obtainable using piecewise linear finite elements. This methodology can be easily extended to higher-order finite element approximations or mixed finite element formulations although only the linear approximation is considered in this paper.

Key words. adaptive methods, finite element methods, centroidal Voronoi tessellation, Delaunay triangulation

AMS subject classifications. 65N50, 65N15

DOI. 10.1137/050643568

1. Introduction.

Adaptive grid generation techniques play an increasingly important role in the numerical solution of partial differential equations (PDEs). An essential ingredient of adaptive meshing techniques is a posteriori error estimators which are quantities that are computable once an approximate solution of the PDE has been determined. The key objectives in designing reliable and efficient a posteriori error estimators and mesh adaptivity techniques are that an existing mesh is refined in such a way that the errors in the approximate solution of the PDE on the new mesh are distributed as uniformly as possible, that those approximate solutions converge, as the mesh size decreases, to the exact solution as well as can be expected, and that the first two objectives are met with a relatively simple complexity. Both mesh adaptivity and a posteriori error estimators have been extensively studied, beginning in the late 1970s [4, 5, 6, 7] and followed by a vast literature. Here, we refer the reader to [2, 11, 31] for references on a posteriori error estimation and mesh adaptivity for elliptic PDEs.

The performance of adaptive methods for PDEs depends not only on the error estimators, but also on the techniques used for adaptively refining and generating meshes. In [14], a convergent adaptive algorithm was proposed for the linear finite element methods applied to the Poisson equation in two dimensions; a sequence of refined triangulations is defined based on an a posteriori error estimator and the

*Received by the editors October 26, 2005; accepted for publication (in revised form) May 19, 2006; published electronically December 5, 2006.
http://www.siam.org/journals/sisc/28-6/64356.html
†Department of Mathematics, University of South Carolina, Columbia, SC 29208 (ju@math.sc.edu). This author's work is partially supported by the University of South Carolina Research and Productive Scholarship under grant RPS 13060-06-12306.
‡School of Computational Science, Florida State University, Tallahassee, FL 32306-4120 (gunzburg@csit.fsu.edu).
§School of Mathematics and System Sciences, Shandong University, Jinan, Shandong 250100, People's Republic of China (wdzhao@math.sdu.edu.cn).

convergence is proved. Another new family of adaptive algorithms was given in [25, 26, 27] and the convergence of the algorithms was also proved.

In many if not most adaptive methods for PDEs, the meshes are refined locally whenever some criterion based on a local error estimator is not satisfied on some elements; the mesh elsewhere in the domain is not changed. However, in an unrefined region, the errors could be so small that, because one has too many grid nodes there, computational resources are wasted. Thus, to achieve some sort of mesh optimality, it is more reasonable to coarsen the meshes in regions where errors are relatively small in addition to refining in regions where the errors are relatively large. For example, in [8], by introducing a coarsening step to the algorithm proposed in [25], an adaptive method is defined that results in certain optimal convergence rates in the energy norm.

In this paper, we propose an adaptive algorithm for linear finite element methods that can distribute the nodes in some optimal way according to a posteriori error estimates, so that the error of the resulting approximate solution is distributed equally over the elements. To some extent, it is close to the mesh smoothing scheme proposed in [3]. We also would like to point out that the techniques described in this paper can be easily extended to higher-order finite element approximations or mixed finite element formulations. The plan of the rest of the paper is as follows. In sections 2 and 3, we respectively discuss the specific a posteriori error estimators and mesh generation and optimization methods that are used to define our mesh adaptivity algorithm. The mesh generation algorithm we use requires the definition of a density function which that algorithm uses to decide how grid points should be distributed. In section 4, we first show how that density function can be related to the a posteriori error estimators and then we provide the description of our mesh adaptation algorithm. In section 5, we use several computational experiments to demonstrate the effectiveness and efficiency of our mesh adaptation approach. Finally, some concluding remarks are given in section 6.

2. Error estimators for linear finite element methods. Let $\Omega \subset \mathbb{R}^2$ be a bounded domain with a Lipschitz boundary $\partial\Omega$. Consider the model elliptic PDE with homogeneous boundary condition

$$\begin{cases} -\nabla \cdot (a\nabla u) = f & \text{in } \Omega, \\ u = 0 & \text{on } \partial\Omega, \end{cases} \tag{2.1}$$

where $f \in L^2(\Omega)$ and $a \in C^1(\Omega)$ with $a(\mathbf{x}) \geq \widetilde{a} > 0$.

There are several types of a posteriori error estimators used in adaptive finite element methods, e.g., explicit error estimators, implicit error estimators, multilevel estimators, and averaging estimators. In this paper, we use only explicit a posteriori error estimators for adaptive mesh generation and refinement because they can be computed directly from the finite element approximate solution and the data of the problem. In the following, we first review some results about explicit a posteriori error estimators in the context of finite element methods for the model problem (2.1).

2.1. Finite element spaces and a priori error estimates. Assume that Ω is a polygonal domain with boundary $\partial\Omega$ and $\mathcal{T}$ is a conforming triangulation of Ω [13]. Denote by h_T the diameter of the triangle $T \in \mathcal{T}$ and by r_T the diameter of the largest circle that can be inscribed in T. Define the regularity ratio of the triangle T by $\kappa_T = h_T/r_T$. If there is a constant κ such that $\kappa_T \leq \kappa$ for all $T \in \mathcal{T}$, then we say that the triangulation $\mathcal{T}$ of Ω is regular. It is worth noting that the assumption of regularity permits partitions of the domain Ω into meshes that may contain elements

of quite different sizes. This observation is very important for adaptive refinement. In the following, we will assume that $\mathcal{T}$ is regular.

Let p denote a nonnegative integer and $\mathbb{P}_p$ the space of polynomials of degree less than or equal to p. The finite element space of degree p associated with the triangulation $\mathcal{T}$ is defined by $V_h = \{v \in C(\overline{\Omega}) \mid v|_T \in \mathbb{P}_p(T) \text{ for all } T \in \mathcal{T}\}$. In this paper, for simplicity, we consider the case $p = 1$; i.e., V_h is the continuous piecewise linear finite element space with respect to $\mathcal{T}$. But the techniques described in the remaining sections can be easily extended to other higher-order approximations.

In the a posteriori error analysis, it is also worthwhile to consider properties of certain patches of elements. Let the patch $\widetilde{T} \in \Omega$ be the union of the triangle T and the other triangles in $\mathcal{T}$ that share at least one common vertex with T. We define

$$h_{\widetilde{T}} = \max_{T' \subset \widetilde{T}} h_{T'} \qquad \text{and} \qquad r_{\widetilde{T}} = \max_{T' \subset \widetilde{T}} r_{T'};$$

then, the regularity of the patch $\widetilde{T}$ is measured by $\kappa_{\widetilde{T}} = h_{\widetilde{T}}/r_{\widetilde{T}}$. It is easy to see that the regularity of the triangulation $\mathcal{T}$ is inherited by each patch $\widetilde{T}$; see, e.g., [2].

Let V be the Hilbert space $H_0^1(\Omega)$. The weak form of problem (2.1) is to find $u \in V$ such that

$$B(u, v) = L(v) \qquad \forall\, v \in V,$$

where B is the bilinear form and L is the linear functional respectively defined by

$$B(u, v) = \int_\Omega a\nabla u \cdot \nabla v \, \mathrm{d}\mathbf{x} \qquad \text{and} \qquad L(v) = \int_\Omega fv \, \mathrm{d}\mathbf{x} \qquad \forall\, u, v \in V.$$

It is clear that $V_h \subset V$. Then, the finite element approximation $u_h \in V_h$ of the problem (2.1) is determined from the problem

$$B(u_h, v_h) = L(v_h) \qquad \forall\, v_h \in V_h.$$

For any $u \in V$, we define its energy norm $\|\cdot\|_E$ by $\|u\|_E = \big(B(u, u)\big)^{1/2}$. We denote by h the piecewise linear function with respect to $\mathcal{T}$ satisfying

$$h(\mathbf{x}) = \max_{T \in \mathcal{T} \text{ and } \mathbf{x} \in \overline{T}} h_T$$

for each vertex $\mathbf{x}$ of $\mathcal{T}$. We also assume that the exact solution $u \in H^2(\Omega)$. Let $e_h = u - u_h$ be the error of the approximate solution u_h; we then have the following classic results about a priori error estimates [13].

THEOREM 1. [illegible] C_1 [illegible] C_2 [illegible] a [illegible] h [illegible]

$$\|e_h\|_E \le C_1 \|\sqrt{a} h^{k-1} |\nabla_k u|\|_{L^2(\Omega)}, \quad k = 1, 2, \tag{2.2}$$

[illegible]

$$\|e_h\|_{L^2(\Omega)} \le C_2 \|\sqrt{a} h^2 |\nabla_2 u|\|_{L^2(\Omega)}. \tag{2.3}$$

2.2. An explicit H^1-type a posteriori error estimator. Let $v \in V$ be chosen arbitrarily; then writing the integral over the whole domain Ω as a sum of integrals over individual triangles gives

$$B(e_h, v) = \sum_{T \in \mathcal{T}} \left\{ \int_T fv \, \mathrm{d}\mathbf{x} - \int_T (a\nabla u_h) \cdot \nabla v \, \mathrm{d}\mathbf{x} \right\}.$$

Let $\mathcal{E}_I$ denote the set of interior edges of $\mathcal{T}$. If T and T' share the common edge $\gamma \in \mathcal{E}_I$, define the jump in the normal flux across the edge γ by

$$[(a\nabla u_h)\cdot \mathbf{n}_\gamma] = (a\nabla u_h)|_T \cdot \mathbf{n}_T + (a\nabla u_h)|_{T'} \cdot \mathbf{n}_{T'},$$

where $\mathbf{n}_T$ is the unit outward normal vector to ∂T. Applying integration by parts and rearranging terms, we then can get

$$B(e_h, v) = \sum_{T\in\mathcal{T}} \int_T rv \,\mathrm{d}\mathbf{x} + \sum_{\gamma\in\mathcal{E}_I} \int_\gamma Rv \,\mathrm{d}s, \tag{2.4}$$

where $r = f + \nabla\cdot(a\nabla u_h)$ and $R = -[(a\nabla u_h)\cdot \mathbf{n}_\gamma]$.

For given $v \in V$, let $I_h v$ be the interpolant of v in V_h. Then, by the orthogonality property $B(e_h, I_h v) = 0$ and (2.4), we have

$$B(e_h, v) = \sum_{T\in\mathcal{T}} \int_T r(v - I_h v) \,\mathrm{d}\mathbf{x} + \sum_{\gamma\in\mathcal{E}_I} \int_\gamma R(v - I_h v) \,\mathrm{d}s \qquad \forall\, v \in V. \tag{2.5}$$

The identity (2.5) plays an important role, indirectly or directly, throughout an a posteriori error analysis of finite element approximations. Due to the coercivity of the bilinear form B on V, the approximation theory, and some norm equivalences, by choosing $v = e_h$, we can obtain the first a posteriori error estimate

$$\begin{aligned} \|e_h\|_E^2 &\le C\bigg\{ \sum_{T\in\mathcal{T}} h_T^2 \|r\|_{L^2(T)}^2 + \sum_{\gamma\in\mathcal{E}_I} h_T \|R\|_{L^2(\gamma)}^2 \bigg\} \\ &= C \sum_{T\in\mathcal{T}} \bigg\{ h_T^2 \|r\|_{L^2(T)}^2 + \frac{1}{2} h_T \|R\|_{L^2(\partial T)}^2 \bigg\}. \end{aligned} \tag{2.6}$$

Except for the constant C, all of the quantities on the right-hand side of (2.6) can be computed explicitly from the finite element solution u_h. Then we obtain an H^1-type local error estimator η_{T,H^1} associated with the element $T \in \mathcal{T}$ defined by

$$\eta_{T,H^1}^2 = h_T^2 \|r\|_{L^2(T)}^2 + \frac{1}{2} h_T \|R\|_{L^2(\partial T)}^2. \tag{2.7}$$

The inequality (2.6) shows that the true error e_h can be bounded from above in terms of the local error estimator η_{T,H^1}; i.e., when η_{T,H^1} is small, the true error e_h must also be small. This property is referred to as the [illegible] of the error estimator η_{T,H^1}. However, we cannot discern anything about the true error e_h on any particular triangle $T \in \mathcal{T}$ from the stability estimate (2.6). Adaptive numerical methods generally also need the fact that the true error e_h is also locally bounded from below by the local error estimator η_{T,H^1}. This type of property is referred to as the [illegible] of the error estimator. By using properly chosen bubble functions, the efficiency of the explicit a posteriori error estimator η_{T,H^1} can also be proved. Details can be found in [2] and the references cited therein. We collect the stability and efficiency results for the error estimator η_{T,H^1} in the following theorem.

THEOREM 2. [illegible] η_{T,H^1} [illegible] (2.7) [illegible] $\eta_{H^1}^2 = \sum_{T\in\mathcal{T}} \eta_{T,H^1}^2$ [illegible] C_1 [illegible] C_2 [illegible] Ω [illegible] a [illegible] $\mathcal{T}$ [illegible]

$$C_1 \bigg\{ \|e_h\|_E^2 + \sum_{T\in\mathcal{T}} h_T^2 \|f - \bar{f}\|_{L^2(T)}^2 \bigg\} \le \|e_h\|_E^2 \le C_2 \eta_{H^1}^2, \tag{2.8}$$

$\bar{f}$ f T T_γ γ

$$\eta_{T,H^1}^2 \le C_2\Big\{\|e_h\|_{E,T_\gamma} + h_T^2\|f-\overline{f}\|_{L^2(T_\gamma)}\Big\}$$

2.3. An explicit L^2-type a posteriori error estimator. Duality arguments can be used to derive L^2-type a posteriori error estimators. The starting point for the application of this technique is the adjoint of the model problem: find $\phi_g \in V$ such that

$$B(v,\phi_g) = (g,v) \qquad \forall v \in V \tag{2.9}$$

with $g \in L^2(\Omega)$ and where $(\cdot,\cdot)$ denotes the $L^2(\Omega)$ inner product. It is assumed that this problem is regular in the sense that the solution $\phi_g \in H^2(\Omega)\cap V$ and there exists a constant C such that

$$\|\phi_g\|_{H^2(\Omega)} \le C\|g\|_{L^2(\Omega)}. \tag{2.10}$$

This assumption is known to hold, in particular, if the domain Ω is convex. The specific choice $g = e_h$ in (2.9) then gives

$$\|e_h\|_{L^2(\Omega)}^2 = B(e_h,\phi_{e_h}).$$

Then, we have

$$\begin{aligned}\|e_h\|_{L^2(\Omega)}^2 \le &\sum_{T\in\mathcal{T}} \|r\|_{L^2(T)}\|\phi_{e_h} - I_h\phi_{e_h}\|_{L^2(T)} \\ &+ \sum_{\gamma\in\mathcal{E}_I} \|R\|_{L^2(\gamma)}\|\phi_{e_h} - I_h\phi_{e_h}\|_{L^2(\gamma)}.\end{aligned} \tag{2.11}$$

By the approximation theory again, combining the inequalities (2.10) with $g = e_h$ and (2.11), we obtain

$$\|e_h\|_{L^2(\Omega)}^2 \le C\sum_{T\in\mathcal{T}}\Big\{h_T^4\|r\|_{L^2(T)}^2 + h_T^3\|R\|_{L^2(\partial T)}^2\Big\},$$

which is similar to (2.6), the only difference being a higher-order scaling in the mesh size; this reflects the expectation of a high-order rate of convergence with respect to the L^2 norm. Let η_{T,L^2} denote the L^2-type local error estimator defined by

$$\eta_{T,L^2}^2 = h_T^4\|r\|_{L^2(T)}^2 + h_T^3\|R\|_{L^2(\partial T)}^2. \tag{2.12}$$

We summarize the results about this local error estimator in the following theorem [2].

THEOREM 3. Ω η_{T,L^2} (2.12) $\eta_{L^2}^2 = \sum_{T\in\mathcal{T}}\eta_{T,L^2}^2$ C Ω a $\mathcal{T}$

$$\|e_h\|_{L^2(\Omega)}^2 \le C\eta_{L^2}^2. \tag{2.13}$$

3. Mesh generation and mesh optimization. There have been many good algorithms developed for mesh generation and mesh optimization; see, e.g., [12, 16, 19, 21, 28, 29]. In this paper, we focus on centroidal Voronoi tessellation based mesh generation as proposed in [15, 16, 19].

3.1. Conforming centroidal Voronoi–Delaunay triangulation. Given an open convex domain $\Omega \in \mathbb{R}^d$ and a set of distinct points $\{\mathbf{x}_i\}_{i=1}^n \subset \Omega$, define for each point $\mathbf{x}_i$, $i = 1, \ldots, n$, the corresponding Voronoi region V_i, $i = 1, \ldots, n$, by

$$V_i = \big\{\mathbf{x} \in \Omega \;\mid\; \|\mathbf{x} - \mathbf{x}_i\| < \|\mathbf{x} - \mathbf{x}_j\| \quad \text{for } j = 1, \ldots, n \quad \text{and} \quad j \neq i\big\}.$$

Clearly, we have $\mathbf{x}_i \in V_i$, $V_i \cap V_j = \emptyset$ for $i \neq j$, and $\cup_{i=1}^n \overline{V}_i = \overline{\Omega}$ so that $\{V_i\}_{i=1}^n$ is a tessellation of Ω. We refer to $\{V_i\}_{i=1}^n$ as the [illegible] of Ω associated with the point set $\{\mathbf{x}_i\}_{i=1}^n$. A point $\mathbf{x}_i$ is called a [illegible] and a subdomain $V_i \subset \Omega$ is referred to as the [illegible] corresponding to the generator $\mathbf{x}_i$.

It is well known that the dual tessellation (in a graph-theoretical sense) to a Voronoi tessellation of Ω is a [illegible]. It is easy to show that the vertices of the Voronoi regions V_i's are the circumcenters of the corresponding Delaunay triangles.

Given a density function $\rho(\mathbf{x})$ defined on Ω, for any region $V \subset \Omega$, we define the [illegible] $\mathbf{x}^*$ of V by

$$\mathbf{x}^* = \frac{\displaystyle\int_V \mathbf{y}\rho(\mathbf{y})\, d\mathbf{y}}{\displaystyle\int_V \rho(\mathbf{y})\, d\mathbf{y}}.$$

DEFINITION 1. [illegible] $\{(\mathbf{x}_i, V_i)\}_{i=1}^n$ [illegible] Ω [illegible] centroidal Voronoi tessellation (CVT) [15] [illegible] $\{\mathbf{x}_i\}_{i=1}^n$ [illegible] $\{V_i\}_{i=1}^n$ [illegible]

$$\mathbf{x}_i = \mathbf{x}_i^* \quad \text{[illegible]} \quad i = 1, \ldots, n.$$

[illegible] centroidal Voronoi–Delaunay triangulation (CVDT).

It is worth noting that a CVT/CVDT may not be unique; see [15]. The extension of CVTs and CVDTs to general surfaces is discussed in [17].

Given any set of points $\{\widetilde{\mathbf{x}}_i\}_{i=1}^n$ on Ω and any tessellation $\{\widetilde{V}_i\}_{i=1}^n$ of Ω, we define the corresponding [illegible] by

$$\mathcal{K}\big(\{(\widetilde{\mathbf{x}}_i, \widetilde{V}_i)\}_{i=1}^n\big) = \sum_{i=1}^n \int_{\widetilde{V}_i} \rho(\mathbf{y})\|\mathbf{y} - \widetilde{\mathbf{x}}_i\|^2\, d\mathbf{y}\,.$$

It has been shown that $\mathcal{K}$ is minimized only if $\{(\widetilde{\mathbf{x}}_i, \widetilde{V}_i)\}_{i=1}^n$ forms a CVT [15]. Although $\mathcal{K}$ may not be directly identified with an energy of some physical system, it is often naturally associated with quantities such as error distortion, variance, and cost in many practical applications.

An important and very useful property of CVTs is that the energy is equally distributed over the Voronoi regions V_i's in an asymptotic way. For example, it was shown in [15] that, in the one-dimensional case,

$$\mathcal{K}_{V_i} \approx \mathcal{K}/n \qquad \text{for } i = 1, \ldots, n,$$

where $\mathcal{K}_{V_i} = \int_{V_i} \rho(\mathbf{x})\|\mathbf{x} - \mathbf{x}_i\|^2 \, d\mathbf{x}$ and $\mathcal{K} = \sum_{i=1}^{n} \mathcal{K}_{V_i}$. For higher-dimensional cases, this property is only a conjecture but its validity has been verified through extensive numerical studies and is widely assumed in practical applications such as vector quantization. As a consequence of this equipartition property, CVTs have important geometric features, including the following.

- For a constant density function, the generators $\{\mathbf{x}_i\}_{i=1}^n$ are uniformly distributed; the Voronoi regions $\{V_i\}_{i=1}^n$ are all almost the same size, and, in the two-dimensional case, most of them are (nearly) congruent convex hexagons [15].
- For a nonconstant density function, the generators $\{\mathbf{x}_i\}_{i=1}^n$ are still locally uniformly distributed, and it is conjectured [15] that, asymptotically, for some constant C,

$$\mathcal{K}_{V_i} = C\rho(\mathbf{x}_i)h_{V_i}^{d+2} \qquad \text{and} \qquad \frac{h_{V_i}}{h_{V_j}} \approx \left(\frac{\rho(\mathbf{x}_j)}{\rho(\mathbf{x}_i)}\right)^{\frac{1}{d+2}}, \tag{3.1}$$

  where h_{V_i} denotes the diameter of V_i and d is the dimension of Ω.

Thus, in principle, one could control the distribution of generators to obtain an equal distribution of the error by connecting the density function $\rho(\mathbf{x})$ to an a posteriori error estimator.

An often used algorithm for constructing CVT/CVDT is the Lloyd method [15].

ALGORITHM 1 (Lloyd's method for CVT). [illegible] Ω [illegible] $\rho(\mathbf{x})$ [illegible] Ω [illegible] n

0. [illegible] n [illegible] $\{\mathbf{x}_i\}_{i=1}^n$ [illegible] Ω.
1. [illegible] $\{V_i\}_{i=1}^n$ [illegible] Ω [illegible] $\{\mathbf{x}_i\}_{i=1}^n$.
2. [illegible] $\{V_i\}_{i=1}^n$ [illegible] $\{\mathbf{x}_i\}_{i=1}^n$.
3. [illegible] $\{(\mathbf{x}_i, V_i)\}_{i=1}^n$ [illegible] 1.

An important property of Lloyd's algorithm is that the energy $\mathcal{K}$ of the Voronoi tessellation $\{(\mathbf{x}_i, V_i)\}_{i=1}^n$ decreases after each iteration [15]. A probabilistic version of Lloyd's method and its parallel implementation were suggested in [24].

If a CVDT mesh is to be used within a discretization method for a PDE, e.g., in a finite element method, some modifications are needed. An obvious one is that the CVDT mesh must conform with the boundary of the domain Ω; i.e., some of the CVDT nodes should be constrained to lie on the boundary so that the boundary conditions of the PDE problem can be enforced.[1]

One can, of course, predefine a set of boundary mesh points and then determine an interior mesh that in some sense "conforms" with the boundary mesh. We choose to instead amend the CVT definition and construction algorithm so that the boundary mesh points are automatically selected in conjunction with the interior mesh points. This results in a better "fit" of the boundary and interior meshes.

First, we generalize the CVT definition. Assume that Ω is compact and the domain boundary $\partial\Omega$ is piecewise smooth; the set of singular points, e.g., corners, is denoted by $P_S = \{\mathbf{z}_i\}_{i=1}^k$. Denote by $\mathbf{Proj}(\mathbf{x})$ the process that projects $\mathbf{x} \in \Omega$ to the closest point to $\mathbf{x}$ on the boundary $\partial\Omega$. Let

$$P_I = \{\mathbf{x}_i \mid \overline{V}_i \cap \partial\Omega = \emptyset\} \qquad \text{and} \qquad P_B = \{\mathbf{x}_i \mid \overline{V}_i \cap \partial\Omega \neq \emptyset\}$$

[1] If $\partial\Omega = \emptyset$, e.g., if Ω is the *surface* of a sphere, Voronoi-based discretizations of PDEs have been discussed in [18].

so that P_I, the set of [illegible], denotes the set of generators that have Voronoi regions that do not intersect the boundary and P_B, the set of [illegible], denotes the set of generators that have Voronoi regions that do intersect the boundary.

DEFINITION 2. [illegible] $\{(\mathbf{x}_i, V_i)\}_{i=1}^n$ [illegible] Ω [illegible] conforming centroidal Voronoi tessellation (CfCVT) [illegible]

- $P_S \subset \{\mathbf{x}_i\}_{i=1}^n$.
- $\mathbf{x}_i = \mathbf{x}_i^*$ *for* $\mathbf{x}_i \in P_I$.
- $\mathbf{x}_i = \mathbf{Proj}(\mathbf{x}_i^*)$ *for* $\mathbf{x}_i \in P_B - P_S$

[illegible] conforming centroidal Voronoi–Delaunay triangulation (CfCVDT).

It is noted that the meaning of singular (corner) points is trivial in two-dimensional space but may need to be more rigorously defined in spaces higher than two dimensions.

An algorithm for constructing a CfCVT/CfCVDT was given in [16, 19] and can be described as follows.[2] We follow Algorithm 1 except that in step 2 the new set of generators is given by

- the centers of mass of the interior Voronoi regions;
- the projections onto the boundary of the centers of mass of the boundary Voronoi regions except if the boundary Voronoi region contains a point in P_S, in which case the new generator is that point.

For this approach, both the number of mesh points on the boundary and their location are not predetermined. However, it is not difficult to show that the number of generators lying on the boundary will never decrease after the first iteration of Lloyd's method. The reason for this is that the nodes on the boundary cannot return to the interior of the domain since their Voronoi regions are obviously always boundary Voronoi regions. Thus, for this approach, the initial position of generators must be chosen well according to the density function ρ; for example, one could determine an ordinary CVT (with no points lying on the boundary) to use as an initial set of generators for the CfCVT construction algorithm.

In practical applications, the domain Ω is often nonconvex and is possibly very complicated [19], so that a main difficulty associated with Lloyd's method for constructing CfCVDTs is the construction of the Voronoi regions. For this reason, we next propose an algorithm for constructing approximate CfCVDTs in two dimensions that does not require the construction of exact Voronoi tessellations.

3.2. Approximate CfCVDT construction. In this section, we propose an algorithm to construct approximate CfCVDTs; we will later use this algorithm within our adaptive methods for mesh generation and optimization. We describe our approach for the two-dimensional case in detail; the generalization to higher dimensions follows similar lines.

Currently, for mesh generation with conforming boundary requirements, [illegible] (CDTs) have been widely used; see, e.g., [29, 30]. The main difference between CDTs and standard Delaunay triangulations is that some geometric constraints such as predetermined node position and node connectivities are added and strictly enforced during the CDT process. For example, the boundary of the domain can be triangulated first, and the resulting boundary triangulation is

[2] Other modified techniques for constructing CfCVTs are given in [16].

then used as a constraint on the conforming triangulation of the whole domain using a CDT. It is worth noting that the dual tessellations of the CDT generally is not an exact Voronoi tessellation, especially near the boundary.

Our algorithm for constructing approximate CfCVDTs is based on the CDT process. Assume that $\Omega \in \mathbb{R}^2$ is a domain with a polygonal boundary.[3] Denote by $P_S = \{\mathbf{z}\}_{i=1}^k$ its corner vertex set as before. An initial conforming triangulation $\mathcal{T}_0 = \{T_i\}_{i=1}^m$ of Ω is generated using the "TRIANGLE" software package [29] that uses the CDT process with a boundary mesh as a constraint and interior Delaunay refinement techniques, or by some other means. Denote by $P = \{\mathbf{x}_i\}_{i=1}^n$ the set of vertices of $\mathcal{T}_0$, by P_B the set of boundary vertices, and by P_I the set of interior vertices. The CDT process guarantees that $P_S \subset P_B$.

For each triangle $T_i = (\mathbf{x}_{i_1}, \mathbf{x}_{i_2}, \mathbf{x}_{i_3}) \in \mathcal{T}_0$, we define

$$\mathbf{x}_{T_i} = \begin{cases} \text{circumcenter of } T_i & \text{if } T_i \text{ is an acute triangle;} \\ \text{the middle point of the longest edge of } T_i & \text{otherwise.} \end{cases}$$

Clearly, $\mathbf{x}_{T_i} \in \overline{T}_i$. For each vertex $\mathbf{x}_i$, we denote by $\{T_{i_k}\}_{k=1}^{m_i} \subset \mathcal{T}_0$ the set of triangles for which $\mathbf{x}_i$ is a vertex, counting in the counterclockwise direction.

First, consider the case $\mathbf{x}_i \in P_I$, i.e., $\mathbf{x}_i$ is an interior vertex. Define U_i by

$$U_i = \text{the polygon formed by } \{\mathbf{x}_{T_{i_k}}\}_{k=1}^{m_i};$$

see Figure 3.1. The polygon U_i can be regarded as an approximation to the Voronoi region V_i associated with $\mathbf{x}_i$. Let $\overline{\mathbf{x}}_i$ denote the center of mass of the U_i with respect to the density function ρ. Denote by $\{\alpha_{i_k}\}_{k=1}^{m_i}$ the associated angles around $\mathbf{x}_i$ corresponding to $\{T_{i_k}\}_{k=1}^{m_i}$. Define

$$\alpha = \begin{cases} \max\{\alpha_{i_k} \mid \overline{T}_{i_k} \cap \partial\Omega \neq \emptyset\} & \text{if } \overline{T}_{i_k} \cap \partial\Omega \neq \emptyset \text{ for some } i_k; \\ 0 & \text{otherwise,} \end{cases}$$

and let $\mathbf{e}_i$ denote the corresponding boundary edge opposite to the angle α_{i_k} such that $\alpha_{i_k} = \alpha$; see Figure 3.1 for illustrations of some cases.

Now, select a parameter θ_{max} ($\pi > \theta_{max} > \pi/2$). Then, define

$$\mathbf{y}_i = \begin{cases} \overline{\mathbf{x}}_i & \text{if } \alpha < \theta_{max}; \\ \mathbf{Proj}_{\mathbf{e}_i}\mathbf{x}_i & \text{otherwise,} \end{cases} \tag{3.2}$$

where $\mathbf{Proj}_{\mathbf{e}_i}\mathbf{x}_i$ denotes the projection of $\mathbf{x}_i$ onto the boundary edge $\mathbf{e}_i$. It is clear that $\mathbf{y}_i$ is still an interior vertex if $\alpha < \theta_{max}$; otherwise, it is a boundary vertex although $\mathbf{x}_i$ is an interior node.

Next, consider the case $\mathbf{x}_i \in P_B$, i.e., $\mathbf{x}_i$ is a boundary vertex. Let $\mathbf{e}_1$ and $\mathbf{e}_2$ denote the two boundary edges having $\mathbf{x}_i$ as the common endpoint, and let $\mathbf{z}_1$ and $\mathbf{z}_2$ denote the midpoints of $\mathbf{e}_1$ and $\mathbf{e}_2$, respectively; see Figure 3.2. The approximate Voronoi region U_i of $\mathbf{x}_i$ is defined by

$$U_i = \text{the polygon formed by } \mathbf{z}_1, \{\mathbf{x}_{T_{i_k}}\}_{k=1}^{m_i}, \text{ and } \mathbf{z}_2;$$

see Figure 3.2. Let $\overline{\mathbf{x}}_i$ denote the center of mass of the U_i associated with the density function ρ.

[3] For domains with curved boundaries, the projection process **Proj** can be easily effected by a damped Newton method; see [28].

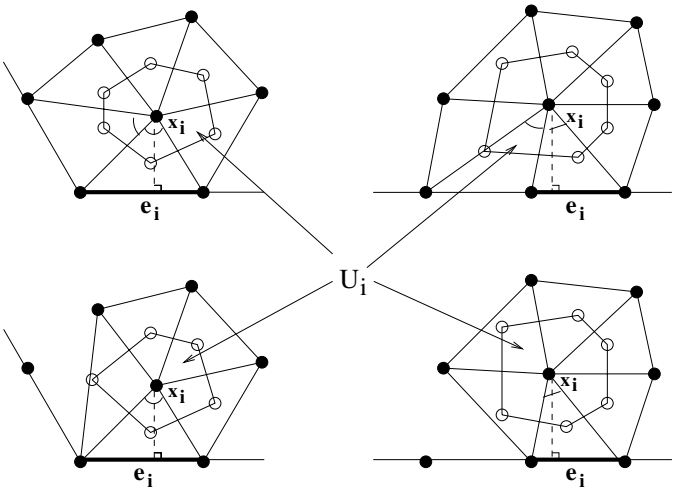


FIG. 3.1. *The approximate Voronoi region U_i for the interior vertex $\mathbf{x}_i$.*

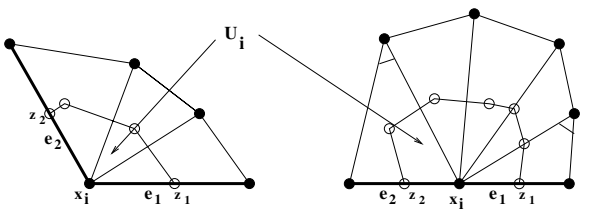


FIG. 3.2. *The approximate Voronoi region U_i for the boundary vertex $\mathbf{x}_i$. Left: Corner vertex. Right: Noncorner vertex.*

If $\mathbf{x}_i \in P_B - P_S$, denote by β_1 and β_2 the angles facing the boundary edges $\mathbf{e}_1$ and $\mathbf{e}_2$, respectively, in $\{T_{i_k}\}_{k=1}^{m_i}$; see Figure 3.2 (right). Let

$$\beta = \max(\beta_1, \beta_2)$$

and select a parameter θ_{min} ($\pi/3 > \theta_{min} > 0$). Then, define

$$\mathbf{y}_i = \begin{cases} \mathbf{x}_i & \text{if } \mathbf{x}_i \in P_S; \\ \mathbf{Proj}_{\overline{\mathbf{z}_1\mathbf{z}_2}}\overline{\mathbf{x}}_i & \text{if } \mathbf{x}_i \in P_B - P_S \text{ and } \beta > \theta_{min}; \\ \overline{\mathbf{x}}_i & \text{if } \mathbf{x}_i \in P_B - P_S \text{ and } \beta \le \theta_{min}, \end{cases} \tag{3.3}$$

where $\mathbf{Proj}_{\overline{\mathbf{z}_1\mathbf{z}_2}}\mathbf{x}_i$ denotes the projection of $\mathbf{x}_i$ onto the segment $\overline{\mathbf{z}_1\mathbf{z}_2}$. It is clear that $\mathbf{y}_i$ is also a boundary vertex if $\mathbf{x}_i$ is a corner vertex, or $\mathbf{x}_i$ is a noncorner vertex but $\beta > \theta_{min}$; otherwise, $\mathbf{y}_i$ becomes an interior vertex although $\mathbf{x}_i$ is on the boundary (We also call it a [illegible].)

[illegible] We can now describe an algorithm for constructing an approximate CfCVDT of the domain Ω.

ALGORITHM 2 (modified Lloyd method for approximate CfCVDT). [illegible] Ω [illegible] $\rho(\mathbf{x})$ [illegible] Ω [illegible] $\mathcal{T}_0$ [illegible] Ω [illegible] $\{\mathbf{x}_i\}_{i=1}^n$ [illegible]

1. [illegible] $\{\mathbf{y}_i\}_{i=1}^n$ [illegible] $\{\mathbf{x}_i\}_{i=1}^n$ [illegible] (3.2) [illegible] (3.3).
2. [illegible] $\{\mathbf{x}_i\}_{i=1}^n = \{\mathbf{y}_i\}_{i=1}^n$ [illegible] $\mathcal{E}_B$ [illegible] $\{\mathbf{x}_i\}_{i=1}^n$.
3. [illegible] Ω [illegible] $\{\mathbf{x}_i\}_{i=1}^n$ [illegible] $\mathcal{E}_B$ [illegible] $\mathcal{T}$.
4. [illegible] $\mathcal{T}$ [illegible] $\mathcal{T}$ [illegible] 1.

In the remainder of this paper, we will use the notation $\mathcal{T}$=CfCVDT($\mathcal{T}_0$,Ω,ρ) to represent the output of Algorithm 2.

Remark 1. To prevent some vertices from frequently jumping back and forth between the boundary and the interior of the domain, more sophisticated controls are needed; for the sake of simplicity, we omit some details in Algorithm 2.

Remark 2. Two user-defined parameters θ_{max} and θ_{min} corresponding respectively to the projection process and the lifting process are used to avoid badly shaped triangles in the region close to the boundary. In our computational experiments, we set $\theta_{max} = 5\pi/9$ and $\theta_{min} = \pi/6$. These are only empirical values, but many experiments lead us to believe they are good choices.

4. CfCVDT-based adaptive finite element methods. Adaptive meshing methods for solving PDEs often takes the following standard form:

0. generate a coarse mesh $\mathcal{T}^{(0)}$ of the domain Ω and set $\ell = 0$;
1. solve the system produced by discretizing the PDE based on $\mathcal{T}^{(\ell)}$ and calculate the local error estimators;
2. if some convergence criteria is satisfied, terminate; otherwise, go to step 3;
3. refine the mesh $\mathcal{T}^{(\ell)}$ based on the local error estimators to get the next level of mesh $\mathcal{T}^{(\ell+1)}$ and set $\ell = \ell + 1$; then go to step 1.

In our adaptive method, we use CfCVDTs to refine and optimize the mesh at each level, but first we need to determine, from the error estimators, the density function used in the CfCVDT algorithm.

4.1. Determination of the density function. Let $\mathcal{T}^{(\ell)}$ denote the triangulation of Ω with vertices $\{\mathbf{x}_i^{(\ell)}\}_{i=1}^{n^{(\ell)}}$ at the refinement level ℓ. Let $\eta_{T,H^1}^{(\ell)}$ and $\eta_{T,L^2}^{(\ell)}$ represent the corresponding local H^1-type and L^2-type error estimators on $T \in \mathcal{T}^{(\ell)}$ at level ℓ defined by (2.7) and (2.12), respectively. A comparison of (2.2) and (2.8) and of (2.3) and (2.13) reveals that it is reasonable to divide both $\eta_{T,H^1}^{(\ell)}$ and $\eta_{T,L^2}^{(\ell)}$ by $\sqrt{a}$ in order to reflect the local variations of true error more accurately. Thus, we define

$$(\xi_{T,H^1}^{(\ell)})^2 = \frac{(\eta_{T,H^1}^{(\ell)})^2}{a_T} \qquad \text{and} \qquad (\xi_{T,L^2}^{(\ell)})^2 = \frac{(\eta_{T,L^2}^{(\ell)})^2}{a_T}, \tag{4.1}$$

where a_T is the mean value of $a(\mathbf{x})$ on the triangle T, i.e., $a_T = \int_T a(\mathbf{x})\,d\mathbf{x}/\text{Area}(T)$. In order to minimize

$$(\xi_{H^1}^{(\ell)})^2 = \sum_{T\in\mathcal{T}^{(\ell)}} (\xi_{T,H^1}^{(\ell)})^2 \qquad \text{or} \qquad (\xi_{L^2}^{(\ell)})^2 = \sum_{T\in\mathcal{T}^{(\ell)}} (\xi_{T,H^1}^{(\ell)})^2,$$

we need to distribute $(\xi_{T,H^1}^{(\ell)})^2$ or $(\xi_{T,L^2}^{(\ell)})^2$ equally over all triangles of $\mathcal{T}^{(\ell)}$.

Set

$$\widetilde{\rho}_{T,H^1}^{(\ell)} = \frac{(\xi_{T,H^1}^{(\ell)})^2}{h_T^4} \qquad \text{and} \qquad \widetilde{\rho}_{T,L^2}^{(\ell)} = \frac{(\xi_{T,L^2}^{(\ell)})^{4/3}}{h_T^4}. \tag{4.2}$$

We then uniquely determine two piecewise linear functions (with respect to $\mathcal{T}^{(\ell)}$) $\rho_{H^1}^{(\ell+1)}$ and $\rho_{L^2}^{(\ell+1)}$ on Ω such that for any vertex $\mathbf{x}_i^{(\ell)}$ of $\mathcal{T}^{(\ell)}$,

$$\rho_{H^1}^{(\ell+1)}(\mathbf{x}_i^{(\ell)}) = \frac{\sum\limits_{T\in S_i} \widetilde{\rho}_{T,H^1}^{(\ell)}}{\mathrm{card}(S_i)} \qquad \text{and} \qquad \rho_{L^2}^{(\ell+1)}(\mathbf{x}_i^{(\ell)}) = \frac{\sum\limits_{T\in S_i} \widetilde{\rho}_{T,L^2}^{(\ell)}}{\mathrm{card}(S_i)}, \tag{4.3}$$

where

$$S_i = \{T \in \mathcal{T}^{(\ell)} \mid \mathbf{x}_i^{(\ell)} \in \overline{T}\}.$$

Note that, in some sense, if the solution $u \in H^2(\Omega)$, we have

$$(\xi_{T,H^1}^{(\ell)})^2 \approx a_T |\nabla_2 u|^2 h_T^4 \qquad \text{and} \qquad (\xi_{T,L^2}^{(\ell)})^2 \approx a_T |\nabla_2 u|^2 h_T^6. \tag{4.4}$$

Combining (4.4) with the CVT/CVDT property (3.1) for $d = 2$, i.e.,

$$\frac{h_{V_i}}{h_{V_j}} \approx \Big(\frac{\rho(\mathbf{x}_j)}{\rho(\mathbf{x}_i)}\Big)^{1/4}$$

for a CVT $\{(\mathbf{x}_i, V_i)\}_{i=1}^n$ of Ω with respect to the density function ρ, it is then not difficult to verify that the CfCVDT mesh $\mathcal{T}^{(\ell+1)}$ generated by the density function $\rho_{H^1}^{(\ell+1)}$ or $\rho_{L^2}^{(\ell+1)}$ will approximately have the property that

$$(\xi_{T_i,H^1}^{(\ell+1)})^2 \approx (\xi_{T_j,H^1}^{(\ell+1)})^2 \qquad \text{or} \qquad (\xi_{T_i,L^2}^{(\ell+1)})^2 \approx (\xi_{T_j,L^2}^{(\ell+1)})^2,$$

respectively, for any triangles $T_i, T_j \in \mathcal{T}^{(\ell+1)}$.

We will refer to the density functions $\rho_{H^1}^{(\ell+1)}$ and $\rho_{L^2}^{(\ell+1)}$ as the H^1-based and L^2-based density functions, respectively. From their defining formulas, it is easy to see that $\rho_{H^1}^{(\ell+1)}$ varies more rapidly than does $\rho_{L^2}^{(\ell+1)}$. We expect that CfCVDT meshes generated using $\rho_{H^1}^{(\ell+1)}$ will produce a finite element approximation with smaller H^1 norm or energy error while those generated using $\rho_{L^2}^{(\ell+1)}$ will tend to have smaller L^2 norm error.

The most time consuming step in the calculations of $\rho_{H^1}^{(\ell+1)}(\mathbf{x})$ and $\rho_{L^2}^{(\ell+1)}(\mathbf{x})$ for any $\mathbf{x} \in \Omega$ is the nearest neighbor search operation since they are defined by interpolation with respect to an unstructured mesh. However, this task can be effected efficiently using the software package "ANN" [1] that is based on the K-D tree algorithm.

Remark 3. In many practical applications, the coefficient in the model equation (2.1) is often a tensor product, i.e., a symmetric, positive definite matrix

$$A(\mathbf{x}) = \begin{pmatrix} a_{11}(\mathbf{x}) & a_{12}(\mathbf{x}) \\ a_{21}(\mathbf{x}) & a_{22}(\mathbf{x}) \end{pmatrix}$$

rather than a scalar-valued function $a(\mathbf{x})$. Under this situation, if the difference between $a_{11}(\mathbf{x})$ and $a_{22}(\mathbf{x})$ is not large locally, then it is still reasonable to scale these estimators by

$$(\xi_{T,H^1}^{(\ell)})^2 = \frac{(\eta_{T,H^1}^{(\ell)})^2}{A_T} \qquad \text{and} \qquad (\xi_{T,L^2}^{(\ell)})^2 = \frac{(\eta_{T,L^2}^{(\ell)})^2}{A_T}, \tag{4.5}$$

where $A_T = \int_T \sqrt{\det(A(\mathbf{x}))}\, d\mathbf{x}/\mathrm{Area}(T)$. If $A(\mathbf{x})$ is strongly anisotropic, then it could not be handled correctly in this framework; the anisotropic CVT meshes proposed in [20], perhaps with some variations, may be able to deal with this case.

. . . . 4. If a higher-order finite element approximation is used, for example, $V_h = \{v \in C(\overline{\Omega}) \mid v|_T \in \mathbb{P}_k(T) \text{ for all } T \in \mathcal{T}\}$, where $k > 1$, then the discrete solution u_h is of kth order convergence in the H^1 norm and $(k+1)$th order in the L^2 norm. Let $\eta^{(\ell)}_{T,H^1}$ and $\eta^{(\ell)}_{T,L^2}$ be the accordingly derived local H^1- and L^2-type error estimators for this approximation; then by similar analysis, it is better to replace (4.2) by

$$\widetilde{\rho}^{(\ell)}_{T,H^1} = \frac{(\xi^{(\ell)}_{T,H^1})^{\frac{4}{k+1}}}{h_T^4} \qquad \text{and} \qquad \widetilde{\rho}^{(\ell)}_{T,L^2} = \frac{(\xi^{(\ell)}_{T,L^2})^{\frac{4}{k+2}}}{h_T^4}. \tag{4.6}$$

. . . . 5. If the mixed finite element method is used, the density function can be determined similarly to the standard finite element approximation since all we need are just explicit a posteriori local error estimators and their orders with respect to the local mesh size h_T.

4.2. Adaptive algorithms based on CfCVDT. Let $\{E_i\}_{i=1}^{k^{(\ell)}}$ denote the set of edges of the ℓth level triangulation $\mathcal{T}^{(\ell)}$. Set $\rho_{E_i} = \rho(\mathbf{z}_i)$ for any density function ρ, where $\mathbf{z}_i$ denotes the midpoint of the edge E_i. We can now define our adaptive finite element method as follows.

Algorithm 3 (CfCVDT-based adaptive finite element method). . . . . Ω . . . . . $N_{max} > 0$ . . . . .

The parameter θ in Algorithm 3 is used here to control the refinement process [26]. The sorting procedure in step 4 can be implemented efficiently using a quick sorting algorithm.

5. Computational experiments. In this section, using computational experiments, we illustrate the effectiveness of CfCVDT-based adaptive finite element methods. Consider the problem

$$\begin{cases} -\nabla \cdot (a\nabla u) + bu = f & \text{in } \Omega, \\ u = g & \text{on } \partial\Omega, \end{cases} \tag{5.1}$$

where $b \in L^\infty(\Omega)$ with $b \geq 0$. Correspondingly, $r = f$ in (2.7) and (2.12) is changed to $r = f - bu_h$. Initial coarse meshes are either chosen to be a uniform Cartesian grid or are produced using the "TRIANGLE" package and are subsequently repeatedly refined by our adaptive methods, i.e., Algorithm 3. For the sake of having something to compare to, we also find finite element approximations of the problem (5.1) using uniform refinements of the initial meshes. We set $L_{max} = 15$, $N_{max} = 20{,}000$, and $\theta = 0.4$ for Algorithm 3. For our adaptive methods, the convergence rate CR with respect to the norm $\|\cdot\|$ at the refinement level ℓ is roughly computed by

$$\text{CR} = \frac{2\log(\|e_{h,\ell}\|/\|e_{h,\ell-1}\|)}{\log(n_{\ell-1}/n_\ell)}, \tag{5.2}$$

where n_ℓ denotes the number of nodes and $e_{h,\ell}$ denotes the error $u - u_h$ at the refinement level ℓ.

We apply the commonly used q measure [22] to evaluate the quality of triangular meshes, where, for any triangle T, q is defined to be twice the ratio of the radius R_T of the largest inscribed circle and the radius r_T of the smallest circumscribed circle, i.e.,

$$q(T) = 2\frac{R_T}{r_T} = \frac{(b+c-a)(c+a-b)+(a+b-c)}{abc},$$

where a, b, and c are side lengths of T. For a given triangulation $\mathcal{T}$, we define

$$q_{min} = \min_{T\in\mathcal{T}} q(T) \qquad \text{and} \qquad q_{avg} = \frac{1}{\text{card}(\mathcal{T})}\sum_{T\in\mathcal{T}} q(T); \tag{5.3}$$

q_{min} measures the quality of the worst triangle and q_{avg} measures the average quality of the mesh $\mathcal{T}$.

5.1. Smooth solution with large gradients. The first illustrative problem is given as follows.

1. Set $\Omega = [-1,1]\times[-1,1]$, $a(x,y) = 10.0\cos(y)$, and $b(x,y) = x^2 + y^2$. The exact solution u is chosen to be

$$u(x,y) = \frac{1.0}{(x-0.5)^2+(y-0.5)^2+0.01} - \frac{1.0}{(x+0.5)^2+(y+0.5)^2+0.01} \tag{5.4}$$

and f and g are determined from u so that (5.1) is satisfied.

It is easy to see that the exact solution u given in (5.4) is a smooth function, i.e., certainly, $u \in C^2(\Omega)$. Note that u achieves its maximum value $99\frac{101}{201}$ at the point $(0.5, 0.5)$ and its minimum value $-99\frac{101}{201}$ at the point $(-0.5, -0.5)$ but decays very

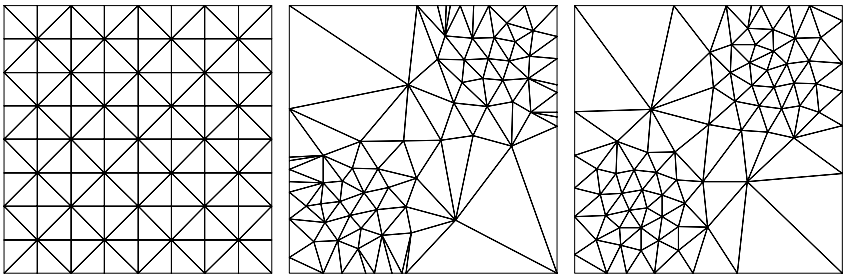


FIG. 5.1. *Initial meshes for Example* 1. *Left: A uniform Cartesian grid with* 81 *nodes. Middle and right: The corresponding CfCVDT meshes with the same number of nodes generated using the density functions* ρ_{H^1} *and* ρ_{L^2}, *respectively.*

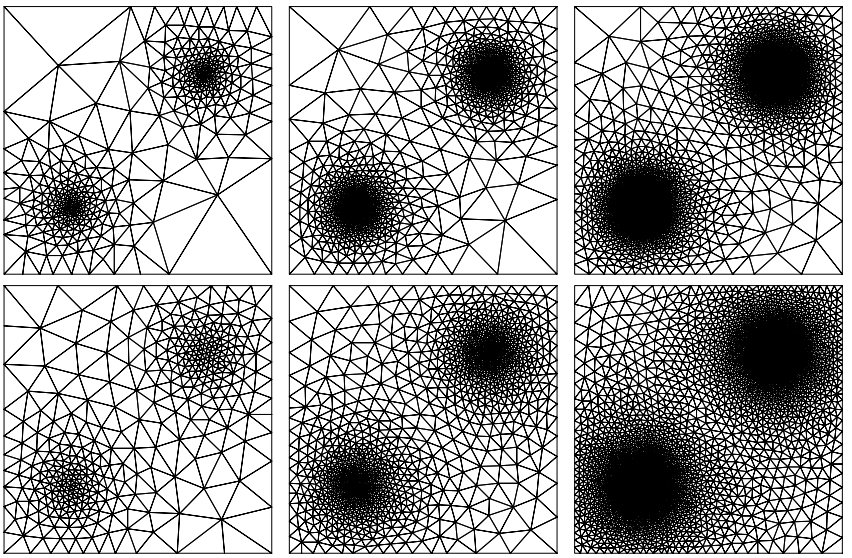


FIG. 5.2. *Repeatedly refined adaptive meshes at some levels generated by the CfCVDT-based adaptive method for Example* 1. *Top:* 394, 1287, 4644 *nodes using the density function* ρ_{H^1}. *Bottom:* 370, 1280, 4629 *nodes using the density function* ρ_{L^2}.

quickly away from its extrema and thus has large gradients near these two points. Note also that $a(x, y)$ and $f(x, y)$ also have relatively rapid variations over Ω.

The initial coarse mesh (the input for step 0 of Algorithm 3) used for the solution of Example 1 is a uniform Cartesian grid consisting of 81 nodes; see Figure 5.1. The corresponding CfCVDT meshes (the output of step 0) with the same number of nodes produced using the density functions ρ_{H^1} and ρ_{L^2} are also given in Figure 5.1. Figure 5.2 presents repeatedly refined meshes at some levels generated using Algorithm 3. The distributions of nodes in the CfCVDT-based adaptive meshes clearly show the accumulation of nodes in the vicinity of the two points near which large gradients in the solution occur. The CfCVDT-based adaptive meshes are "optimal" in the sense that all triangles remain well shaped at all refinement levels, an observation which is supported by the values of q_{min} and q_{avg} given in Table 5.1. It is also clear that the CfCVDT meshes generated using the density function ρ_{H^1} tend to distribute the nodes in a slightly less uniform manner than those generated using the density

TABLE 5.1

Mesh quality, solution errors, and convergence rates for different refinement strategies for Example 1.

ℓ	n_ℓ	q_{min}	q_{avg}	h_{max}/h_{min}	$\|e_h\|_{L^2(\Omega)}$	CR	$\|e_h\|_{H^1(\Omega)}$	CR
Uniform refinement								
0	81	0.828	0.828	1.0	2.3260e+01		1.8187e+02	
1	289	0.828	0.828	1.0	2.2931e+00	3.34	1.0422e+02	0.80
2	1089	0.828	0.828	1.0	9.6260e-01	1.25	6.5176e+01	0.68
3	4225	0.828	0.828	1.0	2.7745e-01	1.80	3.5677e+01	0.87
4	16441	0.828	0.828	1.0	7.1278e-02	1.96	1.8242e+01	0.97
5	66409	0.828	0.828	1.0	1.8071e-02	1.98	9.2255e+00	0.98
Adaptive refinement using ρ_{H^1}								
0	81	0.280	0.805	6.6	7.6651e+00		1.5021e+02	
1	114	0.541	0.916	8.8	2.9722e+00	5.54	9.0709e+01	2.95
2	143	0.459	0.876	20.8	2.2858e+00	2.31	6.1156e+01	3.47
3	229	0.579	0.919	22.5	1.1666e+00	2.86	4.5971e+01	1.21
4	394	0.618	0.926	40.7	7.5734e-01	1.59	3.3524e+01	1.16
5	703	0.574	0.936	45.9	4.6479e-01	1.69	2.4081e+01	1.14
6	1287	0.611	0.941	59.5	2.9602e-01	1.49	1.7575e+01	1.04
7	2426	0.622	0.942	80.9	1.8046e-01	1.56	1.2760e+01	1.01
8	4644	0.651	0.944	114.5	9.7005e-02	1.91	9.2654e+00	0.99
9	8921	0.621	0.945	130.9	5.6155e-02	1.68	6.7420e+00	0.97
10	17095	0.598	0.944	164.5	3.1818e-02	1.75	4.8327e+00	1.02
11	33192	0.610	0.944	176.3	1.6909e-02	1.91	3.4671e+00	1.00
Adaptive refinement using ρ_{L^2}								
0	81	0.563	0.898	5.7	5.7395e+00		1.3966e+02	
1	127	0.679	0.931	7.1	2.1686e+00	4.33	9.0681e+01	1.92
2	211	0.579	0.924	13.4	1.0831e+00	2.74	5.9141e+01	1.68
3	370	0.638	0.938	11.9	5.8917e-01	2.17	4.0737e+01	1.33
4	681	0.675	0.941	17.7	3.2326e-01	1.97	2.9245e+01	1.09
5	1280	0.679	0.940	20.6	1.7163e-01	2.01	2.1150e+01	1.03
6	2412	0.601	0.944	23.5	9.4164e-02	1.90	1.4562e+01	1.18
7	4629	0.680	0.944	27.7	4.8922e-02	2.01	1.0621e+01	0.97
8	8883	0.649	0.944	34.9	2.6327e-02	1.90	7.4636e+00	1.08
9	17099	0.602	0.944	37.4	1.3566e-02	2.03	5.4503e+00	0.96
10	32875	0.609	0.944	41.5	7.2308e-03	1.93	3.8497e+00	1.06

function ρ_{L^2}; this observation, which is also verified by the values of h_{max}/h_{min} in Table 5.1, can be explained by the fact that ρ_{H^1} has larger variations than does ρ_{L^2}.

Table 5.1 contains information about mesh quality, solution errors, and convergence rates at all refinement levels for different refinement strategies for Example 1; the corresponding plots of the error norms ($\|e_h\|_{L^2(\Omega)}$ and $|e_h|_{H^1(\Omega)}$) versus the number of nodes are given in Figure 5.3 where $|e_h|_{H^1(\Omega)}$ denotes the semi-H^1 norm defined by $\|\nabla e_h\|_{L^2(\Omega)}$. One observes that the two CfCVDT-based adaptive methods and the uniform refinement strategy achieve almost perfect convergence rates, i.e., 2 and 1 for $\|e_h\|_{L^2(\Omega)}$ and $|e_h|_{H^1(\Omega)}$, respectively.[4] These, of course, are the expected rates since the exact solution u of Example 1 belongs to $H^2(\Omega)$. The values of q_{min} and q_{avg} given in Table 5.1 demonstrate that the shape quality of the meshes resulting from the CfCVDT-based adaptive strategy is always very good at all levels for both density functions ρ_{L^2} and ρ_{H^1}, although the mesh sizes vary a lot over the Ω, e.g., h_{max}/h_{min} reaches 176.3 at the last level when ρ_{H^1} is used. Note that h_{max}/h_{min} tends to converge for our adaptive methods since u is smooth. Also, as expected, the

[4]The convergence rate for $\|e_h\|_{L^2(\Omega)}$ for the adaptive method using ρ_{H^1} being a little erratic and slightly lower than 2.

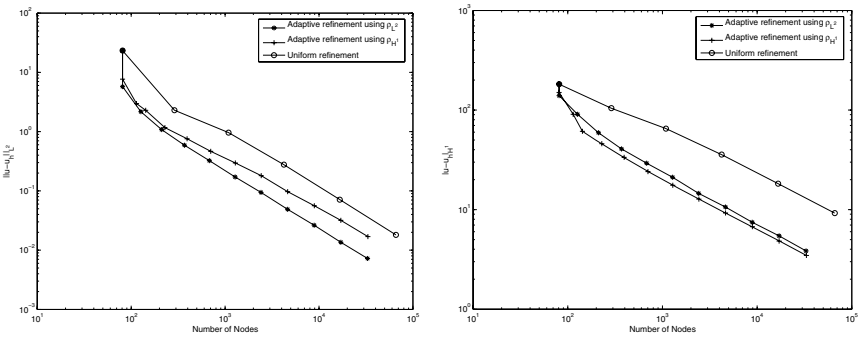


FIG. 5.3. *Error norms versus number of nodes for different refinement strategies for Example* 1. *Left:* $\|e_h\|_{L^2(\Omega)}$. *Right:* $|e_h|_{H^1(\Omega)}$.

adaptive method using ρ_{L^2} as the density function generated approximate solutions u_h having smaller $\|e_h\|_{L^2(\Omega)}$ relative to those obtained using ρ_{H^1}; on the other hand, the latter generated approximate solutions with (slightly) smaller $|e_h|_{H^1(\Omega)}$.

From Table 5.1 and Figure 5.3, one also observes that the CfCVDT-based adaptive methods are much more efficient relative to the uniform refinement strategy. For example, for the uniform refinement method, we have that $\|e_h\|_{L^2(\Omega)} = 1.8071\text{e-}02$ and $|e_h|_{H^1(\Omega)} = 9.2255\text{e+}00$ on the refined mesh with 66,049 nodes. However, for the adaptive methods, the values of these norms are 5.6155e-02 and 6.7420e+00, respectively, on the mesh with only 8,921 nodes when ρ_{H^1} is used as the density function, and 2.6327e-02 and 7.4636e+00, respectively, on the mesh with only 8,883 nodes when ρ_{L^2} is used as the density function. This means that, in the case of the $|e_h|_{H^1(\Omega)}$ norm, the CfCVDT-based adaptive methods are more than 8 times more efficient than the uniform refinement method if considering only the size of the resulting system.

An important optimal property of CfCVDT-based adaptive methods is the equidistribution of the errors over Ω. In order to verify this, we plot, in Figure 5.4, a representative approximate solution u_h and the errors e_h for different refinement methods. It is obvious that the adaptive methods do indeed distribute the errors much more equally than does the uniform refinement method; note that the same scale is used for e_h in Figure 5.4 for all the refinement methods.

5.2. Geometric singularity. The second illustrative problem is given as follows.

EXAMPLE 2. Let $\Omega = \Omega_{R_1} \cup \Omega_{R_2}$, where $\Omega_{R_1} = [-1, 0] \times [0, 1]$ and $\Omega_{R_2} = [0, 1] \times [-1, 1]$ so that Ω is a nonconvex, Γ-shaped region which induces a geometrically based singularity in the solution of the PDE at the origin $(0, 0)$. In (5.1), we set $a(x, y) = 1$ and $b(x, y) = 0$. We use the polar coordinates (r, θ) instead of the Cartesian coordinates $(x, y) = (r\cos\theta, r\sin\theta)$ to describe the exact solution u which is chosen to be

$$u(r,\theta) = s\left(\frac{r-\delta_1}{\delta_2-\delta_1}\right) r^{2/3}\sin\left(\frac{2}{3}\theta\right) + w(r\cos\theta, r\sin\theta) \tag{5.5}$$

with $\delta_1 = 0.02$ and $\delta_2 = 0.25$, where s is the cut-off function

$$s(t) = \begin{cases} 1, & t < 0, \\ -6t^5 + 15t^4 - 10t^3 + 1, & 0 \le t \le 1, \\ 0, & t > 1, \end{cases}$$

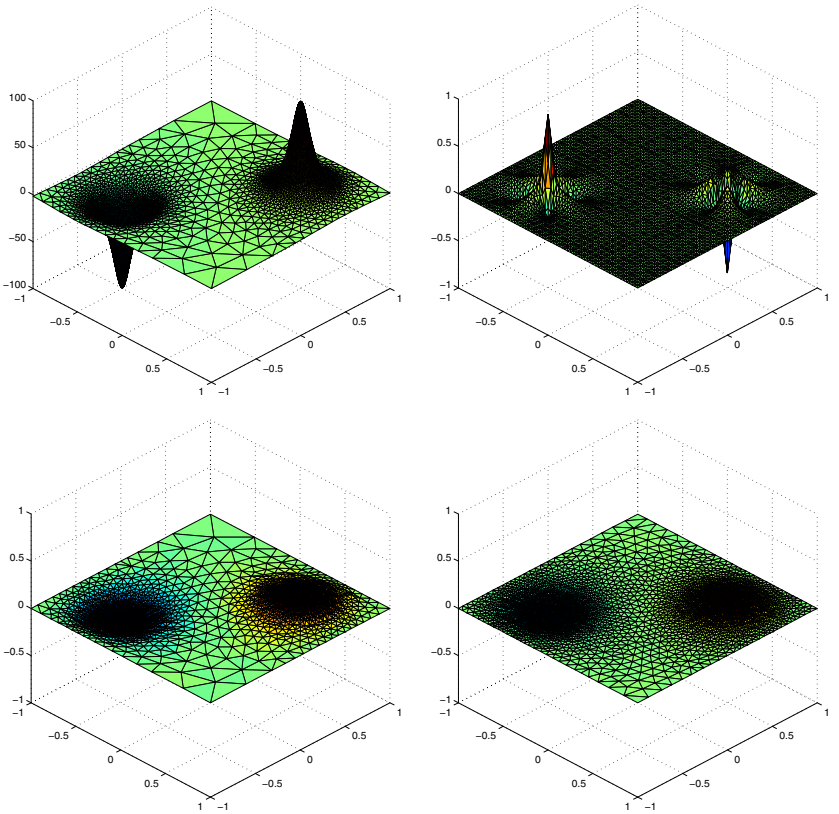


FIG. 5.4. *Plots of an approximate solution and error distributions for different refinement methods for Example* 1. *Top left: A representative approximate solution* u_h. *Top right: The error* e_h *on the mesh with* 4225 *nodes obtained using the uniform refinement method. Bottom left:* e_h *on the mesh with* 4644 *nodes obtained using the CfCVDT-based adaptive method using* ρ_{H^1}. *Bottom right:* e_h *on the mesh with* 4629 *nodes obtained using the CfCVDT-based adaptive method using* ρ_{L^2}.

and $w(x, y) = (x - x^3)(y^2 - y^4)$. Note that w is a smooth function with $w|_{\partial\Omega} = 0$. Then, f and g are again determined from u so that (5.1) is satisfied.

The exponent and angle factor $2/3$ in the exact solution (5.5) emulates the typical singular behavior of solutions of (5.1) in the Γ-shaped domain that has an interior angle equal to $3\pi/2$; see [23]. It is then easy to show that the exact solution u given in (5.5) belongs only to $H^{\frac{5}{3}-\epsilon}(\Omega)$ for any $\epsilon > 0$ and has a strong singularity at the origin. Again, this is the typical regularity one can expect for solutions of (5.1) in a Γ-shaped domain. Note that, in particular, $u \notin H^2(\Omega)$.

The initial coarse mesh (the input for step 0 of Algorithm 3) used for the solution of Example 2 is a uniform Cartesian grid consisting of 65 nodes; see Figure 5.5. The corresponding CfCVDT meshes (the output of step 0) with the same number of nodes produced using the density functions ρ_{H^1} and ρ_{L^2} are also given in Figure 5.5. Figure 5.6 presents repeatedly refined meshes at some levels generated using Algorithm 3. The distributions of nodes in the CfCVDT-based adaptive meshes clearly show the accumulation of nodes near the origin where the singularity in the solution occurs. In order to better visualize the extent of the accumulation, 16-fold magnifications of

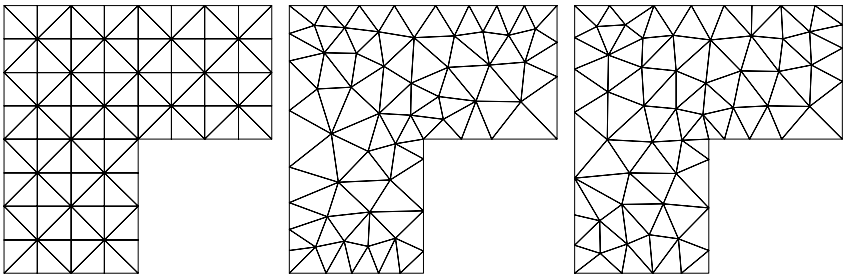


FIG. 5.5. *Initial meshes for Example* 2. *Left: A uniform Cartesian grid with* 65 *nodes. Middle and right: The corresponding CfCVDT meshes with the same number of nodes generated using the density functions* ρ_{H^1} *and* ρ_{L^2}, *respectively.*

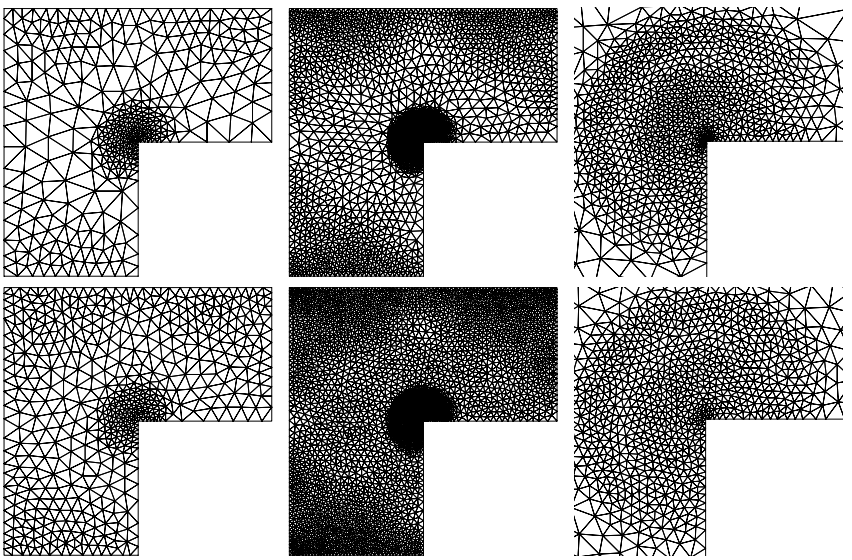


FIG. 5.6. *Refined adaptive meshes at some levels generated by the CfCVDT-based adaptive method for Example* 2. *Top, left to right: For the density function* ρ_{H^1}, 400 *and* 2379 *node meshes and the* 2379 *node case near the singular point magnified* 16 *times. Bottom, left to right: For the density function* ρ_{L^2}, 565 *and* 3617 *nodes and the* 3617 *node case near the singular point magnified* 16 *times.*

the meshes near the origin are also included in Figure 5.6. It is easy to see that again CfCVDT meshes generated using the density function ρ_{H^1} have a little higher concentration of nodes near the singular point than the ones generated using ρ_{L^2}. Also, once again, all triangles remain well shaped at all refinement levels, an observation that is supported by the values of q_{min} and q_{avg} listed in Table 5.2.

Table 5.2 contains information about mesh quality, solution errors, and convergence rates at all refinement levels for different refinement strategies for Example 2; the corresponding plots of the error norms ($\|e_h\|_{L^2(\Omega)}$ and $|e_h|_{H^1(\Omega)}$) versus the number of nodes are given in Figure 5.7. The convergence rates for the uniform refinement method are about 1.48 and 0.85 for $\|e_h\|_{L^2(\Omega)}$ and $|e_h|_{H^1(\Omega)}$, respectively. These rates are a little better than the values 4/3 and 2/3, respectively, that finite element theory predicts for an exact solution $u \in H^{\frac{5}{3}-\epsilon}(\Omega)$; this behavior can possibly be explained

TABLE 5.2

Mesh quality, solution errors, and convergence rates for different refinement strategies for Example 2.

ℓ	n_ℓ	q_{min}	q_{avg}	h_{max}/h_{min}	$\|e_h\|_{L^2(\Omega)}$	CR	$\|e_h\|_{H^1(\Omega)}$	CR
Uniform refinement								
0	65	0.828	0.828	1.0	7.3322e-02		4.5081e-01	
1	225	0.828	0.828	1.0	1.4759e-02	2.31	3.2282e-01	0.48
2	833	0.828	0.828	1.0	4.4383e-03	1.73	1.7465e-01	0.88
3	3201	0.828	0.828	1.0	1.4867e-03	1.58	9.3882e-02	0.89
4	12545	0.828	0.828	1.0	5.5462e-04	1.42	5.0745e-02	0.89
5	49665	0.828	0.828	1.0	1.9910e-04	1.48	2.8144e-02	0.85
Adaptive refinement using ρ_{H^1}								
0	65	0.745	0.929	2.1	3.8000e-02		3.9914e-01	
1	91	0.653	0.915	3.9	1.5791e-02	5.22	3.0636e-01	1.57
2	141	0.602	0.911	6.2	8.8165e-03	2.66	2.0079e-01	1.93
3	232	0.536	0.915	9.2	5.4002e-03	1.97	1.4655e-01	1.27
4	400	0.505	0.923	10.3	2.6396e-03	2.63	1.0745e-01	1.14
5	711	0.579	0.934	15.2	1.3676e-03	2.29	7.7155e-02	1.15
6	1293	0.477	0.934	19.0	6.9433e-04	2.27	5.5410e-02	1.11
7	2379	0.442	0.937	32.2	3.5445e-04	2.21	3.9999e-02	1.07
8	4455	0.490	0.940	67.4	1.9454e-04	1.91	2.8889e-02	1.04
9	8419	0.480	0.941	86.7	9.3771e-05	2.29	2.0725e-02	1.04
10	16005	0.595	0.944	196.6	5.1481e-05	1.87	1.4871e-02	1.03
11	30576	0.517	0.943	294.7	2.5224e-05	2.20	1.0716e-02	1.01
Adaptive refinement using ρ_{L^2}								
0	65	0.645	0.918	2.2	4.2120e-02		4.0369e-01	
1	109	0.731	0.928	2.5	1.6311e-02	3.67	3.2760e-01	0.81
2	180	0.602	0.931	3.6	6.4383e-03	3.71	1.9912e-01	1.99
3	314	0.661	0.933	4.4	3.1812e-03	2.53	1.3687e-01	1.35
4	565	0.648	0.932	5.5	1.6431e-03	2.25	9.7091e-02	1.17
5	1035	0.703	0.938	6.1	9.4153e-04	1.84	7.3907e-02	0.90
6	1925	0.642	0.941	7.9	4.7074e-04	2.23	5.1002e-02	1.20
7	3617	0.649	0.941	9.6	2.5132e-04	1.99	3.8175e-02	0.92
8	6847	0.579	0.944	15.5	1.2593e-04	2.17	2.6637e-02	1.13
9	13048	0.576	0.945	19.0	6.7958e-05	1.91	1.9695e-02	0.94
10	24933	0.579	0.945	28.4	3.5607e-05	2.00	1.3835e-02	1.09

by the superconvergence property of piecewise linear finite element approximations on uniform grids and by the fact that the mesh resolution may still not be fine enough to achieve asymptotic convergence rates; see [9]. However, one sees that the CfCVDT-based adaptive methods still achieve almost perfect convergence rates, i.e., 2 and 1 for $\|e_h\|_{L^2(\Omega)}$ and $|e_h|_{H^1(\Omega)}$, respectively. The values of q_{min} and q_{avg} given in Table 5.2 demonstrate that the quality of the meshes produced by the CfCVDT-based adaptive methods is always very good for both density functions ρ_{H^1} and ρ_{L^2}, even though the mesh sizes vary greatly over the Ω; e.g., h_{max}/h_{min} reaches 194.7 at the last level when the density function ρ_{H^1} is used. It is interesting to observe that, since u does not belong to $H^2(\Omega)$, h_{max}/h_{min} tends to monotonically increase for the adaptive methods. The CfCVDT adaptive method using the density function ρ_{L^2} generated approximate solutions with almost the same values of $\|e_h\|_{L^2(\Omega)}$ as that obtained using the density function ρ_{H^1}; on the other hand, the latter generated approximate solutions with (slightly) smaller values of $|e_h|_{H^1(\Omega)}$.

From Table 5.2 and Figure 5.7, one also observes that the CfCVDT-based adaptive methods are much more efficient relative to the uniform refinement strategy. For example, for the uniform refinement method, we have that $\|e_h\|_{L^2(\Omega)}$ =1.9910e-04 and

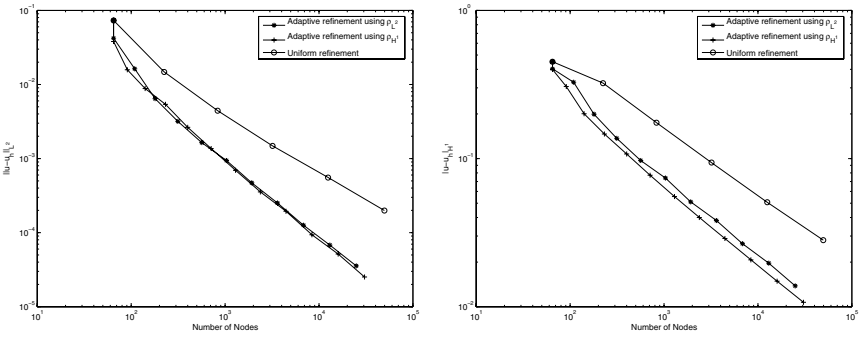


FIG. 5.7. *Error norms versus number of nodes for different refinement strategies for Example* 2. *Left:* $\|e_h\|_{L^2(\Omega)}$. *Right:* $|e_h|_{H^1(\Omega)}$.

$|e_h|_{H^1(\Omega)} = 2.8144\text{e-}02$ on the refined mesh with 49,665 nodes. However, for the adaptive methods, the values of these norms are 1.9454e-04 and 2.8889e-02, respectively, on the mesh with only 4,455 nodes when ρ_{H^1} is used as the density function, and 1.2593e-04 and 2.6637e-02, respectively, on the mesh with only 6,847 nodes when ρ_{L^2} is used as the density function. This means that, in the case of the $\|e_h\|_{H^1(\Omega)}$ norm, the CfCVDT-based adaptive methods are more than 10 times more efficient than the uniform refinement method if only considering the size of the resulting system.

We display a representative approximate solution u_h and the errors e_h for different refinement methods in Figure 5.8. It is again obvious that the CfCVDT-based adaptive methods distribute the errors much more equally over the triangles than does the uniform refinement method.

5.3. Interface singularity. The third illustrative problem is given as follows.

EXAMPLE 3. Let $\Omega = [-1,1] \times [-1,1]$, $b(x,y) = 0$, and

$$a(x,y) = \begin{cases} 1 & (x,y) \in \Omega_1 \cup \Omega_3, \\ 5 & (x,y) \in \Omega_2 \cup \Omega_4, \end{cases}$$

where $\Omega_1 = (0,1) \times (0,1)$, $\Omega_2 = (-1,0) \times (0,1)$, $\Omega_3 = (-1,0) \times (-1,0)$, and $\Omega_4 = (0,1) \times (-1,0)$; see the top left image in Figure 5.9. Note that the coefficient a is discontinuous across the two lines $x = 0$ and $y = 0$. This problem is thus an interface problem [10]. The exact solution u is chosen to be

$$u(r,\theta) = r^{\alpha}\big(p_i \cos(\alpha\theta) + q_i \sin(\alpha\theta)\big) \quad \text{in } \Omega_i, \tag{5.6}$$

with $0 < \alpha < 1$ and for $i = 1,2,3,4$. With a normalization condition such as

$$\sum_{i=1}^{4}(p_i + q_i) = 1,$$

we can easily solve a Sturm–Liouville problem to find that if $\alpha \approx 0.53544094560$ and

$$\{(p_i,q_i)\}_{i=1}^{4} \approx \big\{(14.535673, -0.839562),\ (0.429608, 1.621108),\ (-13.043759, 6.469236),\ (-0.478922, -0.693383)\big\},$$

then u given by (5.6) satisfies

$$\nabla \cdot (a \nabla u) = 0 \quad \text{in } \Omega_i$$

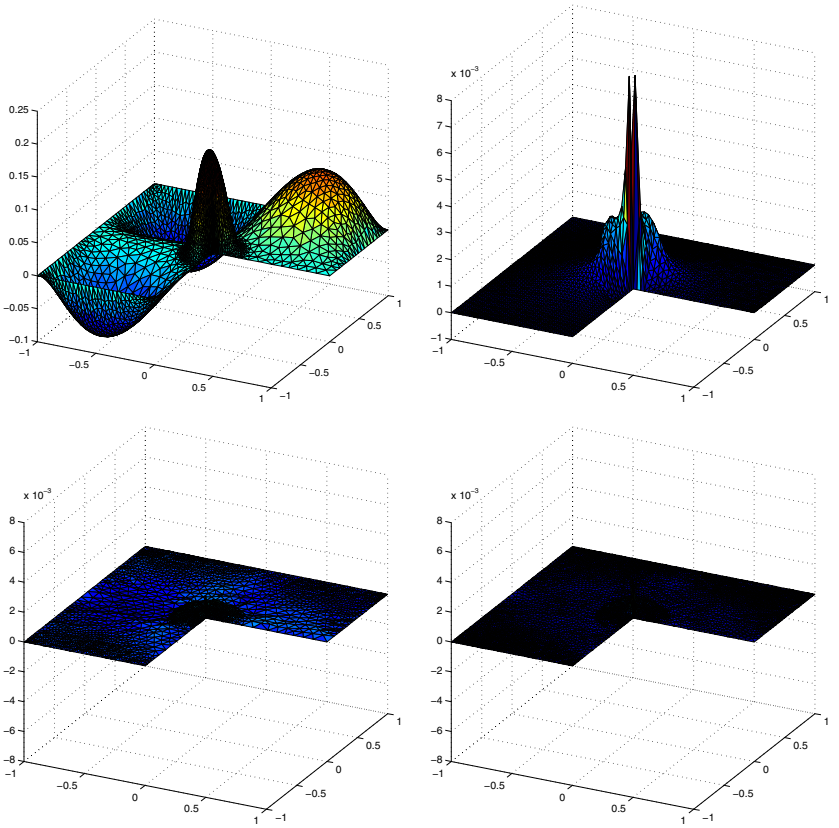


FIG. 5.8. *Plots of an approximate solution and error distributions for different refinement methods for Example* 2. *Top left: A representative approximate solution* u_h. *Top right: The error* e_h *on the mesh with* 3201 *nodes obtained using the uniform refinement method. Bottom left:* e_h *on the mesh with* 2379 *nodes obtained using the CfCVDT-based adaptive method using* ρ_{H^1}. *Bottom right:* e_h *on the mesh with* 3617 *nodes obtained using the CfCVDT-based adaptive method using* ρ_{L^2}.

and the interface conditions

$$\lim_{\theta \to (i\pi/2)^+} u(r,\theta) = \lim_{\theta \to (i\pi/2)^-} u(r,\theta)$$

and

$$\lim_{\theta \to (i\pi/2)^+} a \frac{\partial u(r,\theta)}{\partial \theta} = \lim_{\theta \to (i\pi/2)^-} a \frac{\partial u(r,\theta)}{\partial \theta}$$

for $i = 1, 2, 3, 4$. Note that $u \in H^{1+\alpha-\epsilon}(\Omega)$ for any $\epsilon > 0$ and has a strong singularity at the origin.

The initial coarse mesh, i.e., the input for step 0 of Algorithm 2, used for the solution of Example 3 is the same as that used for Example 1, i.e., a uniform Cartesian grid consisting of 81 nodes; see Figure 5.9. For this interface problem, we need to impose a restriction on the computational meshes: no triangle at any refinement level can straddle the interface lines $x = 0$ and $y = 0$; i.e., the triangulations have to conform to the interfaces, so that the discontinuities of a occur only across mesh edges.

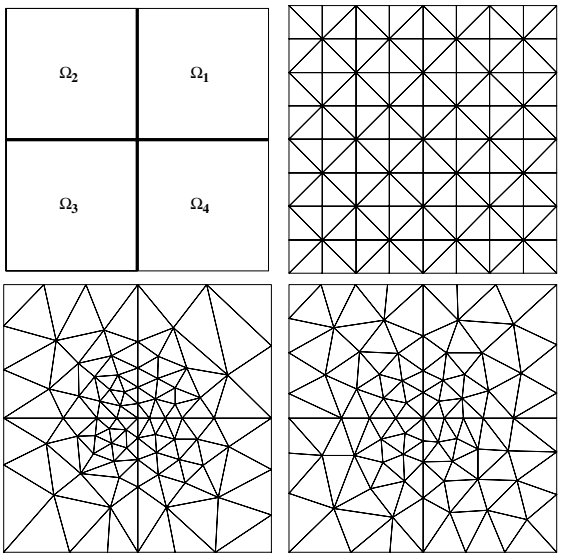


FIG. 5.9. *The domain* Ω *and the initial meshes for Example* 3. *Top left: The domain* Ω *and its partitions. Top right: A uniform Cartesian grid with* 81 *nodes. Bottom: The corresponding CfCVDT meshes generated using the density functions* ρ_{H^1} (*left*) *and* ρ_{L^2} (*right*).

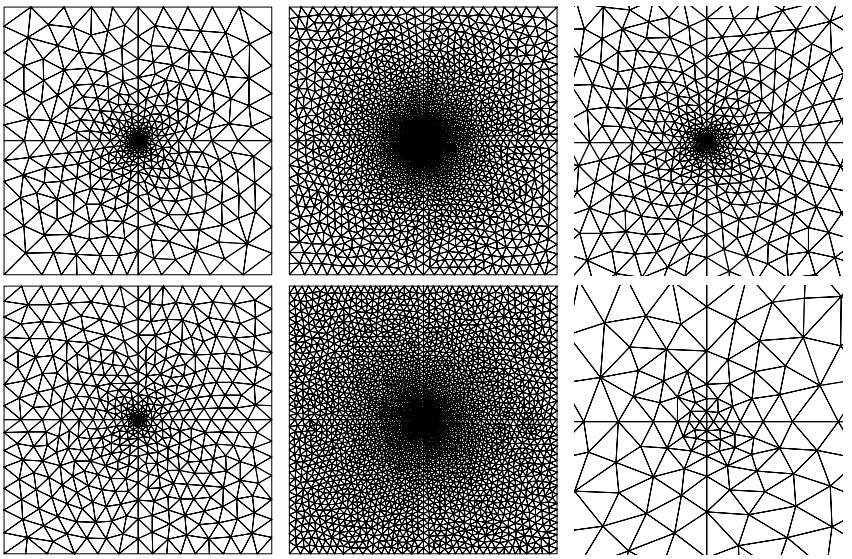


FIG. 5.10. *Refined adaptive meshes at some levels generated by the CfCVDT-based adaptive method for Example* 3. *Top, left to right: For the density function* ρ_{H^1}, 476 *and* 2910 *node meshes and the* 2910 *node case near the singular point magnified* 1024 *times. Bottom, left to right: For the density function* ρ_{L^2}, 436 *and* 2868 *nodes and the* 2868 *node case near the singular point magnified* 1024 *times.*

Uniformly refined grids automatically meet this constraint. Some modifications can be made to Algorithm 2 so that CfCVDT adapted meshes also satisfy this requirement. The initial CfCVDT meshes, i.e., the output of step 0 of Algorithm 2, determined using the density functions ρ_{H^1} and ρ_{L^2} are also given in Figure 5.9. Figure 5.10 displays refined meshes at some levels generated by the CfCVDT-based adaptive method. The

TABLE 5.3

Mesh quality, solution errors, and convergence rates for different refinement strategies for Example 3.

ℓ	n_ℓ	q_{min}	q_{avg}	h_{max}/h_{min}	$\|e_h\|_{L^2(\Omega)}$	CR	$\|e_h\|_{H^1(\Omega)}$	CR
Uniform refinement								
0	81	0.828	0.828	1.0	6.1349e-02		6.5596e-01	
1	289	0.828	0.828	1.0	2.9422e-02	1.06	4.7058e-01	0.48
2	1089	0.828	0.828	1.0	1.3654e-02	1.11	3.2931e-01	0.52
3	4225	0.828	0.828	1.0	6.3877e-03	1.09	2.2900e-01	0.52
4	16441	0.828	0.828	1.0	3.0046e-03	1.09	1.5869e-01	0.53
5	66409	0.828	0.828	1.0	1.4193e-03	1.08	1.0975e-01	0.53
Adaptive refinement using ρ_{H^1}								
0	81	0.489	0.896	5.4	2.8158e-02		5.1027e-01	
1	115	0.557	0.901	12.1	1.4766e-02	3.68	3.8016e-01	1.68
2	176	0.483	0.883	33.7	9.7944e-03	1.93	2.8124e-01	1.42
3	282	0.489	0.899	87.3	5.3705e-03	2.63	2.1415e-01	1.19
4	476	0.522	0.919	206.1	3.2539e-03	1.86	1.5543e-01	1.19
5	853	0.526	0.925	469.2	1.3335e-03	3.04	1.1379e-01	1.07
6	1561	0.455	0.933	1141.8	7.2023e-04	2.04	8.2305e-02	1.07
7	2910	0.447	0.939	3210.3	3.4939e-04	2.32	6.0055e-02	1.01
8	5517	0.484	0.941	6897.3	1.9200e-04	1.87	4.3156e-02	1.03
9	10518	0.487	0.942	13927.4	1.0982e-04	1.73	3.2185e-02	0.91
10	19935	0.521	0.943	30373.2	6.3599e-05	1.71	2.2651e-02	1.10
11	38024	0.596	0.944	53393.1	3.5049e-05	1.85	1.6798e-02	0.93
Adaptive refinement using ρ_{L^2}								
0	81	0.668	0.920	2.7	3.4814e-02		5.4208e-01	
1	139	0.523	0.918	3.7	1.8234e-02	2.40	4.1472e-01	0.99
2	241	0.689	0.929	7.1	8.3887e-03	2.82	2.9469e-01	1.24
3	436	0.529	0.932	11.5	4.3692e-03	2.20	2.1264e-01	1.10
4	808	0.518	0.936	15.8	3.4153e-03	0.80	1.7719e-01	0.60
5	1512	0.659	0.939	23.7	1.2532e-03	3.20	1.1830e-01	1.29
6	2868	0.640	0.944	33.2	7.1078e-04	1.77	8.7174e-02	0.95
7	5475	0.551	0.942	56.8	3.9287e-04	1.83	6.1441e-02	1.08
8	10491	0.619	0.944	90.7	2.1873e-04	1.80	4.6519e-02	0.86
9	20121	0.545	0.944	112.2	1.1865e-04	1.88	3.3943e-02	0.97

distributions of nodes in the CfCVDT-adapted meshes clearly show the accumulation of nodes near the origin. In order to better visualize the extent of the accumulation, a portion of the meshes near the origin, magnified 1024 times, are also included in Figure 5.10. It is easy to see that the CfCVDT-adapted meshes generated using the density function ρ_{H^1} have a much higher concentration of nodes near the singular point, i.e., the origin, than those generated using ρ_{L^2}. This observation is supported by the values of h_{max}/h_{min} listed in Table 5.3. Again, all triangles are well shaped at all refinement levels, an observation that is supported by the values of q_{min} and q_{avg} listed in Table 5.3.

Table 5.3 contains information about mesh quality, solution errors, and convergence rates at all refinement levels for different refinement strategies for Example 3; the corresponding plots of the error norms ($\|e_h\|_{L^2(\Omega)}$ and $|e_h|_{H^1(\Omega)}$) versus the number of nodes are given in Figure 5.11. The convergence rates for the uniform refinement method are about 1.08 and 0.53 for $\|e_h\|_{L^2(\Omega)}$ and $|e_h|_{H^1(\Omega)}$, respectively. These match very well with those predicted by finite element theory, i.e., 2α and α, respectively, for $u \in H^{\alpha-\epsilon}(\Omega)$. However, one sees that the CfCVDT-based adaptive methods still achieve almost perfect convergence rates, i.e., 1.0 and 1.8 for $|e_h|_{H^1(\Omega)}$ and $|e_h|_{L^2(\Omega)}$, respectively. The values of q_{min} and q_{avg} given in Table 5.3 demonstrate that the

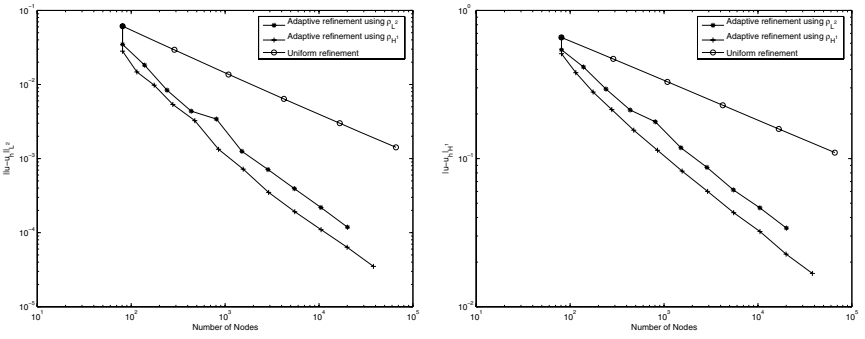


FIG. 5.11. *Error norms versus number of nodes for different refinement strategies for Example* 3. *Left:* $\|e_h\|_{L^2(\Omega)}$. *Right:* $|e_h|_{H^1(\Omega)}$.

quality of the meshes produced by the CfCVDT-based adaptive methods is always very good at all refinement levels and for both density functions ρ_{H^1} and ρ_{L^2}, although the mesh sizes vary greatly over the Ω; e.g., h_{max}/h_{min} reaches 53393.1 at the last level when the density function ρ_{H^1} is used. Again, since u is not in $H^2(\Omega)$, we see that h_{max}/h_{min} tends to monotonically increase for the adaptive methods. Note that in this case, the CfCVDT-based adaptive method using the density function ρ_{H^1} generated approximate solutions having both smaller $\|e_h\|_{L^2(\Omega)}$ and $\|e_h\|_{H^1(\Omega)}$ than using ρ_{L^2}. We believe this phenomenon is due to the strong singularity of the exact solution u while the density functions are constructed based on the assumption that $u \in H^2(\Omega)$.

From Table 5.3 and Figure 5.11, one also observes that the CfCVDT-based adaptive methods are much more efficient relative to the uniform refinement strategy. For example, for the uniform refinement method, we have that $\|e_h\|_{L^2(\Omega)} = 1.4193\text{e-}03$ and $|e_h|_{H^1(\Omega)} = 1.0975\text{e-}01$ on the refined mesh with $66,409$ nodes. However, for the adaptive methods, the values of these norms are 1.3335e-03 and 1.1379e-01, respectively, on the mesh with only 853 nodes when ρ_{H^1} is used as the density function, and 1.2532e-03 and 1.1830e-01, respectively, on the mesh with only $1,512$ nodes when ρ_{L^2} is used as the density function. This means that, in the case of the $|e_h|_{H^1(\Omega)}$ norm, the CfCVDT-based adaptive methods are more than 60 times more efficient than the uniform refinement method if considering only the size of the resulting system.

We display a representative approximate solution u_h and the errors e_h for different refinement methods in Figure 5.12. It is again obvious that the CfCVDT-based adaptive methods distribute the errors much more equally over the triangles than does the uniform refinement method.

5.4. More complicated geometries. Our last two examples involve more complicated geometries and serve to illustrate the robustness and effectiveness of our adaptive CfCVDT-based mesh generation algorithm.

EXAMPLE 4. Let $\Omega = \Omega_S - \Omega_{H^1} \cup \Omega_{H_2}$, where $\Omega_S = [0,1] \times [0,1]$ and Ω_{H^1} and Ω_{H_2} are two open hexagons formed by the set of vertices $\{0.25 + 0.1\cos(j-1)\theta,\ 0.75 + 0.1\sin(j-1)\theta)\}_{j=1}^{6}$ and $\{0.6 + 0.1\cos(j-1)\theta,\ 0.4 + 0.1\sin(j-1)\theta)\}_{j=1}^{6}$ with $\theta = \pi/3$, respectively. Clearly, Ω is a square domain having two hexagonal holes; see Figure 5.13. In general, solutions of (5.1) possess large geometric singularities at

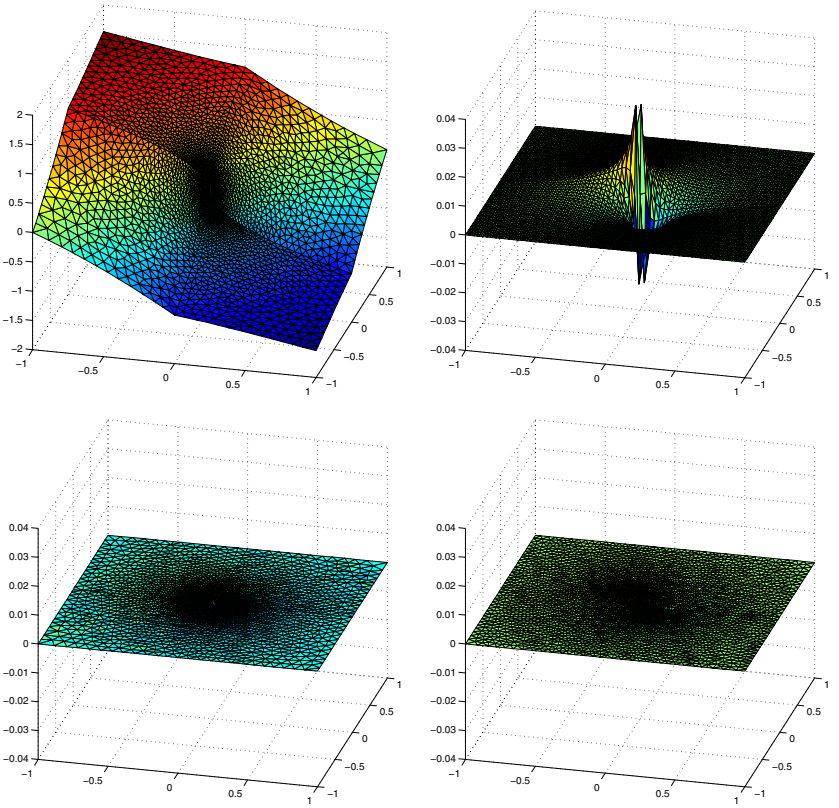


FIG. 5.12. *Plots of an approximate solution and error distributions for different refinement methods for Example* 3. *Top left: A representative approximate solution* u_h. *Top right: The error* e_h *on the mesh with* 4225 *nodes obtained using the uniform refinement method. Bottom left:* e_h *on the mesh with* 2910 *nodes obtained using the CfCVDT-based adaptive method using* ρ_{H^1}. *Bottom right:* e_h *on the mesh with* 2910 *nodes obtained using the CfCVDT-based adaptive method using* ρ_{L^2}.

the vertices of the two interior hexagons. Set

$$a(x,y) = 1 + 10x^2 + y^2, \quad b(x,y) = 1 + x^2, \quad f(x,y) = 1, \text{ and } \quad g(x,y) = 0. \tag{5.7}$$

Although an analytic form of the exact solution u of (5.1) with the data (5.7) is not known, it is known that u belongs only to $H^{\frac{7}{4}-\epsilon}(\Omega)$ for any $\epsilon > 0$ and not to $H^2(\Omega)$.

5. Let $\Omega = \Omega_{S_1} - \Omega_{S_2}$, where $\Omega_{S_1} = [-1,1] \times [-1,1]$ and $\Omega_{S_2} = (-0.5,0.5) \times (-0.5,0.5)$. Set

$$b(x,y) = 0, \quad f(x,y) = 1, \quad g(x,y) = 0 \tag{5.8}$$

and

$$a(x,y) = \begin{cases} 1 & (x,y) \in \Omega_1, \\ 20 & (x,y) \in \Omega_2, \\ 20 & (x,y) \in \Omega_3, \\ 400 & (x,y) \in \Omega_4, \end{cases} \tag{5.9}$$

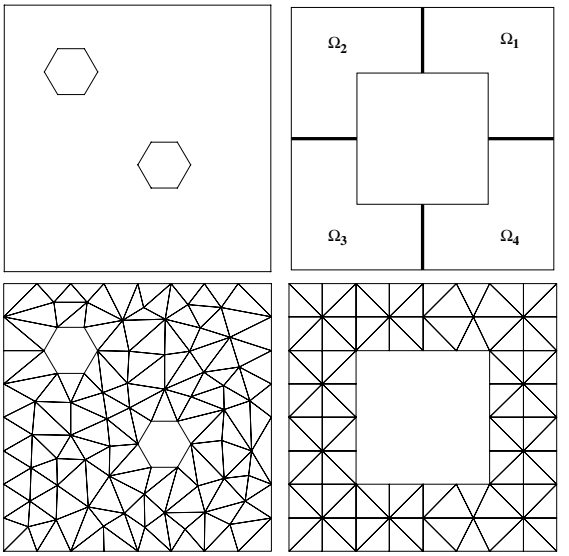


FIG. 5.13. *Top: The domain* Ω *for Examples* 4 (*left*) *and* 5 (*right*). *Bottom: Initial meshes for Examples* 4 (*left*) *and* 5 (*right*).

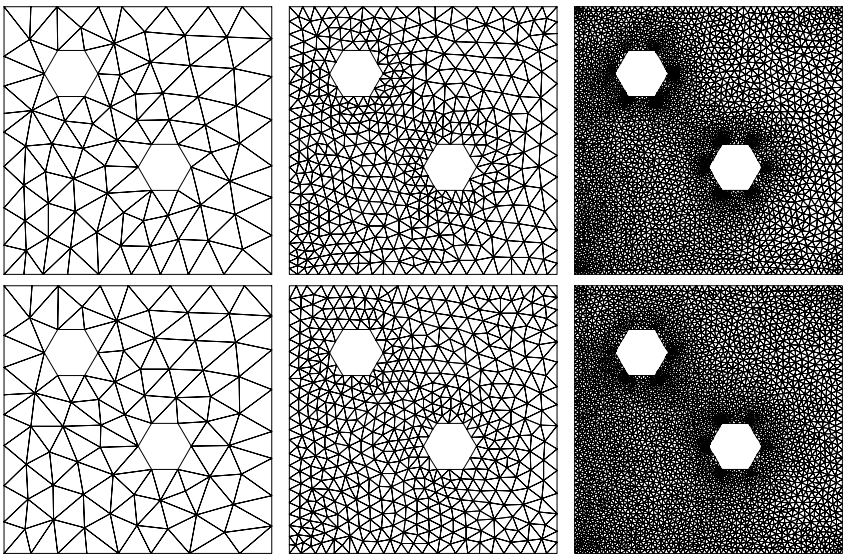


FIG. 5.14. *Repeatedly refined meshes at different levels generated by the CfCVDT-based adaptive methods for Example* 4. *Top:* 94, 507, *and* 3096 *nodes using the density function* ρ_{H^1}. *Bottom:* 94, 524, *and* 3373 *nodes using the density function* ρ_{L^2}.

where Ω_i for $i = 1, 2, 3, 4$ are defined in Figure 5.13. Note that a is discontinuous in Ω. We again do not know an analytic form of the exact solution u of (5.1) with the data (5.8) and (5.9), but we do know that globally u belongs only to $H^1(\Omega)$ (and not $H^2(\Omega)$), but $u|_{\Omega_i} \in H^2(\Omega_i)$ for $i = 1, 2, 3, 4$.

The initial coarse meshes used for the solution of Examples 4 and 5 are displayed in Figure 5.13. Figures 5.14 and 5.15 present repeatedly refined meshes at different levels

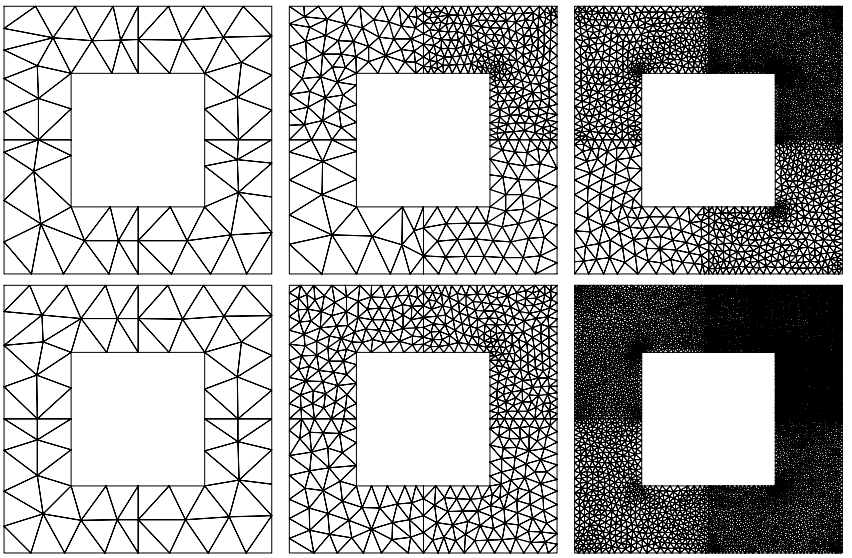


FIG. 5.15. *Repeatedly refined meshes at different levels generated by the CfCVDT-based adaptive methods for Example* 5. *Top:* 70, 409, *and* 2474 *nodes using the density function* ρ_{H^1}. *Bottom:* 70, 492, *and* 3020 *nodes using the density function* ρ_{L^2}.

TABLE 5.4
Mesh quality by different refinement methods for Example 4.

Uniform refinement						
ℓ	0	1	2	3	4	5
n_l	94	336	1261	4875	19159	75951
q_{min}	0.467	0.467	0.467	0.467	0.467	0.467
q_{avg}	0.862	0.862	0.862	0.862	0.862	0.862
h_{max}/h_{min}	2.4	2.4	2.4	2.4	2.4	2.4
Adaptive refinement using ρ_{H^1}						
ℓ	0	1	2	3	4	5
n_l	94	146	280	507	921	1680
q_{min}	0.721	0.727	0.614	0.715	0.636	0.649
q_{avg}	0.919	0.936	0.941	0.942	0.936	0.941
h_{max}/h_{min}	1.7	2.5	2.7	3.2	3.9	5.8
ℓ	6	7	8	9		
n_l	3096	5770	10838	20506		
q_{min}	0.538	0.556	0.567	0.529		
q_{avg}	0.940	0.941	0.941	0.945		
h_{max}/h_{min}	9.4	10.9	18.8	30.7		
Adaptive refinement using ρ_{L^2}						
ℓ	0	1	2	3	4	5
n_l	94	160	287	524	968	1797
q_{min}	0.721	0.661	0.628	0.655	0.663	0.620
q_{avg}	0.918	0.935	0.939	0.942	0.938	0.941
h_{max}/h_{min}	1.6	2.1	2.2	2.7	3.0	3.8
ℓ	6	7	8	9		
n_l	3373	6384	12168	23237		
q_{min}	0.625	0.465	0.608	0.571		
q_{avg}	0.942	0.944	0.943	0.944		
h_{max}/h_{min}	4.0	5.9	6.5	8.6		

TABLE 5.5
Mesh quality by different refinement methods for Example 5.

Uniform refinement						
ℓ	0	1	2	3	4	5
n_l	70	232	832	3136	12160	47872
q_{min}	0.828	0.828	0.828	0.828	0.828	0.828
q_{avg}	0.845	0.845	0.845	0.845	0.845	0.845
h_{max}/h_{min}	1.3	1.3	1.3	1.3	1.3	1.3
Adaptive refinement using ρ_{H^1}						
ℓ	0	1	2	3	4	5
n_l	70	90	137	233	409	734
q_{min}	0.548	0.653	0.681	0.583	0.518	0.541
q_{avg}	0.903	0.917	0.931	0.930	0.929	0.38
h_{max}/h_{min}	1.5	2.4	3.9	5.8	9.6	15.1
ℓ	6	7	8	9	10	11
n_l	1344	2474	4608	8685	16519	31555
q_{min}	0.524	0.540	0.493	0.531	0.472	0.419
q_{avg}	0.935	0.939	0.942	0.943	0.944	0.942
h_{max}/h_{min}	19.9	33.6	86.4	170.0	288.1	381.9
Adaptive refinement using ρ_{L^2}						
ℓ	0	1	2	3	4	5
n_l	94	141	276	492	888	1625
q_{min}	0.747	0.720	0.697	0.690	0.534	0.612
q_{avg}	0.921	0.931	0.936	0.935	0.935	0.943
h_{max}/h_{min}	1.38	2.2	4.8	4.5	6.8	6.6
ℓ	6	7	8	9	10	
n_l	3020	5612	10477	19611	38354	
q_{min}	0.635	0.502	0.567	0.595	0.582	
q_{avg}	0.941	0.944	0.945	0.945	0.944	
h_{max}/h_{min}	9.1	11.9	13.9	15.5	25.2	

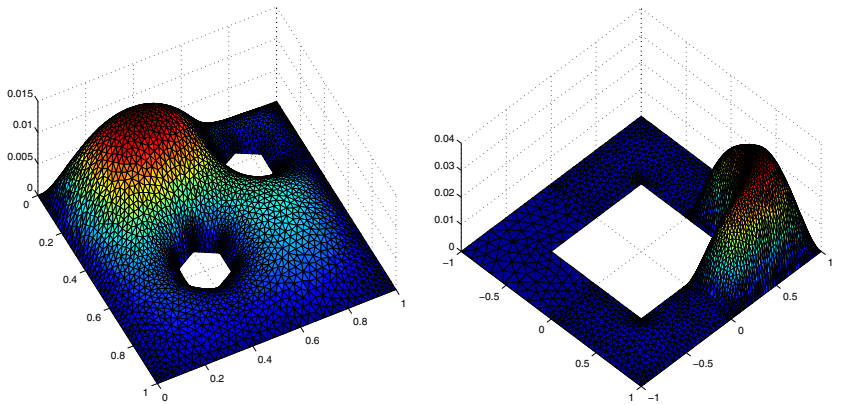


FIG. 5.16. *Plots of the approximate solutions u_h obtained using the CfCVDT-based adaptive methods. Left: u_h for Example* 4 *on the mesh with* 3096 *nodes generated using ρ_{H^1}. Right: u_h for Example* 5 *on the mesh with* 2474 *nodes generated using ρ_{H^1}.*

generated by the CfCVDT-based adaptive methods for Examples 4 and 5, respectively, for both density functions ρ_{H^1} and ρ_{L^2}. Information about mesh quality is given in Tables 5.4 and 5.5, respectively. Figure 5.16 displays approximate solutions u_h computed by our adaptive methods for the two examples.

6. Conclusions. In this paper, we presented an efficient adaptive mesh refining algorithm for elliptic PDEs that combines a posteriori error estimation with centroidal Voronoi–Delaunay tessellations of domains in two dimensions. The two ingredients are well linked together by the fact that the density function required by the second one is defined and computed from the first one with standard interpolations. Various numerical experiments were carried out and showed that our techniques always obtained optimal convergence rates for the piecewise linear finite elements with respect to both H^1 and L^2 norms and worked pretty robustly. This mesh adaptation strategy can be easily generalized and applied to higher-order finite element approximations or mixed finite element formulations. We also would like to remark that we are currently studying the extension of this methodology to problems in three dimensions and to time dependent problems.

Acknowledgment. The authors would like to thank the referees for their valuable suggestions which greatly improved the paper.

REFERENCES

[1] S. Arya and D. Mount, *Approximate nearest neighbor searching*, in Proceedings of the 4th Annual ACM–SIAM Symposium on Discrete Algorithms (SODA'93), ACM, New York, SIAM, Philadelphia, 1993, pp. 271–280.

[2] M. Ainsworth and J. Oden, *A Posteriori Error Estimation in Finite Element Analysis*, Wiley, New York, 2002.

[3] R. E. Bank and R. K. Smith, *Mesh smoothing using a posteriori error estimates*, SIAM J. Numer. Anal., 34 (1997), pp. 979–997.

[4] I. Babuška and A. Miller, *A feedback finite element method with a posteriori error estimation Part* I, Comput. Methods Appl. Mech. Engrg., 61 (1987), pp. 1–40.

[5] I. Babuška and W. C. Rheinboldt, *Error estimates for adaptive finite element computations*, SIAM J. Numer. Anal., 15 (1978), pp. 736–754.

[6] I. Babuška and W. C. Rheinboldt, *A posteriori error estimates for the finite element method*, Internat. J. Numer. Methods Engrg., 12 (1978), pp. 1597–1615.

[7] I. Babuška and W. C. Rheinboldt, *A posteriori error analysis of finite element solutions for one-dimensional problems*, SIAM J. Numer. Anal., 18 (1981), pp. 565–589.

[8] P. Binev, W. Dahmen, and R. DeVore, *Adaptive finite element methods with convergence rates*, Numer. Math., 97 (2004), pp. 219–268.

[9] S. Brenner, *Multigrid methods for the computation of singular solutions and stress intensity factors* I: *Corner singularities*, Math. Comp., 68 (2003), pp. 559–583.

[10] S. Brenner and L.-Y. Sung, *Multigrid methods for the computation of singular solutions and stress intensity factors* III: *Interface singularities*, Comput. Methods Appl. Mech. Engrg., 192 (2003), pp. 4687–4702.

[11] C. Carstensen, *All first-order averaging techniques for a posteriori finite element error control on unstructured grids are efficient and reliable*, Math. Comp., 73 (2003), pp. 1153–1165.

[12] L. Chen and J. Xu, *Optimal Delaunay triangulation*, J. Comput. Math., 22 (2004), pp. 299–308.

[13] P.-G. Ciarlet, *The Finite Element Method for Elliptic Problems*, North–Holland, Amsterdam, 1978.

[14] W. Dörfler, *A convergent adaptive algorithm for Poisson's equation*, SIAM J. Numer. Anal., 33 (1996), pp. 1106–1124.

[15] Q. Du, V. Faber, and M. Gunzburger, *Centroidal Voronoi tessellations: Applications and algorithms*, SIAM Rev., 41 (1999), pp. 637–676.

[16] Q. Du and M. Gunzburger, *Grid generation and optimization based on centroidal Voronoi tessellations*, Appl. Math. Comput., 133 (2002), pp. 591–607.

[17] Q. Du, M. D. Gunzburger, and L. Ju, *Constrained centroidal Voronoi tessellations for surfaces*, SIAM J. Sci. Comput., 24 (2003), pp. 1488–1506.

[18] Q. Du, M. Gunzburger, and L. Ju, *Voronoi-based finite volume methods, optimal Voronoi meshes and PDEs on the sphere*, Comput. Methods Appl. Mech. Engrg., 192 (2003), pp. 3933–3957.

[19] Q. Du and D. Wang, *Tetrahedral mesh generation and optimization based on centroidal Voronoi tessellations*, Internat. J. Numer. Methods Engrg., 56 (2003), pp. 1355–1373.
[20] Q. Du and D. Wang, *Anisotropic centroidal Voronoi tessellations and their applications*, SIAM. J. Sci. Comput., 26 (2005), pp. 737–761.
[21] D. Field, *Laplacian smoothing and Delaunay triangulation*, Comm. Appl. Numer. Methods, 4 (1988), pp. 709–712.
[22] D. Field, *Quantitative measures for initial meshes*, Internat. J. Numer. Methods Engrg., 47 (2000), pp. 887–906.
[23] P. Grisvard, *Elliptic Problems in Nonsmooth Domains*, Pitman, Boston, 1985.
[24] L. Ju, Q. Du, and M. Gunzburger, *Probabilistic methods for centroidal Voronoi tessellations and their parallel implementations*, Parallel Comput., 28 (2002), pp. 1477–1500.
[25] P. Morin, R. H. Nochetto, and K. G. Siebert, *Data oscillation and convergence of adaptive FEM*, SIAM J. Numer. Anal., 38 (2000), pp. 466–488.
[26] P. Morin, R. H. Nochetto, and K. G. Siebert, *Convergence of adaptive finite element methods*, SIAM Rev., 44 (2002), pp. 631–658.
[27] P. Morin, R. Nochetto, and K. Siebert, *Local problems on stars: A posteriori error estimators, convergence, and performance*, Math. Comp., 72 (2003), pp. 1067–1097.
[28] P.-O. Persson and G. Strang, *A simple mesh generator in MATLAB*, SIAM Rev., 46 (2004), pp. 329–345.
[29] J. Shewchuk, *Triangle: Engineering a 2D quality mesh generator and Delaunay triangulator*, in Applied Computational Geometry, Lecture Notes in Comput. Sci. 1148, Springer-Verlag, New York, 1996, pp. 203–222.
[30] J. Shewchuk, *What is a good linear element? Interpolation, conditioning and quality measures*, in Proceedings of the 11th International Meshing Roundtable, Sandia National Laboratories, Albuquerque, NM, 2002, pp. 115–126.
[31] R. Verfürth, *A Review of A Posteriori Error Estimation and Adaptive Mesh-Refinement Techniques*, Wiley-Teubner, Chichester, UK, 1996.

SIAM J. SCI. COMPUT.
Vol. 28, No. 6, pp. 2054–2074
© 2006 Society for Industrial and Applied Mathematics

AN EMBEDDED BOUNDARY METHOD FOR THE WAVE EQUATION WITH DISCONTINUOUS COEFFICIENTS*

HEINZ-OTTO KREISS† AND N. ANDERS PETERSSON‡

Abstract. A second order accurate embedded boundary method for the two-dimensional wave equation with discontinuous wave propagation speed is described. The wave equation is discretized on a Cartesian grid with constant grid size and the interface (across which the wave speed is discontinuous) is allowed to intersect the mesh in an arbitrary fashion. By using ghost points on either side of the interface, previous embedded boundary techniques for the Neumann and Dirichlet problems are generalized to satisfy the jump conditions across the interface to second order accuracy. The resulting discretization of the jump conditions has the desirable property that each ghost point can be updated independently of all other ghost points, resulting in a fully explicit time-integration method. We prove that the one-dimensional restriction of the method is stable without damping for arbitrary locations of the interface relative to the grid. For the two-dimensional case, the previously developed fourth order A^TA-dissipation is generalized to handle jump conditions. We demonstrate that this operator provides sufficient stabilization to enable long-time simulations while being weak enough to preserve the accuracy of the solution. Numerical examples are given where the method is used to study electromagnetic scattering of a plane wave by a dielectric cylinder. The numerical solutions are evaluated against the analytical solution due to Mie, and pointwise second order accuracy is confirmed.

Key words. wave equation, discontinuous coefficients, jump condition, embedded boundary, finite differences

AMS subject classifications. 65M06, 65M12

DOI. 10.1137/050641399

1. Introduction.

This paper describes a second order accurate Cartesian embedded boundary method for the two-dimensional wave equation with discontinuous wave propagation speed. Motivated by wave propagation problems from applications like seismology, acoustics, and general relativity, where the underlying differential equations are systems of second order hyperbolic partial differential equations, we develop a numerical method that directly discretizes the wave equation in second order formulation. This approach extends previous research [1, 2, 3] to the case of discontinuous coefficients.

For every second order hyperbolic system, there is an equivalent but larger first order system. For example, two-dimensional acoustic wave propagation is governed by a scalar second order wave equation for the pressure, or by a system of three first order hyperbolic equations governing the two velocity components and the pressure. Most previous numerical methods for this type of problem have focused on the first order formulation [4, 5, 6]. For linear wave propagation, a staggered grid is often used to avoid complications with stability of extra numerical boundary conditions [7] and spurious waves traveling in the wrong direction [8]. However, Cartesian staggered grid discretizations are difficult to generalize to handle complex geometries, that is, boundaries that intersect the grid in an arbitrary way.

*Received by the editors September 28, 2005; accepted for publication (in revised form) May 24, 2006; published electronically December 5, 2006. This work was performed under the auspices of the U.S. Department of Energy by the University of California, Lawrence Livermore National Laboratory, under contract W-7405-Eng-48.

http://www.siam.org/journals/sisc/28-6/64139.html

†Träskö-Storö Institute of Mathematics, Stockholm, Sweden (hokreiss@nada.kth.se).

‡Center for Applied Scientific Computing, LLNL, Livermore, CA (andersp@llnl.gov).

The accuracy of a finite difference approximation of the wave equation with discontinuous coefficients was analyzed by Brown [9]. For the one-dimensional case, he proved that the amplitude error in reflected and transmitted waves is determined by the accuracy by which the jump conditions are discretized, while the phase error is determined by the accuracy of the discretization in the interior of the domain. In the one-dimensional case, Tikhonov and Samarskiĭ's [10] averaging formula was used to obtain a second order accurate approximation without explicitly discretizing the jump conditions. Numerical calculations indicated a significant benefit of combining the second order treatment of the jump conditions with a fourth order method in the interior of the domain. Unfortunately, it is not known how to generalize the averaging formula to the two-dimensional case when the discontinuity is not aligned with the grid, and the combinations of a first order treatment of the jump conditions with a second or fourth order formula away from the interface gave less impressive results than in the one-dimensional case.

A fully second order immersed interface method for solving the two-dimensional acoustic wave equation with discontinuous coefficients was developed by Zhang and LeVeque [11]. Here, the problem was written as a hyperbolic system of three first order equations and special difference formulas were developed near the interface, which take the location of the interface and the jump conditions into account to achieve second order accuracy. For each grid point next to the interface, a linear system with 54 equations for 54 unknowns had to be solved to find the values of the coefficients in the local difference formula. For more complicated hyperbolic systems (such as the elastic wave equation), even larger systems of equations must be solved to set up the coefficients.

Finite difference methods for the time-dependent Maxwell equations have been studied extensively, and a relatively recent overview of the field is provided by Hesthaven [12]. Maxwell's equations are usually written as a first order hyperbolic system and a Cartesian staggered grid is often employed to discretize the equations in space. Several second, fourth, and even higher order accurate schemes have been derived to handle embedded discontinuous coefficients in the one-dimensional case [13, 14, 15, 16], but these methods generalize only to the two-dimensional case when the material interface is parallel to a grid direction. A brief description of a compact fourth order accurate implicit staggered grid scheme based on Pade' approximations is presented by Turkel and Yefet [17]. Here the Dirichlet condition associated with a perfectly electric conducting boundary is satisfied on an embedded boundary by locally modifying the uniform mesh to make the nearest grid points fall on the boundary. However, it is not clear how this approach can be generalized to handle jump conditions across material interfaces. Furthermore, this approach can result in a severe time step restriction due to small cells at the boundary. An alternative to the staggered grid approach is presented by Cai and Deng [18], who solve the first order hyperbolic system on a node-based grid by combining a Lax–Wendroff scheme in the interior with an upwinding approach for satisfying jump conditions across an embedded material interface. However, solving a first order system on a node-based grid can lead to the aforementioned problem with spurious waves traveling in the wrong direction [8]. Furthermore, the mesh needs to be finer to achieve the same accuracy compared to when the wave equation is solved in second order formulation [1].

Higher order finite difference schemes for Maxwell's equations have also been extended to the curvilinear case [19] and to overlapping grids [20], but these approaches rely on the availability of a smooth mesh conforming to the interface. We also remark that there are many Cartesian embedded boundary methods for elliptic and parabolic

partial differential equations, e.g., [21, 22, 23, 24, 25, 26].

In this paper, we consider the scalar second order wave equation in a two-dimensional domain $\Omega = \Omega_I \cup \Omega_{II}$, with a piecewise constant coefficient $\rho(\boldsymbol{x}) > 0$,

$$\rho(\boldsymbol{x}) = \begin{cases} \rho_I, & \boldsymbol{x} \in \Omega_I, \\ \rho_{II}, & \boldsymbol{x} \in \Omega_{II}. \end{cases}$$

Let $u(\boldsymbol{x}, t)$ and $w(\boldsymbol{x}, t)$ denote the solutions in the subdomains Ω_I and Ω_{II}, satisfying

$$u_{tt} = \frac{1}{\rho_I}\Delta u + F(\boldsymbol{x}, t), \qquad \boldsymbol{x} \in \Omega_I, \quad t \geq 0, \tag{1}$$

$$w_{tt} = \frac{1}{\rho_{II}}\Delta w + F(\boldsymbol{x}, t), \quad \boldsymbol{x} \in \Omega_{II}, \quad t \geq 0, \tag{2}$$

$$u(\boldsymbol{x}, 0) = U_0(\boldsymbol{x}), \quad u_t(\boldsymbol{x}, 0) = U_1(\boldsymbol{x}), \qquad \boldsymbol{x} \in \Omega_I, \tag{3}$$

$$w(\boldsymbol{x}, 0) = W_0(\boldsymbol{x}), \quad w_t(\boldsymbol{x}, 0) = W_1(\boldsymbol{x}), \qquad \boldsymbol{x} \in \Omega_{II}. \tag{4}$$

Here, $F(\boldsymbol{x}, t)$ is a forcing function. Let Γ be the smooth interface between Ω_I and Ω_{II}, across which the solutions are coupled by the jump conditions

$$u = w, \qquad \boldsymbol{x} \in \Gamma, \quad t \geq 0, \tag{5}$$

$$\frac{1}{\rho_I}\frac{\partial u}{\partial n} = -\frac{1}{\rho_{II}}\frac{\partial w}{\partial n}, \quad \boldsymbol{x} \in \Gamma, \quad t \geq 0. \tag{6}$$

Note that the normal derivatives are taken outward from both Ω_I and Ω_{II}. Hence we have the minus sign in the latter equation.

For the purpose of our discussion, we assume that Ω_I is a bounded domain inside Ω_{II}, and that Ω_{II} has a rectangular outer boundary where Dirichlet or Neumann boundary conditions are enforced. The interface between Ω_I and Ω_{II} can however have an arbitrary smooth shape. As will be demonstrated below, the method can be generalized to several subdomains. Nonrectangular outer boundaries can be handled using the embedded boundary techniques in [2] and [3].

We discretize the two-dimensional wave equations on a Cartesian grid $\boldsymbol{x}_{i,j} = (ih, jh)$ in space, where $h > 0$ is the grid size, and let $t_n = n\delta_t$, $n = 0, 1, 2, \ldots$, denote the time-discretization with step size $\delta_t > 0$. We take $u^n_{i,j}$ and $w^n_{i,j}$ to be the difference approximations of $u(x_i, y_j, t_n)$ and $w(x_i, y_j, t_n)$, respectively. At all grid points interior to Ω_I, a second order accurate approximation of the Laplacian of u is given by

$$\Delta_h u^n_{i,j} =: \frac{1}{h^2}(u^n_{i+1,j} + u^n_{i-1,j} + u^n_{i,j+1} + u^n_{i,j-1} - 4u^n_{i,j}), \quad \boldsymbol{x}_{i,j} \in \Omega_I. \tag{7}$$

In order to form (7) at all grid points in Ω_I, we also define $u^n_{i,j}$ at the set of ghost points G_I:

$$G_I = \{(i, j),\ \boldsymbol{x}_{i,j} \notin \Omega_I,\ \text{but at least one of } \boldsymbol{x}_{i\pm1,j} \in \Omega_I \text{ or } \boldsymbol{x}_{i,j\pm1} \in \Omega_I\}.$$

A corresponding formula is used to approximate the Laplacian of w at all interior grid points of Ω_{II} using a corresponding set of ghost points G_{II}.

The solution at the ghost points are determined by the jump conditions (5) and (6). We start by considering the ghost point $u_{i,j}$; see Figure 1.

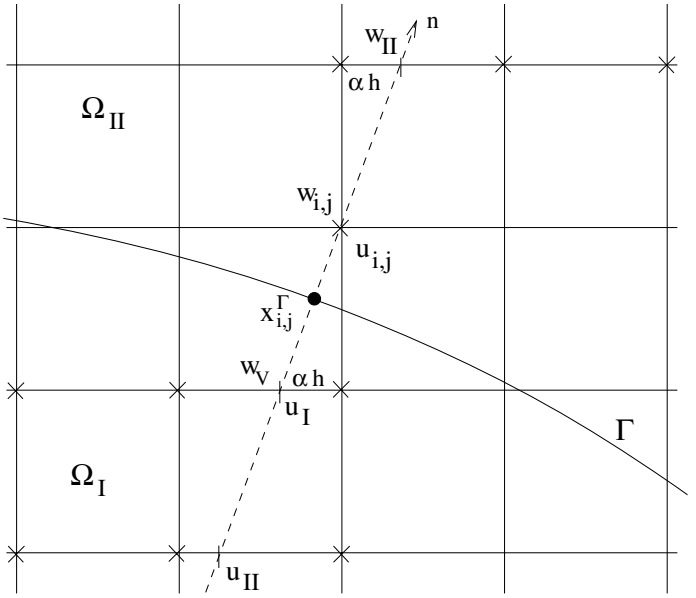


FIG. 1. *The points used for discretizing the jump conditions.*

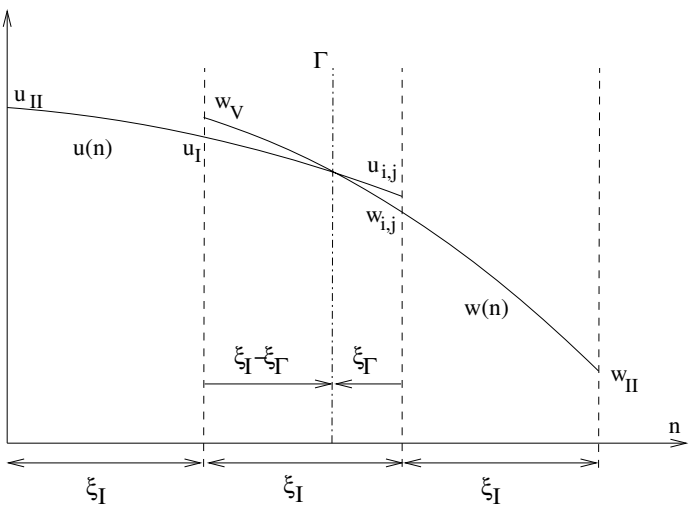


FIG. 2. *The solution along the normal to the interface is continuous, but its first derivative is discontinuous.*

We can derive second order approximations for the value and normal derivative on the boundary using Lagrange interpolation between $u_{i,j}$, u_I, and u_{II} (see Figure 2):

$$B^D_{i,j}u =: g^D_0(\xi_\Gamma)u_{i,j} + g^D_I(\xi_\Gamma)u_I + g^D_{II}(\xi_\Gamma)u_{II} = u(\boldsymbol{x}^\Gamma_{i,j}) + \mathcal{O}(h^2), \qquad (i,j) \in G_I, \tag{8}$$

$$B^N_{i,j}u =: g^N_0(\xi_\Gamma)u_{i,j} + g^N_I(\xi_\Gamma)u_I + g^N_{II}(\xi_\Gamma)u_{II} = \frac{\partial u}{\partial n}(\boldsymbol{x}^\Gamma_{i,j}) + \mathcal{O}(h^2), \quad (i,j) \in G_I. \tag{9}$$

The values u_I and u_{II} are obtained by Lagrange interpolation, in this case, along the

horizontal grid lines y_{j-1} and y_{j-2},

$$u_I^n = r_0 u_{i,j-1}^n + r_1 u_{i-1,j-1}^n + r_2 u_{i-2,j-1}^n, \tag{10}$$

$$u_{II}^n = r_3 u_{i,j-2}^n + r_4 u_{i-1,j-2}^n + r_5 u_{i-2,j-2}^n. \tag{11}$$

The coefficients g^D and g^N are given in section 4.

To derive corresponding formulas for w along the same normal, we introduce the virtual ghost point value w_v, defined at the same location as u_I; see Figure 1. Since w_v is outside Ω_{II} and it is in general located in between grid points, this value is only used to form the discrete jump conditions. Lagrange interpolation between w_v, $w_{i,j}$, and w_{II} (see Figure 2) gives for $(i,j) \in G_I$

$$\begin{aligned} V_{i,j}^D w &=: g_0^D(\xi_I - \xi_\Gamma) w_v + g_I^D(\xi_I - \xi_\Gamma) w_{i,j} + g_{II}^D(\xi_I - \xi_\Gamma) w_{II} \\ &= w(\boldsymbol{x}_{i,j}^\Gamma) + \mathcal{O}(h^2), \end{aligned} \tag{12}$$

$$\begin{aligned} V_{i,j}^N w &=: g_0^N(\xi_I - \xi_\Gamma) w_v + g_I^N(\xi_I - \xi_\Gamma) w_{i,j} + g_{II}^N(\xi_I - \xi_\Gamma) w_{II} \\ &= \frac{\partial w}{\partial n}(\boldsymbol{x}_{i,j}^\Gamma) + \mathcal{O}(h^2). \end{aligned} \tag{13}$$

Note that there is a symmetry between the interpolation formulas for u_I and w_{II} that simplifies the calculation of w_{II}. If the normal going through $\boldsymbol{x}_{i,j}$ intersects the horizontal grid line $y = y_{j-1}$ at $x = x_i - \alpha h$, the extension of the same normal will intersect the grid line $y = y_{j+1}$ at $x = x_i + \alpha h$. The Lagrange interpolation formula for w_{II}^n therefore becomes

$$w_{II}^n = r_0 w_{i,j+1}^n + r_1 w_{i+1,j+1}^n + r_2 w_{i+2,j+1}^n.$$

Inserting the discrete approximations of the value and normal derivatives of u and w into the jump conditions (5), (6) results in the discretized jump conditions

$$B_{i,j}^D u^n = V_{i,j}^D w^n, \tag{14}$$

$$\frac{1}{\rho_I} B_{i,j}^N u^n = -\frac{1}{\rho_{II}} V_{i,j}^N w^n. \tag{15}$$

Hence, we get a 2×2 linear system for the unknowns $(u_{i,j}^n, w_v^n)$:

$$\begin{aligned} &\begin{pmatrix} g_0^D(\xi_\Gamma) & -g_0^D(\xi_I - \xi_\Gamma) \\ g_0^N(\xi_\Gamma)/\rho_I & g_0^N(\xi_I - \xi_\Gamma)/\rho_{II} \end{pmatrix} \begin{pmatrix} u_{i,j}^n \\ w_v^n \end{pmatrix} \\ &= \begin{pmatrix} g_I^D(\xi_I - \xi_\Gamma) w_{i,j}^n + g_{II}^D(\xi_I - \xi_\Gamma) w_{II}^n - g_I^D(\xi_\Gamma) u_I^n - g_{II}^D(\xi_\Gamma) u_{II}^n \\ -(g_I^N(\xi_I - \xi_\Gamma) w_{i,j}^n + g_{II}^N(\xi_I - \xi_\Gamma) w_{II}^n)/\rho_{II} - (g_I^N(\xi_\Gamma) u_I^n + g_{II}^N(\xi_\Gamma) u_{II}^n)/\rho_I \end{pmatrix}. \end{aligned} \tag{16}$$

We solve this system for $u_{i,j}^n$ and ignore the virtual ghost point value w_v^n. A formula for the w-ghost points can be derived in a similar way.

Note that this discretization of the jump conditions has the desirable property that each ghost point can be updated independently of all other ghost points. Hence, there is no coupling along the boundary.

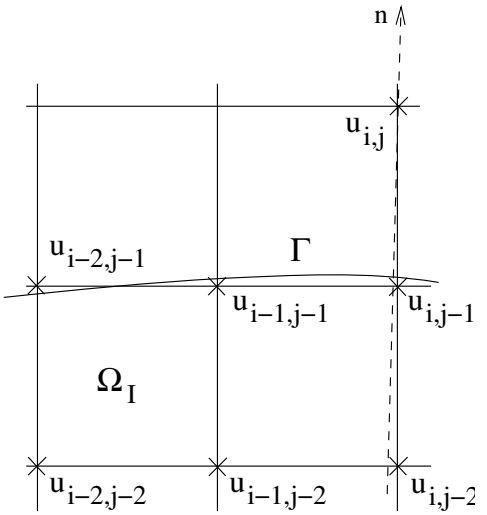


FIG. 3. *The Lagrange interpolation formula for u_I can in rare cases contain outside points.*

Our practical experience with the above interpolation procedure from previous work [2, 3] and the numerical examples in section 6 indicate that it is almost always possible to find enough interior points to compute u_I, u_{II}, and w_{II}, as long as the interface Γ is well resolved on the grid. However, there are rare cases when one or two of the points in the interpolation stencil for u_I can be outside the domain; see Figure 3. In this case, the interface curve has a local maxima just above y_{j-1}, between x_{i-1} and x_i, such that the normal going through $u_{i,j}$ has the form $\boldsymbol{n} = (\zeta, 1-\zeta^2/2)^T$, $\zeta > 0$, $\zeta \ll 1$. Since $\zeta > 0$, we choose to interpolate from $u_{i-2,j-1}$, $u_{i-1,j-1}$, and $u_{i,j-1}$ to obtain u_I. In this situation it is possible for $u_{i-2,j-1}$ to be outside the domain Ω_I, even though the interface is well resolved on the grid. Since the interpolation formula for updating $u_{i,j}$ contains an outside point, we call $u_{i,j}$ a coupled ghost point. However, $u_{i-2,j-1}$ is itself a ghost point which uses only interior points in its interpolation stencils. Such points are called uncoupled ghost points. Hence, if we first compute the uncoupled ghost point value $u_{i-2,j-1}$ we will have all values necessary for computing the coupled ghost point value $u_{i,j}$. In all smooth interface cases we have tried, any coupled ghost points have depended only on uncoupled ghost points, allowing us to use a pointwise procedure where we first update all uncoupled ghost points, followed by updating any coupled ghost points. Some attempts to treat boundaries with corners with the embedded boundary method were reported in [2]. However, the solution can exhibit singularities at the corners which present a more severe set of challenges than merely finding enough grid points to interpolate from. In the following we assume that the interface is smooth and well resolved on the grid.

The remainder of the paper is organized as follows. In section 2, we prove that the discretization of the jump conditions is stable in the one-dimensional case. A simplified stability argument, based on a modified equation model, is presented in section 3. The solvability of the discrete jump conditions in the two-dimensional setting is demonstrated in section 4. Section 5 discusses time-integration and a generalization of the A^TA-dissipation operator (see [2]), which is used to stabilize the scheme for long time-integrations. In section 6, we use the method to study electromagnetic scattering of a plane wave by a dielectric cylinder, where there is a century-old analytical

solution due to Mie [27]. The method is finally applied to two more complicated scattering problems: one including many subdomains and the other with a more irregular interface.

2. Stability. We start by considering the one-dimensional wave equation with discontinuous wave propagation speed,

$$u_{tt} = u_{xx}, \qquad x_{min} \le x \le 0, \quad t \ge 0, \tag{17}$$

$$w_{tt} = c^2 w_{xx}, \quad 0 \le x \le x_{max}, \quad t \ge 0, \tag{18}$$

$$u(x,0) = U_0(x), \quad u_t(x,0) = U_1(x), \qquad x_{min} \le x \le 0, \tag{19}$$

$$w(x,0) = W_0(x), \quad w_t(x,0) = W_1(x), \qquad 0 \le x \le x_{max}, \tag{20}$$

where $c^2 > 0$, $x_{min} < 0$, and $x_{max} > 0$, subject to the Dirichlet boundary conditions

$$u(x_{min}, t) = 0, \quad w(x_{max}, t) = 0, \quad t \ge 0. \tag{21}$$

At the interface, the jump conditions are

$$u(0,t) = w(0,t),$$

$$u_x(0,t) = c^2 w_x(0,t).$$

Using the notation in Figure 4, we discretize the problem in space on a uniform grid $x_\nu = -\alpha h + \nu h$, with grid size $h > 0$, and denote a grid function by $u_\nu(t) = u(x_\nu, t)$. Divided difference operators are defined by $D_+^x u_\nu = (u_{\nu+1} - u_\nu)/h$ and $D_-^x u_\nu = D_+^x u_{\nu-1}$. To model the two-dimensional case, we shift the grid so the discontinuity in the wave speed does not coincide with a grid point, i.e., $0 < \alpha < 1$. We want to focus the discussion on the stability of the discretization of the jump conditions, and we take x_{min} and x_{max} to coincide with grid points, i.e., $x_{min} = x_{-N}$ and $x_{max} = x_N$.

The semidiscrete problem becomes

$$\frac{d^2 u_\nu}{dt^2} = D_+^x D_-^x u_\nu, \qquad \nu = 0, -1, -2, \ldots, -N+1, \tag{22}$$

$$\frac{d^2 w_\nu}{dt^2} = c^2 D_+^x D_-^x w_\nu, \qquad \nu = 1, 2, 3, \ldots, N-1, \tag{23}$$

subject to Dirichlet boundary conditions

$$u_{-N} = 0, \quad w_N = 0, \quad t \ge 0. \tag{24}$$

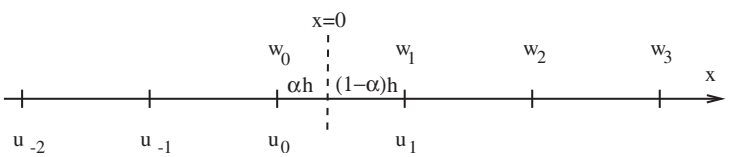


FIG. 4. *Notation for the one-dimensional problem.*

We discretize the jump conditions to second order accuracy by

$$(1-\alpha)u_0 + \alpha u_1 = (1-\alpha)w_0 + \alpha w_1, \tag{25}$$

$$D_+^x u_0 + h\left(\alpha - \frac{1}{2}\right) D_+^x D_-^x u_0 = c^2 \left(D_+^x w_0 + h\left(\alpha - \frac{1}{2}\right) D_+^x D_-^x w_1 \right). \tag{26}$$

By using (22) and (23), we can write (25), (26) as

$$\alpha u_1 + (\alpha - 1) w_0 = (\alpha - 1) u_0 + \alpha w_1,$$

$$u_1 + c^2 w_0 = u_0 + c^2 w_1 + h^2 \left(\alpha - \frac{1}{2}\right) \left(\frac{d^2 w_1}{dt^2} - \frac{d^2 u_0}{dt^2} \right).$$

Solving for w_0 and u_1 yields

$$w_0 = \frac{1}{1-\alpha(1-c^2)} \left(u_0 - \alpha(1-c^2) w_1 + h^2 \alpha \left(\alpha - \frac{1}{2}\right) \left(\frac{d^2 w_1}{dt^2} - \frac{d^2 u_0}{dt^2} \right) \right), \tag{27}$$

$$u_1 = \frac{1}{1-\alpha(1-c^2)} \Bigg(c^2 w_1 + (1-\alpha)(1-c^2) u_0 + h^2 (1-\alpha) \left(\alpha - \frac{1}{2}\right) \left(\frac{d^2 w_1}{dt^2} - \frac{d^2 u_0}{dt^2} \right) \Bigg). \tag{28}$$

By using (27)–(28) we can eliminate w_0, u_1 from (22)–(23) and obtain

$$b_{11} \frac{d^2 u_0}{dt^2} + b_{12} \frac{d^2 w_1}{dt^2} = \frac{1}{h^2} \left(a_{10} u_{-1} + a_{11} u_0 + a_{12} w_1 \right),$$

$$b_{21} \frac{d^2 u_0}{dt^2} + b_{22} \frac{d^2 w_1}{dt^2} = \frac{1}{h^2} \left(a_{21} u_0 + a_{22} w_1 + a_{23} w_2 \right).$$

Here,

$$b_{11} = 1 - \alpha(1-c^2) + \left(\alpha - \frac{1}{2}\right)(1-\alpha), \quad b_{12} = -\left(\alpha - \frac{1}{2}\right)(1-\alpha), \tag{29}$$

$$b_{21} = \alpha \left(\alpha - \frac{1}{2}\right), \quad b_{22} = \frac{1}{c^2} \left(1 - \alpha(1-c^2) - c^2 \alpha \left(\alpha - \frac{1}{2}\right) \right) \tag{30}$$

and

$$a_{10} = 1 - \alpha(1-c^2), \quad a_{11} = -2 + (1+\alpha)(1-c^2), \quad a_{12} = c^2, \tag{31}$$

$$a_{21} = 1, \quad a_{22} = -2 + \alpha(1-c^2), \quad a_{23} = 1 - \alpha(1-c^2). \tag{32}$$

We can write the semidiscrete problem in matrix form,

$$\begin{pmatrix} 1 & 0 & & & & & & & \\ & \ddots & & & & & & & \\ & 0 & 1 & 0 & & & & & \\ & & 0 & b_{11} & b_{12} & & & & \\ & & & b_{21} & b_{22} & 0 & & & \\ & & & & 0 & 1/c^2 & 0 & & \\ & & & & & & \ddots & & \\ & & & & & & 0 & 1/c^2 \end{pmatrix} \frac{d^2}{dt^2} \begin{pmatrix} u_{-N+1} \\ \vdots \\ u_{-1} \\ u_0 \\ w_1 \\ w_2 \\ \vdots \\ w_{N-1} \end{pmatrix}$$

$$= \frac{1}{h^2} \begin{pmatrix} -2 & 1 & & & & & & \\ & \ddots & & & & & & \\ & 1 & -2 & 1 & & & & \\ & & a_{10} & a_{11} & a_{12} & & & \\ & & & a_{21} & a_{22} & a_{23} & & \\ & & & & 1 & -2 & 1 & \\ & & & & & \ddots & & \\ & & & & & 1 & -2 \end{pmatrix} \begin{pmatrix} u_{-N+1} \\ \vdots \\ u_{-1} \\ u_0 \\ w_1 \\ w_2 \\ \vdots \\ w_{N-1} \end{pmatrix},$$

which we write as

$$\mathcal{B}\mathbf{w}_{tt} = \frac{1}{h^2}\mathcal{A}\mathbf{w}. \tag{33}$$

By (31), (32), the matrix $\mathcal{A}$ is negative diagonally dominant and the off-diagonal elements are positive. Therefore it can be symmetrized by a diagonal scaling $\mathcal{D} > 0$,

$$\tilde{\mathbf{w}} = \begin{pmatrix} \ddots & & & & & & & \\ & d_{-2} & & & & & & \\ & & d_{-1} & & & & 0 & \\ & & & d_0 & & & & \\ & 0 & & & d_1 & & & \\ & & & & & d_2 & & \\ & & & & & & \ddots & \end{pmatrix} \begin{pmatrix} \vdots \\ u_{-2} \\ u_{-1} \\ u_0 \\ w_1 \\ w_2 \\ \vdots \end{pmatrix} =: \mathcal{D}\mathbf{w}, \tag{34}$$

and (33) becomes

$$\tilde{\mathcal{B}}\tilde{\mathbf{w}}_{tt} = \frac{1}{h^2}\tilde{\mathcal{A}}\tilde{\mathbf{w}}, \quad \tilde{\mathcal{B}} = \mathcal{D}\mathcal{B}\mathcal{D}^{-1}, \quad \tilde{\mathcal{A}} = \mathcal{D}\mathcal{A}\mathcal{D}^{-1} = \tilde{\mathcal{A}}^* < 0. \tag{35}$$

Without restriction we can assume that $d_0 = 1$. Then d_1 is determined by the condition that

$$\begin{pmatrix} 1 & 0 \\ 0 & d_1 \end{pmatrix} \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & d_1^{-1} \end{pmatrix} = \begin{pmatrix} a_{11} & d_1^{-1}a_{12} \\ d_1 a_{21} & a_{22} \end{pmatrix}$$

is symmetric. By (31), (32), $d_1 = \sqrt{a_{12}/a_{21}} = c$. The corresponding 2×2 submatrix of $\mathcal{B}$ becomes

$$\tilde{\mathcal{B}}_1 = \begin{pmatrix} 1 & 0 \\ 0 & d_1 \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & d_1^{-1} \end{pmatrix} = \begin{pmatrix} b_{11} & b_{12}/c \\ cb_{21} & b_{22} \end{pmatrix}.$$

We want to show that

$$\frac{1}{2}(\tilde{\mathcal{B}} + \tilde{\mathcal{B}}^*) > 0, \quad \tilde{\mathcal{B}} = \mathcal{D}\mathcal{B}\mathcal{D}^{-1}. \tag{36}$$

We need to prove this only for $\tilde{\mathcal{B}}_1$, since the remaining part of $\tilde{\mathcal{B}}$ is identical to $\mathcal{B}$, i.e., diagonal with positive elements. By (29), (30),

$$b_{11} = (1-\alpha)\left(\frac{1}{2} + \alpha\right) + \alpha c^2 > 0, \tag{37}$$

$$b_{22} = \frac{1}{c^2}\left(1 - \alpha + \alpha c^2 \left(\frac{3}{2} - \alpha\right)\right) > 0. \tag{38}$$

The characteristic equation, $\det(\frac{1}{2}(\tilde{\mathcal{B}}_1 + \tilde{\mathcal{B}}_1^*) - \mu I) = 0$, is

$$\mu^2 - \mu(b_{11} + b_{22}) + b_{11}b_{22} - \frac{1}{4}\left(\frac{b_{12}}{c} + cb_{21}\right)^2 = 0,$$

with discriminant

$$\Delta = (b_{11} + b_{22})^2 - 4\left(b_{11}b_{22} - \frac{1}{4}\left(\frac{b_{12}}{c} + cb_{21}\right)^2\right) = (b_{11} - b_{22})^2 + \left(\frac{b_{12}}{c} + cb_{21}\right)^2 > 0.$$

The roots are

$$\mu_{1,2} = \frac{1}{2}\left(b_{11} + b_{22} \pm \sqrt{\Delta}\right).$$

Since $b_{11} + b_{22} > 0$ and $\Delta > 0$, both roots are real and positive if $(b_{11} + b_{22})^2 > \Delta$. We have

$$\frac{1}{4}\left((b_{11} + b_{22})^2 - \Delta\right) = b_{11}b_{22} - \frac{1}{4}\left(\frac{b_{12}}{c} + cb_{21}\right)^2$$

$$= \frac{1}{c^2}\left((1-\alpha)\left(\frac{1}{2} + \alpha\right) + \alpha c^2\right)\left(1 - \alpha + \alpha c^2\left(\frac{3}{2} - \alpha\right)\right) - \frac{\left(\alpha - \frac{1}{2}\right)^2}{4}(1 - \alpha - \alpha c^2)^2$$

$$= \frac{1}{c^2}\left((1-\alpha)^2\left(\frac{1}{2} + \alpha\right) - \frac{(\alpha - \frac{1}{2})^2}{4}(1-\alpha)^2\right) > 0.$$

Thus $\frac{1}{2}(\tilde{\mathcal{B}} + \tilde{\mathcal{B}}^*)$ is positive definite.

If $\alpha = \frac{1}{2}$, then, by (29)–(30), $b_{12} = b_{21} = 0$ and $\tilde{\mathcal{B}} = \tilde{\mathcal{B}}^* > 0$. Since $\tilde{\mathcal{A}} = \tilde{\mathcal{A}}^* < 0$, there is an energy estimate and the method is stable.

If $\alpha \neq \frac{1}{2}$, then we make an eigenvector expansion. $\tilde{\mathbf{w}} = e^{\lambda t}\tilde{\mathbf{w}}_0$ is a solution of (35) if $\lambda, \tilde{\mathbf{w}}_0$ are solutions of the eigenvalue problem

$$\left(\lambda^2 \tilde{\mathcal{B}} - \frac{1}{h^2}\tilde{\mathcal{A}}\right)\tilde{\mathbf{w}}_0 = 0. \tag{39}$$

If we can prove that the eigenvalues λ^2 are real, distinct, and negative, then there are no growing modes and the approximation is stable.

If λ^2 is real, then $\tilde{\mathbf{w}}_0$ is also real and the symmetry of $\tilde{\mathcal{A}}$ gives us

$$\lambda^2\langle \tilde{\mathbf{w}}_0, \tilde{\mathcal{B}}\tilde{\mathbf{w}}_0\rangle = \frac{\lambda^2}{2}\langle \tilde{\mathbf{w}}_0, (\tilde{\mathcal{B}} + \tilde{\mathcal{B}}^*)\tilde{\mathbf{w}}_0\rangle = \frac{1}{h^2}\langle \tilde{\mathbf{w}}_0, \tilde{\mathcal{A}}\tilde{\mathbf{w}}_0\rangle,$$

where $\langle \mathbf{u}, \mathbf{v}\rangle$ denotes the usual L_2 inner product. Since $\tilde{\mathcal{A}} < 0$ and $\tilde{\mathcal{B}} + \tilde{\mathcal{B}}^* > 0$, it follows that $\lambda^2 < 0$.

We need the following lemma.

LEMMA 1. [illegible]

$$\begin{pmatrix} d_1 & e_1 & 0 & \cdots & \cdots & \cdots & 0 \\ l_2 & d_2 & e_2 & 0 & \cdots & \cdots & 0 \\ 0 & l_3 & d_3 & e_3 & 0 & \cdots & 0 \\ 0 & & & & 0 & l_n & d_n \end{pmatrix} \begin{pmatrix} z_1 \\ \\ \\ z_n \end{pmatrix} = 0, \tag{40}$$

[illegible] $\theta = 0$ [illegible] $e_j \neq 0$ [illegible] $l_j \neq 0$, [illegible] θ [illegible] (proof [illegible])

Assume that all $e_j \neq 0$ and θ is not a simple eigenvalue. Since the invariant subspace corresponding to θ has dimension larger than one, we can construct an eigenvector with $z_1 = 0$. But then $z_2 = z_3 = \cdots = z_n = 0$, which is a contradiction. Correspondingly, if all $l_j \neq 0$, then there is a solution with $z_n = 0$ which again implies that $z_{n-1} = z_{n-2} = \cdots = z_1 = 0$. This proves the lemma. □

We shall now prove that the eigenvalues λ^2 of (39) are simple for all $0 < \alpha < 1$. λ^2 is an eigenvalue of (39) if zero is an eigenvalue of the tridiagonal matrix $Q = h^2\lambda^2\tilde{\mathcal{B}} - \tilde{\mathcal{A}}$. The conditions of Lemma 1 are violated if there is some value of α where at least one element on the subdiagonal of Q is zero [illegible] one element on the superdiagonal of Q is zero. The only off-diagonal elements of $\tilde{\mathcal{B}}$ are $\tilde{b}_{12}$ and $\tilde{b}_{21}$, $\mathcal{D} > 0$, and $a_{10} = a_{23} = (1-\alpha) + \alpha c^2 > 0$ for all $0 \leq \alpha \leq 1$. Hence, Lemma 1 can only be violated if

$$h^2\lambda^2\tilde{b}_{12} - \tilde{a}_{12} = 0 \quad \text{and} \quad h^2\lambda^2\tilde{b}_{21} - \tilde{a}_{21} = 0. \tag{41}$$

By (29), (30), (31), and (32) we can write (41) as

$$-h^2\lambda^2\left(\alpha - \frac{1}{2}\right)(1-\alpha) - c^2 = 0, \quad h^2\lambda^2\alpha\left(\alpha - \frac{1}{2}\right) - 1 = 0,$$

i.e.,

$$\frac{\alpha - 1}{\alpha} = c^2,$$

which is a contradiction because $c^2 > 0$, but the left-hand side is negative for $0 < \alpha < 1$. Thus the eigenvalues λ^2 of (39) are simple. They are also real because they are solutions of the characteristic equation and smooth functions of α. For $\alpha = \frac{1}{2}$ they are real. Therefore they are real for all values of α because they can only become complex at some value $\alpha = \alpha_0$ if there is an eigenvalue which is not simple. This completes the proof of the following theorem.

THEOREM 1. [illegible] λ^2 [illegible] (39) [illegible] (22), (23) [illegible] (24) [illegible] (25), (26) [illegible]

3. The modified equation. Instead of the complete proof of the last section (which is quite complicated), we shall now use modified equations and show stability for the low and intermediate frequencies. Note that this analysis does not provide any information about the highest frequencies and therefore does not prove stability of the scheme. As we shall see in section 6, numerical calculations show that our A^TA-dissipation takes care of any instabilities caused by the highest frequencies in our scheme. We discuss the modified equation technique here because in the multi-dimensional case, the use of modified equations is the only way to obtain stability information. Such an analysis can be found in [2]. Since in [2] we neglected the truncation error in the normal direction, it is important that the restriction of the approximation to one dimension is completely stable.

Even though extensive numerical experiments with our two-dimensional scheme indicate that it is stable with A^TA-dissipation, we note that there might be other schemes with stable modified equations which are unstable.

We introduce the dependent variable

$$v(x,t) = u(-x,t),$$

assume $x_{min} \to \infty$, $x_{max} \to \infty$, and write (17), (18) as a half-plane problem for the system

$$\begin{pmatrix} v \\ w \end{pmatrix}_{tt} = \begin{pmatrix} 1 & 0 \\ 0 & c^2 \end{pmatrix} \begin{pmatrix} v \\ w \end{pmatrix}_{xx}, \quad 0 \le x < \infty, \quad t \ge 0, \tag{42}$$

with the modified jump conditions

$$v(0,t) + h^2\beta v_{xx}(0,t) = w(0,t) + h^2\beta w_{xx}(0,t), \tag{43}$$

$$v_x(0,t) + h^2\gamma v_{xxx}(0,t) = -c^2\left(w_x(0,t) + h^2\gamma w_{xxx}(0,t)\right). \tag{44}$$

We obtain (43) and (44) from (25) and (26) by adding the leading truncation error term. A simple but tedious calculation shows that

$$\beta = \frac{(1-\alpha)\alpha}{2}, \quad \gamma = \frac{-2+6\alpha-3\alpha^2}{6}.$$

We use mode analysis to discuss stability. The general solutions of type

$$\begin{pmatrix} v(x,t) \\ w(x,t) \end{pmatrix} = e^{st} \begin{pmatrix} \tilde{v}(x,s) \\ \tilde{w}(x,s) \end{pmatrix}, \quad \mathrm{Re}(s) \ge 0,$$

where $(\tilde{v}(x,s), \tilde{w}(x,s))$ are bounded, are given by

$$v(x,t) = e^{st}e^{-sx}v_0, \quad w(x,t) = e^{st}e^{-(s/c)x}w_0. \tag{45}$$

Introducing (45) into the boundary conditions (43), (44) gives us

$$\begin{aligned}(1+h^2\beta s^2)v_0 - (1+h^2\beta s^2)w_0 &= 0,\\ (1+h^2\gamma s^2)v_0 + c^2(1+h^2\gamma s^2)w_0 &= 0.\end{aligned}$$

Thus, nontrivial solutions exist if and only if

$$-(1+h^2\beta s^2)(1+h^2\gamma s^2) = 0.$$

Hence, instabilities can only be present for $s = \mathcal{O}(1/h)$, i.e., for high frequencies. (However, the proof in the previous section shows that this cannot happen.)

4. Solvability of the two-dimensional discrete jump conditions. Let the 2×2 matrix on the left-hand side of (16) be P. The linear system has a unique solution if $\det P \neq 0$. We have

$$\det P = g_0^D(\xi_\Gamma)g_0^N(\xi_I - \xi_\Gamma)/\rho_{II} + g_0^D(\xi_I - \xi_\Gamma)g_0^N(\xi_\Gamma)/\rho_I.$$

The coefficients in the Dirichlet formula (8) are

$$g_0^D(\xi) = \frac{(\xi_I - \xi)(2\xi_I - \xi)}{2\xi_I^2} + \delta, \quad g_I^D(\xi) = \frac{\xi(2\xi_I - \xi)}{\xi_I^2} - 2\delta, \quad g_{II}^D(\xi) = \frac{\xi(\xi - \xi_I)}{2\xi_I^2} + \delta,$$

where $\delta \approx 0.25$ is a constant that removes the small-cell time step restriction while preserving the second order accuracy; see [3]. The coefficients in the Neumann formula (9) are (see [2])

$$g_0^N(\xi) = \frac{3\xi_I - 2\xi}{2\xi_I^2}, \quad g_I^N(\xi) = \frac{2\xi - 2\xi_I}{\xi_I^2}, \quad g_{II}^N(\xi) = \frac{\xi_I - 2\xi}{2\xi_I^2}.$$

Since $0 \le \xi_\Gamma \le \xi_I$ and $\delta > 0$,

$$0 < \delta \le g_0^D \le 1 + \delta, \quad 0 < \frac{1}{2\xi_I} \le g_0^N \le \frac{3}{2\xi_I}.$$

Hence

$$\frac{\delta}{2\xi_I}\left(\frac{1}{\rho_I} + \frac{1}{\rho_{II}}\right) \le \det P \le \frac{3(1+\delta)}{2\xi_I}\left(\frac{1}{\rho_I} + \frac{1}{\rho_{II}}\right).$$

For all possible directions of the normal, $h \le \xi_I \le \sqrt{2}h$, and we conclude that

$$\det P \ge C > 0, \quad C = \frac{\delta}{2\sqrt{2}h}\left(\frac{1}{\rho_I} + \frac{1}{\rho_{II}}\right).$$

The fact that the lower bound of $\det P$ is proportional to δ shows that the δ-term is essential for the solvability of the discrete jump conditions for general locations of the interface relative to the grid. The δ-term was originally designed for Dirichlet boundary conditions to remove the small-cell time step restriction (cf. [3]), and it also serves that purpose here.

5. Time-integration and A^TA-dissipation. The discretized jump conditions (16) can in principle be used to eliminate all ghost points from the discrete approximation of the Laplacians of u and w (7). Since the jump conditions couple the solutions on both sides of the interface, the matrix form of the discrete Laplacian becomes

$$\begin{aligned} \Delta_h u(\boldsymbol{X}_I, t_n) &= A_{11}\boldsymbol{u}^n + A_{12}\boldsymbol{w}^n, \\ \Delta_h w(\boldsymbol{X}_{II}, t_n) &= A_{21}\boldsymbol{u}^n + A_{22}\boldsymbol{w}^n, \end{aligned} \qquad A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}.$$

Here, $\boldsymbol{X}_I$ and $\boldsymbol{X}_{II}$ are vectors of all grid point coordinates inside Ω_I and Ω_{II}, respectively, and $\boldsymbol{u}$ and $\boldsymbol{w}$ are the discrete solutions at those grid points. If all eigenvalues of A are distinct, real, and negative, one can show that the scheme

$$\begin{pmatrix} \rho_I(\boldsymbol{u}^{n+1} - 2\boldsymbol{u}^n + \boldsymbol{u}^{n-1})/\delta_t^2 \\ \rho_{II}(\boldsymbol{w}^{n+1} - 2\boldsymbol{w}^n + \boldsymbol{w}^{n-1})/\delta_t^2 \end{pmatrix} = A \begin{pmatrix} \boldsymbol{u}^n \\ \boldsymbol{w}^n \end{pmatrix} + \boldsymbol{F}(t_n) \tag{46}$$

is stable for sufficiently small time steps δ_t [1]. However, the embedded boundary approximation of the jump conditions breaks the symmetry of the matrix A so it is not possible to guarantee this property. In fact, numerical examples (see section 6) indicate that the above scheme suffers from a mild instability.

In the previous embedded boundary methods for the wave equation subject to Neumann [2] and Dirichlet [3] boundary conditions, we damped the instability by adding a small fourth order term of the type $h^3 A^T A(\boldsymbol{u}_t, \boldsymbol{w}_t)^T$ to the right-hand side of (46). While this technique turned out to work very well in practice for the Neumann and Dirichlet problems, it is not directly amendable to the current problem because the matrix A couples the solutions across the interface. Consequently, it becomes complicated to evaluate A^T on a solution vector without explicitly forming the matrix A.

It is not difficult to modify the previous dissipation technique to work with the jump conditions. We first describe the idea for the continuous problem. For the wave equation (1)–(2) with jump conditions (5)–(6), we can evaluate the normal derivative of the solutions on either side of the interface as functions of time,

$$\frac{\partial u}{\partial n}(\boldsymbol{x}, t) =: f_I(\boldsymbol{x}, t), \quad \boldsymbol{x} \in \Gamma,\ t \geq 0, \tag{47}$$

$$\frac{\partial w}{\partial n}(\boldsymbol{x}, t) =: f_{II}(\boldsymbol{x}, t), \quad \boldsymbol{x} \in \Gamma,\ t \geq 0. \tag{48}$$

Hence, once the continuous problem with jump conditions has been solved, we can in principle recompute the same solution by solving two uncoupled Neumann problems:

$$\begin{aligned} \rho_I u_{tt} &= \Delta u + F(\boldsymbol{x}, t), && \boldsymbol{x} \in \Omega_I, && t \geq 0, \\ \frac{\partial u}{\partial n}(\boldsymbol{x}, t) &= f_I(\boldsymbol{x}, t), && \boldsymbol{x} \in \Gamma, && t \geq 0, \\ u(\boldsymbol{x}, 0) = U_0(\boldsymbol{x}), \quad u_t(\boldsymbol{x}, 0) &= U_1(\boldsymbol{x}), && && \boldsymbol{x} \in \Omega_I, \end{aligned} \tag{49}$$

and

$$\begin{aligned} \rho_{II} w_{tt} &= \Delta w + F(\boldsymbol{x}, t), && \boldsymbol{x} \in \Omega_{II}, && t \geq 0, \\ \frac{\partial w}{\partial n}(\boldsymbol{x}, t) &= f_{II}(\boldsymbol{x}, t), && \boldsymbol{x} \in \Gamma, && t \geq 0, \\ w(\boldsymbol{x}, 0) = W_0(\boldsymbol{x}), \quad w_t(\boldsymbol{x}, 0) &= W_1(\boldsymbol{x}), && && \boldsymbol{x} \in \Omega_{II}. \end{aligned} \tag{50}$$

Our basic idea is to use the stabilized embedded boundary scheme for the Neumann problem described in [2], where the forcing functions f_I and f_{II} are computed on the fly during the time evolution.

For conciseness, we describe only the details for the Neumann problem (49), which is discretized on the same Cartesian grid as above, leading to the same set of ghost points G_I where the discrete boundary conditions are applied,

$$B_{i,j}^N u^n = f_I(\boldsymbol{x}_{i,j}^\Gamma, t_n), \quad (i,j) \in G_I. \tag{51}$$

The formula (7) for discretizing the Laplacian of u is the same as before, but since the Neumann problem for u is decoupled from w, we now get the matrix representation

$$\Delta_h u(\boldsymbol{X}_I, t_n) = A_I \boldsymbol{u}^n + \boldsymbol{b}_I^n, \tag{52}$$

after all ghost points have been eliminated from (7). The vector $\boldsymbol{b}_I$ contains the discrete boundary forcing function corresponding to the forcing function $f_I(\boldsymbol{x}^\Gamma, t)$ and is only nonzero at grid points just inside the boundary; see [2] for details.

The stabilized scheme for the Neumann problem is

$$\begin{aligned}\rho_I(\boldsymbol{u}^{n+1} - 2\boldsymbol{u}^n + \boldsymbol{u}^{n-1})/\delta_t^2 &= A_I \boldsymbol{u}^n + \boldsymbol{b}^n + \boldsymbol{F}(t_n) \\ &\quad - \epsilon \frac{h^3}{\delta_t} A_I^T \left(A_I \boldsymbol{u}^n + \boldsymbol{b}^n - A_I \boldsymbol{u}^{n-1} - \boldsymbol{b}^{n-1} \right),\end{aligned} \tag{53}$$

where $\epsilon > 0$ is a small constant. Numerical experiments indicate that $\epsilon = \mathcal{O}(10^{-3})$ is sufficient to allow for very long time-integrations (10^6 time steps or more). The scheme (54) can be recast into an equivalent form that makes it easier to generalize to handle the jump conditions. Let us define an extended solution vector $\bar{\boldsymbol{u}}$ that includes the solution at all interior points as well as the ghost points. Given $\bar{\boldsymbol{u}}$, all values are defined to evaluate the discrete Laplacian $\Delta_h u_{i,j}$ (7) at all points $\boldsymbol{x}_{i,j} \in \Omega_I$. We write the discrete Laplacian at all these points as $\Delta_h \bar{\boldsymbol{u}}$ and arrive at the equivalent method:

1. Given $\boldsymbol{u}^n$, define $\bar{\boldsymbol{u}}^n$ by assigning all ghost points to satisfy the discretized Neumann boundary condition (51).
2. Update all interior points $\boldsymbol{x}_{i,j} \in \Omega_I$ by

$$\begin{aligned}\rho_I(\boldsymbol{u}^{n+1} - 2\boldsymbol{u}^n + \boldsymbol{u}^{n-1})/\delta_t^2 &= \Delta_h \bar{\boldsymbol{u}}^n + \boldsymbol{F}(t_n) \\ &\quad - \epsilon \frac{h^3}{\delta_t} A_I^T \left(\Delta_h \bar{\boldsymbol{u}}^n - \Delta_h \bar{\boldsymbol{u}}^{n-1} \right).\end{aligned} \tag{54}$$

Using this formulation shows that the matrix A_I is only needed to evaluate the dissipation term, and only for the matrix-vector product $A_I^T \boldsymbol{y}$, where $\boldsymbol{y} = \Delta_h \bar{\boldsymbol{u}}^n - \Delta_h \bar{\boldsymbol{u}}^{n-1}$. Note that the boundary forcing $f_I(\boldsymbol{x}^\Gamma, t)$ influences only the ghost point values and has no bearing on A_I^T.

To satisfy the discrete jump conditions (14)–(15), it appears that we must first calculate a corresponding value of f_I to use in (51). Given $\boldsymbol{u}^n$, this can be achieved by first calculating the values at all ghost points G_I by solving the 2×2 linear system (16). We can then use the embedded boundary formula (51) to evaluate f_I. However, it is a trivial exercise to see that enforcing (51) with this f_I results in the same ghost point values as those we started with. Hence, it is not necessary to calculate f_I, and we can simply replace the inhomogeneous Neumann condition by the discrete jump conditions (16). The same principle applies to the Neumann problem in Ω_{II} and we arrive at the damped scheme for the problem with jump conditions:

1. Given $\boldsymbol{u}^n$ and $\boldsymbol{w}^n$, define $\bar{\boldsymbol{u}}^n$ by assigning all ghost points $u_{i,j}$ to satisfy the discrete jump conditions along normals going through $(i,j) \in G_I$,

$$B^D_{i,j} u^n = V^D_{i,j} w^n,$$
$$\frac{1}{\rho_I} B^N_{i,j} u^n = -\frac{1}{\rho_{II}} V^N_{i,j} w^n.$$

2. Given $\boldsymbol{u}^n$ and $\boldsymbol{w}^n$, define $\bar{\boldsymbol{w}}^n$ by assigning all ghost points $w_{i,j}$ to satisfy the discrete jump conditions along normals going through $(i,j) \in G_{II}$,

$$B^D_{i,j} w^n = V^D_{i,j} u^n,$$
$$\frac{1}{\rho_{II}} B^N_{i,j} w^n = -\frac{1}{\rho_I} V^N_{i,j} u^n.$$

3. Update all interior points $\boldsymbol{x}_{i,j} \in \Omega_I$ by

$$\rho_I(\boldsymbol{u}^{n+1} - 2\boldsymbol{u}^n + \boldsymbol{u}^{n-1})/\delta_t^2 = \Delta_h \bar{\boldsymbol{u}}^n + \boldsymbol{F}(t_n) - \epsilon \frac{h^3}{\delta_t} A_I^T \left(\Delta_h \bar{\boldsymbol{u}}^n - \Delta_h \bar{\boldsymbol{u}}^{n-1}\right).$$

4. Update all interior points $\boldsymbol{x}_{i,j} \in \Omega_{II}$ by

$$\rho_{II}(\boldsymbol{w}^{n+1} - 2\boldsymbol{w}^n + \boldsymbol{w}^{n-1})/\delta_t^2 = \Delta_h \bar{\boldsymbol{w}}^n + \boldsymbol{F}(t_n) - \epsilon \frac{h^3}{\delta_t} A_{II}^T \left(\Delta_h \bar{\boldsymbol{w}}^n - \Delta_h \bar{\boldsymbol{w}}^{n-1}\right).$$

Here, A_{II} denotes the matrix representation of the discrete Laplacian subject to discrete Neumann conditions in Ω_{II}, and $\bar{\boldsymbol{w}}^n$ is the extended solution vector holding the solution at all interior grid points of Ω_{II} as well as at the ghost points G_{II}. Note that the matrices A_I and A_{II} do not need to be formed explicitly; cf. [2].

The wave equation subject to jump condition is equivalent to the two uncoupled wave equations subject to Neumann boundary conditions (49)–(50), and the theory in [2] shows that the $h^3 A^T A u_t$-dissipation term inflicts an error which is $\mathcal{O}(h^2)$ for these problems. Hence, we conclude that the dissipation term also inflicts an $\mathcal{O}(h^2)$ error for the wave equation with jump conditions.

6. Numerical examples. To test the accuracy of the numerical scheme, we begin by considering electromagnetic scattering of a plane incident wave by a dielectric circular cylinder of radius R. In this section, we follow the notation of electromagnetics and let $\kappa_e \geq 1$ denote the relative permittivity of the dielectric material but assume the same permeability inside and outside the dielectric. By assuming TE_z polarization and scaling time to obtain unit speed of light in vacuum, we arrive at the following problem for the z-component of the magnetic field:

$$\frac{\partial^2 H^{(z)}}{\partial t^2} = \Delta H^{(z)}, \qquad R^2 < x^2 + y^2 < \infty, \tag{55}$$

$$\frac{\partial^2 H^{(z)}}{\partial t^2} = \frac{1}{\kappa_e} \Delta H^{(z)}, \qquad x^2 + y^2 < R^2, \tag{56}$$

subject to the jump conditions

$$\left[H^{(z)}\right] = 0, \quad \left[\frac{1}{\kappa_e} \frac{\partial H^{(z)}}{\partial n}\right] = 0, \quad x^2 + y^2 = R^2.$$

Here, $\partial/\partial n$ denotes the normal derivative on the boundary of the dielectric cylinder and $\kappa_e = 1$ outside the cylinder. There is an analytical solution of this problem due

TABLE 1

Max error at time $T = 10.0$ for the case $\kappa_e = 2$ inside the dielectric cylinder (u_{err}) and outside of it (w_{err}), with A^TA-dissipation ($\epsilon = 10^{-3}$) and without dissipation.

N	h	u_{err}	u_{err}, A^TA	w_{err}	w_{err}, A^TA
201	1.5×10^{-2}	4.50×10^{-2}	4.47×10^{-2}	2.07×10^{-2}	2.09×10^{-2}
401	7.5×10^{-3}	1.12×10^{-2}	1.11×10^{-2}	5.44×10^{-3}	5.49×10^{-3}
801	3.75×10^{-3}	2.81×10^{-3}	2.80×10^{-3}	1.30×10^{-3}	1.31×10^{-3}

TABLE 2

Max error at time $T = 200.0$ for the case $\kappa_e = 2$ inside the dielectric cylinder (u_{err}) and outside of it (w_{err}). The coefficient in the A^TA-dissipation was $\epsilon = 10^{-3}$.

N	h	Time steps	u_{err}, A^TA	w_{err}, A^TA
201	1.5×10^{-2}	26,666	3.38×10^{-2}	3.57×10^{-2}
401	7.5×10^{-3}	53,333	8.42×10^{-3}	7.82×10^{-3}
801	3.75×10^{-3}	106,666	2.31×10^{-3}	1.97×10^{-3}

to Mie (see, for example, [28, p. 667]). Let the incident wave have angular frequency ω and wave number k:

$$H_I^{(z)}(x, y, t) = e^{i(kx-\omega t)}.$$

In polar coordinates (ρ, θ), the incident, scattered, and transmitted fields are

$$H_I^{(z)} = e^{i\omega t} \sum_{n=-\infty}^{\infty} i^{-n} J_n(k\rho) e^{in\theta}, \qquad \rho > R, \tag{57}$$

$$H_S^{(z)} = e^{i\omega t} \sum_{n=-\infty}^{\infty} i^{-n} a_n H_n^{(2)}(k\rho) e^{in\theta}, \quad \rho > R, \tag{58}$$

$$H_D^{(z)} = e^{i\omega t} \sum_{n=-\infty}^{\infty} i^{-n} b_n J_n(mk\rho) e^{in\theta}, \quad \rho < R, \tag{59}$$

where $m = \sqrt{\kappa_e}$, J_n is the Bessel function of the first kind of order n, and $H^{(2)}$ is the Hankel function of the second kind of order n (corresponding to waves propagating outward). The coefficients a_n and b_n are given in [28, p. 667].

The scattering problem was solved numerically for the case $R = 1$, $\kappa_e = 2$, $\omega = k = 2\pi$. The computational domain was the square $-1.5 \le x \le 1.5$, $-1.5 \le y \le 1.5$, and the exact solution was imposed as a Dirichlet boundary condition on the outer boundary. The exact solution was also imposed as initial condition. The error in the numerical solution measured in max norm over all internal grid points was evaluated at times $T = 10.0$ and $T = 200.0$; see Tables 1 and 2, respectively. Note that the error is second order accurate at both times and of the same order of magnitude for the same grid sizes. Studying the error as function of time (see Figure 5) reveals that the A^TA-dissipation is only necessary for long-time computations. Furthermore, there is no noticable growth of the error after long times when the dissipation is used, indicating that the damping is very weak.

As a second test, we consider the same geometry as above but reduce the wave speed inside the cylinder by setting $\kappa_e = 10$. Hence the wave length inside the cylinder

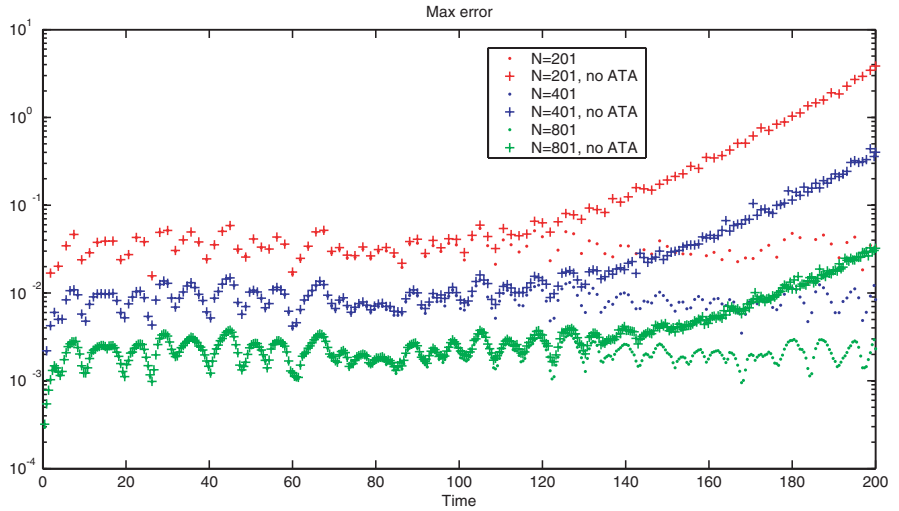


FIG. 5. *The max error inside the dielectric cylinder (u_{err}) as function of time for the case $\kappa_e = 2$, for different grid sizes, with and without A^TA-dissipation.*

TABLE 3

Max error at time $T = 10.0$ for the case $\kappa_e = 10$ inside the dielectric cylinder (u_{err}) and outside of it (w_{err}), with A^TA-dissipation ($\epsilon = 10^{-3}$) and without dissipation.

N	h	u_{err}	u_{err}, A^TA	w_{err}	w_{err}, A^TA
401	7.5×10^{-3}	2.17×10^{-1}	2.16×10^{-1}	6.52×10^{-2}	6.48×10^{-2}
801	3.75×10^{-3}	5.48×10^{-2}	5.47×10^{-2}	1.62×10^{-2}	1.61×10^{-2}

will be a factor $\sqrt{10}$ smaller than outside of it. For this reason, a smaller grid size is needed to resolve the solution and the grid with $N = 201$ is no longer adequate. The errors in the solution at times $T = 10$ and $T = 200$ are given in Tables 3 and 4, respectively. Even though the errors are larger in this case, they are still second order accurate at both times and of the same order of magnitude for the same grid sizes. The solution at time $T = 10$ is shown in Figure 6.

To illustrate that the method generalizes to many subdomains, we consider the scattering of a plane incoming wave on a media with different wave speeds inside each bubble; see Figure 7. This calculation was started from homogeneous initial data and driven on the left boundary by the data $u(-2, y, t) = \sin(8\pi t)$. The remaining outer boundaries had homogeneous Neumann conditions. The solution is shown at time $T = 5$. The ambient media has unit wave speed, so this time corresponds closely to the first arrival of the solution at the right boundary. Note that the speed inside the top bubble (which breaks the otherwise symmetrical configuration) equals the unit speed in the ambient media, but the jump conditions are enforced across its interface. Nevertheless, the contour lines of the numerical solution display a high degree of symmetry about the $y = 0$ axis, indicating that the truncation errors in the jump conditions are very small.

To further illustrate the flexibility of the method, we calculate the scattering around an irregular shape described by a periodic cubic spline curve; see Figure 8. Note that the interface has both concave and convex parts. The wave speed inside

TABLE 4

Max error at time $T = 200.0$ for the case $\kappa_e = 10$ inside the dielectric cylinder (u_{err}) and outside of it (w_{err}). The coefficient in the A^TA-dissipation was $\epsilon = 10^{-3}$.

N	h	Time steps	u_{err}, A^TA	w_{err}, A^TA
401	7.5×10^{-3}	53,333	2.03×10^{-1}	7.16×10^{-2}
801	3.75×10^{-3}	106,666	5.81×10^{-2}	1.75×10^{-2}

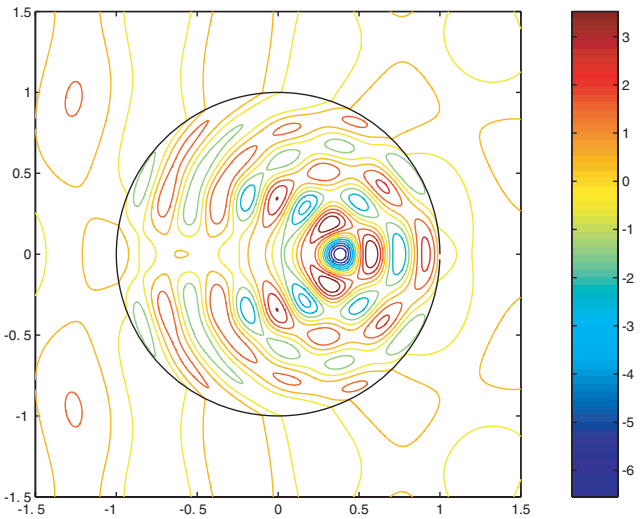


FIG. 6. *The Mie scattering solution at time $T = 10$ with $\kappa_e = 10$. Note the focusing of the wave inside the cylinder.*

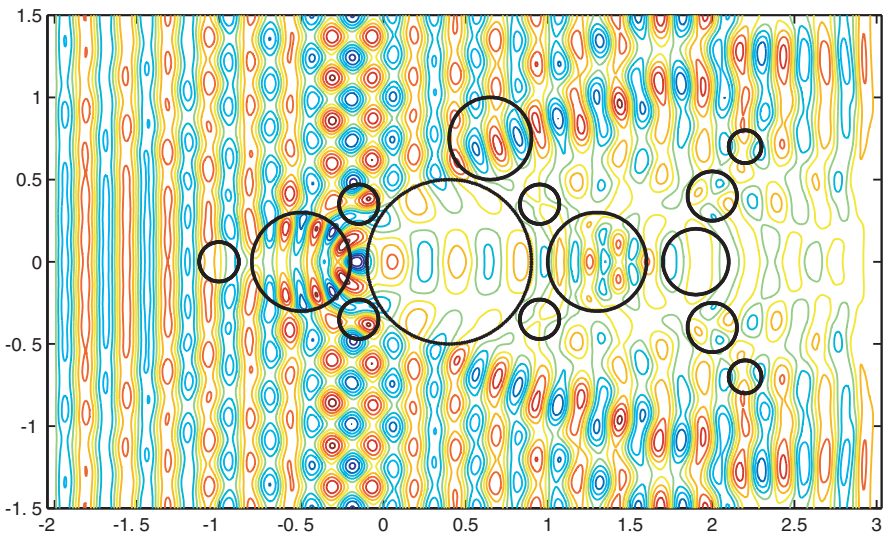


FIG. 7. *An incoming planar wave scattered by several bubbles with both larger and smaller wave speed compared to the ambient media.*

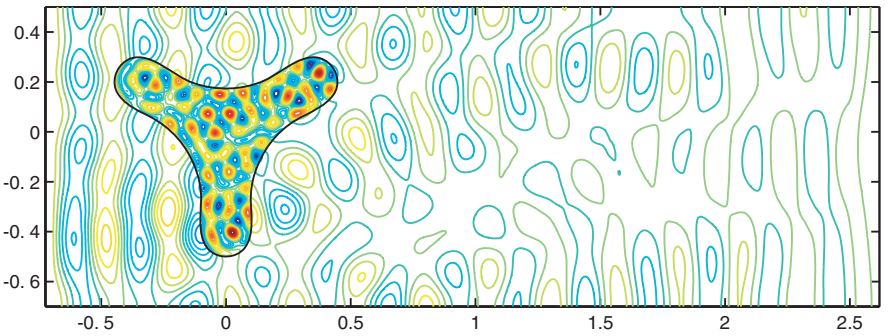


FIG. 8. *An incoming planar wave scattered by an irregular shape described by a periodic spline curve. The wave speed inside the curve was a factor* $\sqrt{10}$ *smaller than on the outside.*

the shape was $1/\sqrt{10}$ and the ambient wave speed was one. As in the previous example, the calculation was started from homogeneous initial conditions and driven on the left boundary by the data $u(-0.7, y, t) = \sin(8\pi t)$. Homogeneous Neumann conditions were imposed on the remaining boundaries. The solution is shown at time $T = 3.3$, which approximately corresponds to when the initial wave front reaches the right boundary.

7. Conclusions. We have developed a second order accurate embedded boundary method for the two-dimensional wave equation with discontinuous coefficients. The current method uses the same grid size throughout the computational domain, but as the jump in wave speed across the interface gets larger, it becomes obvious that this approach either overresolves the solution on one side of the interface or underresolves it on the other side. It would therefore be desirable to extend the method to handle different mesh sizes on different sides of the interface; some initial steps have been taken in this direction. We also expect to generalize the method to handle the more complicated jump conditions associated with the elastic wave equation with discontinuous coefficients.

REFERENCES

[1] H.-O. KREISS, N. A. PETERSSON, AND J. YSTRÖM, *Difference approximations for the second order wave equation*, SIAM J. Numer. Anal., 40 (2002), pp. 1940–1967.

[2] H.-O. KREISS, N. A. PETERSSON, AND J. YSTRÖM, *Difference approximations of the Neumann problem for the second order wave equation*, SIAM J. Numer. Anal., 42 (2004), pp. 1292–1323.

[3] H.-O. KREISS AND N. A. PETERSSON, *A second order accurate embedded boundary method for the wave equation with Dirichlet data*, SIAM J. Sci. Comput., 27 (2006), pp. 1141–1167.

[4] R. B. PEMBER, J. B. BELL, P. COLLELLA, W. Y. CRUTCHFIELD, AND M. WELCOME, *An adaptive Cartesian grid method for unsteady compressible flow in irregular regions*, J. Comput. Phys., 120 (1995), pp. 278–304.

[5] M. J. BERGER, C. HELZEL, AND R. J. LEVEQUE, *h-Box methods for the approximation of hyperbolic conservation laws on irregular grids*, SIAM J. Numer. Anal., 41 (2003), pp. 893–918.

[6] A. DITKOWSKI, K. DRIDI, AND J. S. HESTHAVEN, *Convergent Cartesian grid methods for Maxwell's equations in complex geometries*, J. Comput. Phys., 170 (2001), pp. 39–80.

[7] B. GUSTAFSSON, H.-O. KREISS, AND J. OLIGER, *Time Dependent Problems and Difference Methods*, Wiley–Interscience, New York, 1995.

[8] G. BROWNING, H-.O. KREISS, AND J. OLIGER, *Mesh refinement*, Math. Comp., 27 (1973), pp. 29–39.

[9] D. L. Brown, *A note on the numerical solution of the wave equation with piecewise smooth coefficients*, Math. Comput., 42 (1984), pp. 369–391.
[10] A. N. Tikhonov and A. A. Samarskiĭ, *Homogeneous difference schemes*, Zh. Vychisl. Mat. i Mat. Fiz., 1 (1961), pp. 5–63.
[11] C. Zhang and R. LeVeque, *The immersed interface method for acoustic wave equations with discontinuous coefficients*, Wave Motion, 25 (1997), pp. 237–263.
[12] J. S. Hesthaven, *High-order accurate methods in time-domain computational electromagnetics*, Advances in Imaging and Electron Physics, 127 (2003), pp. 59–123.
[13] S. Zhao and G. W. Wei, *High-order FDTD methods via derivative matching for Maxwell's equations with material interfaces*, J. Comput. Phys., 200 (2004), pp. 60–103.
[14] A. Yefet and E. Turkel, *Fourth order compact implicit method for the Maxwell equations with discontinuous coefficients*, Appl. Numer. Math., 33 (2000), pp. 125–134.
[15] A. Yefet and P. G. Petropoulos, *A staggered fourth-order accurate explicit finite difference scheme for the time-domain Maxwell's equations*, J. Comput. Phys., 168 (2001), pp. 286–315.
[16] Z. Xie, C.-H. Chan, and B. Zhang, *An explicit fourth-order staggered finite-difference time-domain method for Maxwell's equations*, J. Comput. Appl. Math., 147 (2002), pp. 75–98.
[17] E. Turkel and A. Yefet, *On the construction of a high order difference scheme for complex domains in a Cartesian grid*, Appl. Numer. Math., 33 (2000), pp. 113–124.
[18] W. Cai and S. Z. Deng, *An upwinding embedded boundary method for Maxwell's equations in media with material interfaces: 2D case*, J. Comput. Phys., 190 (2003), pp. 159–183.
[19] Z. Xie, C.-H. Chan, and B. Zhang, *An explicit fourth-order curvilinear staggered-grid FDTD method for Maxwell's equations*, J. Comput. Phys., 175 (2002), pp. 739–763.
[20] T. A. Driscoll and B. Fornberg, *Block pseudospectral methods for Maxwell's equations* II: *Two-dimensional, discontinuous-coefficient case*, SIAM J. Sci. Comput., 21 (1999), pp. 1146–1167.
[21] R. Weller and G. H. Shortley, *Calculation of stresses within the boundary of photoelastic models*, J. Appl. Mech., 6 (1939), pp. A71–A78.
[22] H. Johansen and P. Colella, *A Cartesian grid embedded boundary method for Poisson's equation on irregular domains*, J. Comput. Phys., 147 (1998), pp. 60–85.
[23] P. A. Berthelsen, *A decomposed immersed interface method for variable coefficient elliptic equations with nonsmooth and discontinuous solutions*, J. Comput. Phys., 197 (2004), pp. 364–386.
[24] A. L. Fogelson and J. P. Keener, *Immersed interface methods for Neumann and related problems in two and three dimensions*, SIAM J. Sci. Comput., 22 (2000), pp. 1630–1654.
[25] Z. Li and K. Ito, *Maximum principle preserving schemes for interface problems with discontinuous coefficients*, SIAM J. Sci. Comput., 23 (2001), pp. 339–361.
[26] A. Wiegmann and K. P. Bube, *The immersed interface method for nonlinear differential equations with discontinuous coefficients and singular sources*, SIAM J. Numer. Anal., 35 (1998), pp. 177–200.
[27] G. Mie, *Beiträge zur optik trüber medien, speziell kolloidaler metallösungen*, Annalen der Physik, 25 (1908), pp. 377–445.
[28] C. A. Balanis, *Advanced Engineering Electromagnetics*, Wiley, New York, 1989.

SIAM J. SCI. COMPUT.
Vol. 28, No. 6, pp. 2075–2094
© 2006 Society for Industrial and Applied Mathematics

ANALYSIS OF A FOURTH-ORDER SCHEME FOR A THREE-DIMENSIONAL CONVECTION-DIFFUSION MODEL PROBLEM*

ASHVIN GOPAUL† AND MUDDUN BHURUTH†

Abstract. We derive closed form expressions for the eigenvalues and discrete solution arising from a 19-point compact discretization of a three-dimensional convection-diffusion problem. It is shown that the coefficient matrix is positive definite when the cell-Reynolds number is greater than some critical value. By analyzing the terms composing the discrete solution, we prove that an oscillation-free discrete solution is guaranteed whenever the cell-Reynolds number exceeds a value which is grid-size dependent. An interesting result is that as the mesh size is refined, this value approaches the Golden Mean.

Key words. convection-diffusion equation, compact fourth-order scheme, eigenvalues, oscillations

AMS subject classifications. 65N06, 65N22, 65Q05, 35J25

DOI. 10.1137/S1064827502410797

1. Introduction.

Consider the three-dimensional problem

$$-\Delta u + (p,\, q,\, r)^{\mathrm{T}} \cdot \nabla u = f \tag{1.1}$$

on the unit cube $\Omega = (0,\, 1)^3$ subject to Dirichlet boundary conditions $u = g$ on $\partial\Omega$. We assume that the constants p, q, and r are positive on Ω. A central difference approximation to (1.1) leads to a seven-point stencil. Greif [4] has shown that applying one step of the cyclic reduction method followed by a reordering of the unknowns yields a reduced linear system with more favorable properties for solution by iterative methods. The reduced computational molecule uses 19 grid points and is noncompact.

In this paper we consider a 19-point compact fourth-order discretization [7] of the three-dimensional convection-diffusion equation (1.1). Comparison of the advantages and implementation costs of the fourth-order scheme with the central difference scheme have been carried out by Zhang [12]. The central difference scheme, although very accurate for small cell-Reynolds numbers, produces oscillatory solutions in the convection-dominated case. Although the upwind scheme is stable, discretizations over a fine mesh would result in very large linear systems and therefore slow convergence of the iterative schemes for computing the numerical solutions. One attractive advantage in using the fourth-order scheme analyzed for a model problem in this paper is that a much coarser discretization will give numerical solutions of comparable accuracy.

Numerical results reported by Gupta and Zhang [5] and our own experience with the fourth-order scheme for various convection-diffusion problems indicate that oscillation-free discrete solutions are obtained even in the convection-dominated case. The nonoscillatory property of the numerical solutions arising from fourth-order approximations and the unconditional stability with basic iterative methods has been

*Received by the editors November 4, 2002; accepted for publication (in revised form) May 24, 2006; published electronically December 5, 2006.
http://www.siam.org/journals/sisc/28-6/41079.html
†Department of Mathematics, Faculty of Science, University of Mauritius, Reduit, Mauritius (a.gopaul@uom.ac.mu, mbhuruth@uom.ac.mu).

fully exploited in the efficient design of stable multigrid methods for convection-diffusion problems. The nonoscillatory property of the numerical solutions ensures that the residual is smooth and therefore there is no need to use defect-correction techniques. Stable multigrid methods using high-order compact discretizations for two-dimensional convection-diffusion problems are reported in [6, 8] and for three-dimensional problems in [5].

There has been no theoretical analysis of the matrix arising from the 19-point discretization of the three-dimensional problem (1.1). We carry out such an analysis here in the special case of convection in the x-direction, as the simplified matrix structure allows such an analysis, which is otherwise difficult. We also apply a Fourier technique to obtain an analytical expression for the discrete solution and we analyze the quality of the discrete solution produced by the fourth-order scheme. We do not address the question of convergence of iterative methods for the solution of the resulting nonsymmetric linear system.

2. A model three-dimensional problem. We consider the case when $q = r = 0$ in (1.1). On a uniform grid with grid spacing $h = 1/(n+1)$ on Ω, the unit cube, we denote $u_{i,j,k} = u(ih,\, jh,\, kh)$ and $f_{i,j,k} = f(ih,\, jh,\, kh)$, and we let $\gamma = ph/2$ denote the cell-Reynolds number. The 19-point computational molecule is shown in Figure 2.1. The coefficients are defined as follows: $s_1 = 24 + 4\gamma^2$, $s_2 = s_3 = s_4 = s_5 = -2$, $s_{12} = s_{13} = s_{14} = s_{15} = -(1-\gamma)$, $s_8 = s_9 = s_{10} = s_{11} = -1$, $s_6 = -2(1-\gamma+\gamma^2)$, $s_7 = -2(1+\gamma+\gamma^2)$, and $s_{16} = s_{17} = s_{18} = s_{19} = -(1+\gamma)$.

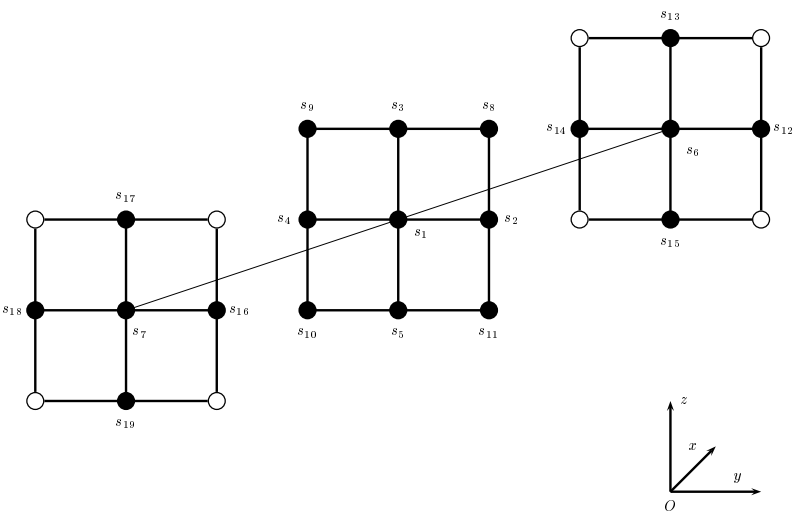


FIG. 2.1. *Computational molecule of a* 19*-point scheme.*

Considering the natural lexicographic ordering done by z-planes, the resulting linear system $A\mathbf{u} = \mathbf{f}$ has coefficient matrix

$$A = \text{blocktridiag}\,[A_-,\, A_0,\, A_+]\,, \tag{2.1}$$

TABLE 2.1
Elements of L_0, K_-, K_0, and K_+.

	a	b	c
L_0	-1	-2	-1
K_-	$-(1+\gamma)$	$-2(1+\gamma+\gamma^2)$	$-(1+\gamma)$
K_0	-2	$24+4\gamma^2$	-2
K_+	$-(1-\gamma)$	$-2(1-\gamma+\gamma^2)$	$-(1-\gamma)$

where the matrices A_-, A_0, and A_+ are given by

$$A_- = \text{tridiag}\,[L_-,\, K_-,\, L_-]\,, \quad A_0 = \text{tridiag}\,[L_0,\, K_0,\, L_0]\,, \quad A_+ = \text{tridiag}\,[L_+,\, K_+,\, L_+]\,.$$

The matrices L_0, K_-, K_0, and K_+ are of the form tridiag $[a, b, c]$, where the nonzero entries a, b, and c are shown in Table 2.1. The matrices L_- and L_+ are diagonal and we write them as $L_- = \text{diag}\,[-(1+\gamma)]$ and $L_+ = \text{diag}\,[-(1-\gamma)]$. The symmetric tridiagonal matrices L_0, K_-, K_0, and K_+ have the same eigenvector matrix $V = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n]$, where the eigenvectors $\mathbf{v}_j$ for $j = 1, 2, 3, \ldots, n$ are given by

$$\mathbf{v}_j = \sqrt{2h}\,[\sin\, j\pi h, \sin\, 2j\pi h, \ldots, \sin\, nj\pi h]^{\mathrm{T}}\,. \tag{2.2}$$

Denoting the eigenvalues of L_0, K_-, K_0, and K_+ by $\lambda_j^{(L_0)}$, $\lambda_j^{(K_-)}$, $\lambda_j^{(K_0)}$, and $\lambda_j^{(K_+)}$, respectively, we have

$$\begin{aligned}
\lambda_j^{(L_0)} &= -2 - 2\cos\, j\pi h,\\
\lambda_j^{(K_-)} &= -2(1+\gamma+\gamma^2) - 2(1+\gamma)\cos\, j\pi h,\\
\lambda_j^{(K_0)} &= 24 + 4\gamma^2 - 4\cos\, j\pi h,\\
\lambda_j^{(K_+)} &= -2(1-\gamma+\gamma^2) - 2\text{sign}(1-\gamma)|1-\gamma|\cos\, j\pi h.
\end{aligned}$$

For the eigenvalue problem $A_0\mathbf{w} = \theta^{(A_0)}\mathbf{w}$, we consider an eigenvector $\mathbf{w}$ of the form $\mathbf{w} = [\xi_1\mathbf{v}_j, \xi_2\mathbf{v}_j, \ldots, \xi_n\mathbf{v}_j]$, where the scalars ξ_i are to be determined from

$$\begin{aligned}
\xi_1 K_0\mathbf{v}_j + \xi_2 L_0\mathbf{v}_j &= \theta\xi_1\mathbf{v}_j,\\
\xi_{i-1}L_0\mathbf{v}_j + \xi_i K_0\mathbf{v}_j + \xi_{i+1}L_0\mathbf{v}_j &= \theta\xi_i\mathbf{v}_j \quad \text{for } i = 2, \ldots, n-1,\\
\xi_{n-1}L_0\mathbf{v}_j + \xi_n K_0\mathbf{v}_j &= \theta\xi_n\mathbf{v}_j.
\end{aligned} \tag{2.3}$$

Since $\mathbf{v}_j$ is an eigenvector of L_0 and K_0, (2.3) can be written in the form

$$\begin{bmatrix}
\lambda_j^{(K_0)} & \lambda_j^{(L_0)} & & & \\
\lambda_j^{(L_0)} & \lambda_j^{(K_0)} & \lambda_j^{(L_0)} & & \\
 & \ddots & \ddots & \ddots & \\
 & & \lambda_j^{(L_0)} & \lambda_j^{(K_0)} & \lambda_j^{(L_0)} \\
 & & & \lambda_j^{(L_0)} & \lambda_j^{(K_0)}
\end{bmatrix}
\begin{bmatrix} \xi_1 \\ \xi_2 \\ \vdots \\ \xi_n \end{bmatrix}
= \theta
\begin{bmatrix} \xi_1 \\ \xi_2 \\ \vdots \\ \xi_n \end{bmatrix}. \tag{2.4}$$

For a given j, the n eigenvalues $\theta_{j,k}^{(A_0)}$, $1 \le k \le n$, of the tridiagonal system (2.4) are given by

$$\begin{aligned}
\theta_{j,k}^{(A_0)} &= \lambda_j^{(K_0)} + 2\left(\text{sign}(\lambda_j^{(L_0)})\right)\lambda_j^{(L_0)}\cos k\pi h\\
&= 24 + 4\gamma^2 - 4\cos k\pi h - 4(1+\cos k\pi h)\cos j\pi h.
\end{aligned} \tag{2.5}$$

The eigenvector $\mathbf{w}_{j,k}$ corresponding to each $\theta_{j,k}$, $1 \leq k \leq n$, is given by

$$\mathbf{w}_{j,k} = [\sin k\pi h\, \mathbf{v}_j,\, \sin 2k\pi h\, \mathbf{v}_j,\, \ldots, \sin ik\pi h\, \mathbf{v}_j, \ldots \sin nk\pi h\, \mathbf{v}_j]^{\mathrm{T}}. \tag{2.6}$$

Now consider the eigenvalue problem $A_+\mathbf{w} = \theta^{(A_+)}\mathbf{w}$. Noting that L_+ is diagonal and K_+ is a symmetric tridiagonal matrix, we obtain the eigenvalues, $\{\theta_{j,k}^{(A_+)}\}_{1\leq j,k\leq n}$, of A_+ as

$$\begin{aligned}\theta_{j,\,k}^{(A_+)} &= \lambda_j^{(K_+)} + 2\left(\operatorname{sign}(\lambda_j^{(L_+)})\right)\lambda_j^{(L_+)}\cos k\pi h \\ &= -2(1-\gamma+\gamma^2) - 2\operatorname{sign}(1-\gamma)|1-\gamma|(\cos j\pi h + \cos k\pi h).\end{aligned} \tag{2.7}$$

Furthermore, A_+ has the same eigenvectors $\{\mathbf{w}_{j,k}\}_{1\leq j,k\leq n}$, as A_0.

Similarly, for the eigenvalue problem $A_-\mathbf{w} = \theta^{(A_-)}\mathbf{w}$, we find that the eigenvalues of A_- are given by

$$\begin{aligned}\theta_{j,\,k}^{(A_-)} &= \lambda_j^{(K_-)} + 2\left(\operatorname{sign}(\lambda_j^{(L_-)})\right)\lambda_j^{(L_-)}\cos k\pi h \\ &= -2(1+\gamma+\gamma^2) - 2(1+\gamma)(\cos j\pi h + \cos k\pi h),\end{aligned} \tag{2.8}$$

and the n^2 eigenvectors are the same vectors $\{\mathbf{w}_{j,k}\}_{1\leq j,k\leq n}$ as in (2.6).

For the eigenvalue problem $A\mathbf{z} = \phi\mathbf{z}$, we consider an eigenvector $\mathbf{z}$ [9] of the form $\mathbf{z} = [\psi_1\mathbf{w}_{j,k},\, \psi_2\mathbf{w}_{j,k},\, \ldots,\, \psi_{n^2}\mathbf{w}_{j,k}]$. The resulting set of equations is given by

$$\begin{aligned}\psi_1 A_0\mathbf{w}_{j,k} + \psi_2 A_+\mathbf{w}_{j,k} &= \phi\psi_1\mathbf{w}_{j,k}, \\ \psi_{i-1}A_-\mathbf{w}_{j,k} + \psi_i A_0\mathbf{w}_{j,k} + \psi_{i+1}A_+\mathbf{w}_{j,k} &= \phi\psi_i\mathbf{w}_{j,k}, \quad i = 2,\, \ldots, n^2-1, \\ \psi_{n^2-1}A_-\mathbf{w}_{j,k} + \psi_{n^2}A_0\mathbf{w}_{j,k} &= \phi\psi_{n^2}\mathbf{w}_{j,k}.\end{aligned} \tag{2.9}$$

Since $\mathbf{w}_{j,k}$ is a common eigenvector of A_0, A_+, and A_-, (2.9) can be written in the form

$$\begin{bmatrix} \theta_{j,k}^{(A_0)} & \theta_{j,k}^{(A_+)} & & & \\ \theta_{j,k}^{(A_-)} & \theta_{j,k}^{(A_0)} & \theta_{j,k}^{(A_+)} & & \\ & \ddots & \ddots & \ddots & \\ & & \theta_{j,k}^{(A_-)} & \theta_{j,k}^{(A_0)} & \theta_{j,k}^{(A_+)} \\ & & & \theta_{j,k}^{(A_-)} & \theta_{j,k}^{(A_0)} \end{bmatrix} \begin{bmatrix} \psi_1 \\ \psi_2 \\ \vdots \\ \psi_{n^2} \end{bmatrix} = \phi \begin{bmatrix} \psi_1 \\ \psi_2 \\ \vdots \\ \psi_{n^2} \end{bmatrix}. \tag{2.10}$$

For fixed j and k, the n eigenvalues $\phi_{i,\,j,\,k}$ of the tridiagonal system (2.10) are given by

$$\phi_{i,\,j,\,k} = \theta_{j,k}^{(A_0)} + 2\left(\operatorname{sign}(\theta_{j,k}^{(A_+)})\right)\sqrt{\theta_{j,k}^{(A_-)}\theta_{j,k}^{(A^+)}}\cos i\pi h,\ 1 \leq i \leq n. \tag{2.11}$$

It then follows from (2.5), (2.7), (2.8), and (2.11) that the n^3 eigenvalues $\{\phi_{i,\,j,\,k}\}_{1\leq i,\,j,\,k\leq n}$ of the three-dimensional coefficient matrix A in (2.1) are given by

$$\begin{aligned}\phi_{i,\,j,\,k} &= 24 + 4\gamma^2 - 4C_k - 4(1+C_k)C_j \\ &\quad + 4\left(1+\gamma^2+\gamma^4 + 2(C_j+C_k) + (1-\gamma^2)(C_j+C_k)^2\right)^{\frac{1}{2}} C_i,\end{aligned}$$

where $C_m = \cos m\pi h$ for $m = i, j, k$, and $1 \le i, j, k \le n$. By letting

$$\zeta = 1 + \gamma^2 + \gamma^4 + 2\left(\cos j\pi h + \cos k\pi h\right) + \left(1 - \gamma^2\right)\left(\cos j\pi h + \cos k\pi h\right)^2, \tag{2.12}$$

we have the following result.

THEOREM 2.1. [illegible] $\phi_{i,j,k}$ $1 \le i, j, k \le n$ [illegible] $n^3 \times n^3$ [illegible] A [illegible] (2.1) [illegible]

$$\phi_{i,j,k} = 24 + 4\gamma^2 - 4\cos k\pi h - 4\left(1 + \cos k\pi h\right)\cos j\pi h + 4\zeta^{\frac{1}{2}}\cos i\pi h. \tag{2.13}$$

In the following, we prove that for a given grid spacing h, the eigenvalues of A are all positive if the cell-Reynolds number γ lies outside a certain interval.

LEMMA 2.1. [illegible] n [illegible] j [illegible] k $1 \le j, k \le n$ [illegible]

$$\vartheta_{j,k} = \cos j\pi h + \cos k\pi h < -1.$$

[illegible] ζ [illegible] (2.12) [illegible] γ [illegible]

$$\left[\frac{1}{2}\left(1 + \vartheta_{j,k} + \sqrt{\vartheta_{j,k}^2 - 2\vartheta_{j,k} - 3}\,\right), \frac{1}{2}\left(-1 - \vartheta_{j,k} + \sqrt{\vartheta_{j,k}^2 - 2\vartheta_{j,k} - 3}\,\right)\right].$$

[illegible] We find that

$$1 + \gamma^2 + \gamma^4 + 2\vartheta_{j,k} + (1 - \gamma^2)\vartheta_{j,k}^2 = \left(\gamma^2 + \frac{1}{2}\left(1 - \vartheta_{j,k}\right)^2\right)^2 + \frac{1}{4}\left(3 - \vartheta_{j,k}\right)\left(1 + \vartheta_{j,k}\right)^3. \tag{2.14}$$

From (2.14), we deduce that $1 + \gamma^2 + \gamma^4 + 2\vartheta_{j,k} + (1 - \gamma^2)\vartheta_{j,k}^2$ is positive for all $\gamma > 0$ if $\vartheta_{j,k} \ge -1$. For $\vartheta_{j,k} < -1$,

$$1 + \gamma^2 + \gamma^4 + 2\vartheta_{j,k} + (1 - \gamma^2)\vartheta_{j,k}^2 = \left(\gamma - \varsigma_1\right)\left(\gamma - \varsigma_2\right)\left(\gamma - \varsigma_3\right)\left(\gamma - \varsigma_4\right),$$

where $\varsigma_1 = \frac{1}{2}(1 + \vartheta_{j,k} + \sqrt{\vartheta_{j,k}^2 - 2\vartheta_{j,k} - 3}\,)$, $\varsigma_2 = \frac{1}{2}(1 + \vartheta_{j,k} - \sqrt{\vartheta_{j,k}^2 - 2\vartheta_{j,k} - 3}\,)$, $\varsigma_3 = \frac{1}{2}(-1 - \vartheta_{j,k} + \sqrt{\vartheta_{j,k}^2 - 2\vartheta_{j,k} - 3}\,)$, and $\varsigma_4 = \frac{1}{2}(-1 - \vartheta_{j,k} - \sqrt{\vartheta_{j,k}^2 - 2\vartheta_{j,k} - 3}\,)$. Noting that the roots ς_2 and ς_4 are negative for all $\vartheta_{j,k} < -1$, we find that $1 + \gamma^2 + \gamma^4 + 2\vartheta_{j,k} + (1 - \gamma^2)\vartheta_{j,k}^2$ is positive if γ lies outside the interval

$$\left[\frac{1}{2}\left(1 + \vartheta_{j,k} + \sqrt{\vartheta_{j,k}^2 - 2\vartheta_{j,k} - 3}\,\right), \frac{1}{2}\left(-1 - \vartheta_{j,k} + \sqrt{\vartheta_{j,k}^2 - 2\vartheta_{j,k} - 3}\,\right)\right]. \qquad \square$$

Note that the above result applies for a given pair (j, k). Considering all those pairs (j, k) for which $\vartheta_{j,k} < -1$ gives the next result.

LEMMA 2.2. [illegible] $J = \{(j, k), 1 \le j, k \le n | \vartheta_{j,k} < -1\}$ [illegible] $(j, k) \in J$ [illegible] $I_{\vartheta_{j,k}}$ [illegible]

$$\left[\frac{1}{2}\left(1 + \vartheta_{j,k} + \sqrt{\vartheta_{j,k}^2 - 2\vartheta_{j,k} - 3}\,\right), \frac{1}{2}\left(-1 - \vartheta_{j,k} + \sqrt{\vartheta_{j,k}^2 - 2\vartheta_{j,k} - 3}\,\right)\right].$$

[illegible] $S = \bigcup_J I_{\vartheta_{j,k}}$ [illegible] $\gamma \notin S$ ζ [illegible] (2.12) [illegible]

Finally, we show that, for n large enough, the matrix A is positive definite if γ lies outside some subset of $(0, \infty)$.

THEOREM 2.2. [illegible] n $\gamma \notin \bigcup_J I_{\vartheta_{j,k}}$ [illegible] $\{\phi_{i,j,k}\}_{1\leq i,j,k\leq n}$ [illegible] (2.13) [illegible]

[illegible]. For $\gamma \notin \bigcup_J I_{\vartheta_{j,k}}$, the eigenvalues $\phi_{i,j,k}$ are all real and the minimum eigenvalue is given by

$$\phi_{n,1,1} \;=\; 24 \,+\, 4\gamma^2 \,-\, 8C_1 \,-\, 4C_1^2 \,+\, 4\bigg(1+\gamma^2+\gamma^4+4C_1+4(1-\gamma^2)C_1^2\bigg)^{\frac{1}{2}} C_n.$$

For n large enough, we have

$$\phi_{n,1,1} \approx 12 + 4\gamma^2 - 4\sqrt{\gamma^4 - 3\gamma^2 + 9}.$$

Writing

$$12 + 4\gamma^2 - 4\sqrt{\gamma^4 - 3\gamma^2 + 9} = \frac{144\gamma^2}{12 + 4\gamma^2 + 4\sqrt{\gamma^4 - 3\gamma^2 + 9}},$$

we find that the minimum eigenvalue $\phi_{n,1,1}$ is positive. It thus follows that all the eigenvalues are positive if $\gamma \notin \cup_J I_{\vartheta_{j,k}}$. ∎

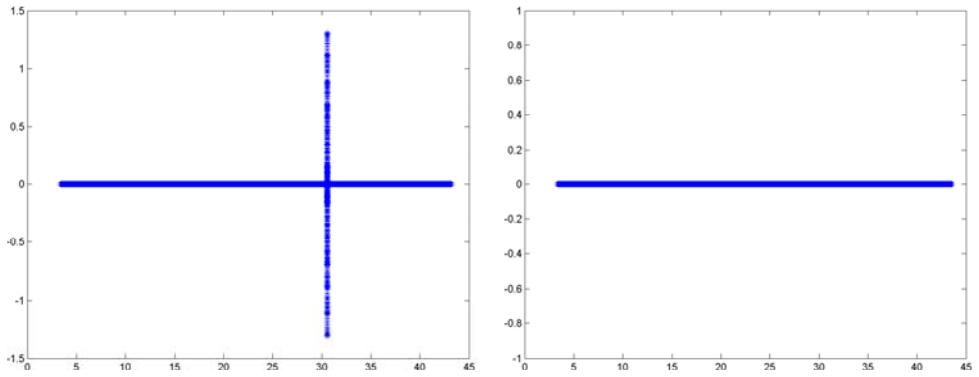
We show in Table 2.2, for different mesh sizes h, endpoints $\gamma_{\min}$ and $\gamma_{\max}$ of intervals outside which the eigenvalues of A are all positive. These endpoints are the analytical bounds computed by considering the set $\cup_J I_{\vartheta_{j,k}}$.

TABLE 2.2
Endpoints $\gamma_{\min}$ and $\gamma_{\max}$ of intervals for different mesh sizes h.

	$\gamma_{\min}$	$\gamma_{\max}$
$h = 1/8$	0.2581	1.4375
$h = 1/16$	0.1502	1.5729
$h = 1/24$	0.2038	1.5980
$h = 1/48$	0.1051	1.6130
$h = 1/96$	0.0355	1.6168

We show in Figure 2.2 the eigenspectrum of A for the cases $\gamma = 1.60$ and $\gamma = 1.62$. We have chosen a mesh size of $h = 1/65$. Observe that there are complex eigenvalues when $\gamma = 1.60$ as $\gamma \in \bigcup_J I_{\vartheta_{j,k}}$ for this case.

The above analysis reveals that the eigenvalues of A are positive if γ lies outside a small subset of $(0, \infty)$. It would therefore be interesting to detect values of γ for which the matrix A is symmetrizable so that the symmetrized linear system can be efficiently solved by the conjugate gradient method. Unfortunately the matrix A cannot be symmetrized by a real diagonal matrix $Q \in \mathbb{R}^{n^3\times n^3}$ when $\gamma > 0$. To see this, consider the block diagonal matrix $Q = \text{diag}\,[Q_1, Q_2, \ldots, Q_n]$, where each Q_i is an $n^2 \times n^2$ diagonal matrix and let Q_{ij}, $j = 1, 2, 3, \ldots n$, denote the $n\times n$ jth diagonal block of Q_i. For the matrix $\widehat{A} = Q^{-1}AQ$, we require that $\widehat{A}^{\mathrm{T}} = \widehat{A}$. Since A_0, the block diagonal entry of A, is symmetric, we need only to satisfy the condition $Q_i^{-1}A_-Q_{i-1} = Q_{i-1}^{-1}A_+Q_i$. This then results in the condition $Q_{ij}K_+Q_{i-1,j}^{-1} = Q_{ij}^{-1}K_-Q_{i-1,j}$. We can then show that we require that $(1+\gamma+\gamma^2)(1-\gamma) = (1-\gamma+\gamma^2)(1+\gamma)$. This condition is satisfied only in the case $\gamma = 0$.



(a) $\gamma = 1.60$. (b) $\gamma = 1.62$.

FIG. 2.2. *Eigenspectrum of A for two different values of* γ.

3. Discrete solutions of three-dimensional problem. Recently Elman and Ramage [2] have described a method to analyze the nature of the discrete solutions produced by the discretization of a model two-dimensional convection-diffusion equation using the Galerkin finite element method. In [3] we have derived analytical expressions for the discrete solution arising from a fourth-order compact discretization of a model two-dimensional convection-diffusion problem and by analyzing the terms composing the discrete solution, we showed that the discrete solution did not contain any oscillations. In this section, we extend the analysis to the three-dimensional equation

$$-\Delta u + pu_x = 0 \quad \text{on} \quad \Omega = (0,1) \times (0,1) \times (0,1), \tag{3.1}$$

with Dirichlet boundary conditions on $\partial\Omega$. For a uniform mesh $h = 1/(n+1)$, we consider the corresponding linear system $A\mathbf{u} = \mathbf{f}$ with coefficient matrix A given in (2.1).

Denoting by V the eigenvector matrix whose columns are the vectors $\mathbf{v}_j$, $j = 1, \ldots, n$, in (2.2), we obtain the decomposition $A_0 = (\mathcal{V}P)\,\mathcal{D}^{(A_0)}\,(\mathcal{V}P)^{\mathrm{T}}$, where $\mathcal{V}$ is a block diagonal matrix given by $\mathcal{V} = \operatorname{diag}(V, V, \ldots, V)$, P is an $n^2 \times n^2$ permutation matrix, and $\mathcal{D}^{(A_0)}$ is a block diagonal matrix with diagonal blocks $D_j^{(A_0)}$ given by

$$D_j^{(A_0)} = \operatorname{tridiag}\left[\lambda_j^{(L_0)}, \lambda_j^{(K_0)}, \lambda_j^{(L_0)}\right].$$

Similarly the matrices A_- and A_+ can be decomposed in the forms $A_+ = (\mathcal{V}P)\,\mathcal{D}^{(A_+)} \cdot (\mathcal{V}P)^{\mathrm{T}}$ and $A_- = (\mathcal{V}P)\,\mathcal{D}^{(A_-)}\,(\mathcal{V}P)^{\mathrm{T}}$, respectively, where the matrices $\mathcal{D}^{(A_-)}$ and $\mathcal{D}^{(A_+)}$ are block diagonal with diagonals blocks $D_j^{(A_-)}$ and $D_j^{(A_+)}$, respectively, given by

$$D_j^{(A_-)} = \operatorname{tridiag}\left[\lambda_j^{(L_-)}, \lambda_j^{(K_-)}, \lambda_j^{(L_-)}\right], \quad D_j^{(A_+)} = \operatorname{tridiag}\left[\lambda_j^{(L_+)}, \lambda_j^{(K_+)}, \lambda_j^{(L_+)}\right].$$

Also, the matrices $D_j^{(A_0)}$, $D_j^{(A_-)}$, and $D_j^{(A_+)}$ can be diagonalized as follows: $V^{\mathrm{T}} D_j^{(A_0)} V = \operatorname{diag}[\theta_{j,1}^{(A_0)}, \theta_{j,2}^{(A_0)}, \ldots, \theta_{j,n}^{(A_0)}]$, $V^{\mathrm{T}} D_j^{(A_-)} V = \operatorname{diag}[\theta_{j,1}^{(A_-)}, \theta_{j,2}^{(A_-)}, \ldots, \theta_{j,n}^{(A_-)}]$, and $V^{\mathrm{T}} D_j^{(A_+)} V = \operatorname{diag}[\theta_{j,1}^{(A_+)}, \theta_{j,2}^{(A_+)}, \ldots, \theta_{j,n}^{(A_+)}]$.

Now let $\mathcal{W}$ denote an $n^3 \times n^3$ block diagonal matrix with diagonal matrices equal to $\mathcal{V}P\mathcal{V}$ on the diagonals and let $\overline{P}$ be an $n^3 \times n^3$ permutation matrix. Then the matrix A can be decomposed as

$$A = \left(\mathcal{W}\overline{P}\right) \mathcal{T}^{(A)} \left(\mathcal{W}\overline{P}\right)^{\mathrm{T}}, \tag{3.2}$$

where $\mathcal{T}^{(A)}$ is a block diagonal matrix with diagonal blocks $T_{j,k}^{(A)}$ given by $T_{j,k}^{(A)} = \text{tridiag}[\theta_{j,k}^{(A_-)}, \theta_{j,k}^{(A_0)}, \theta_{j,k}^{(A_+)}]$. Using (3.2), we can now express the linear system $A\mathbf{u} = \mathbf{f}$ as

$$(\mathcal{W}\overline{P})\mathcal{T}^{(A)}(\mathcal{W}\overline{P})^{\mathrm{T}}\mathbf{u} = \mathbf{f}. \tag{3.3}$$

Letting $\mathbf{y} = (\mathcal{W}\overline{P})^{\mathrm{T}}\mathbf{u}$, (3.3) becomes

$$\mathcal{T}^{(A)}\mathbf{y} = \overline{P}^{\mathrm{T}}\mathcal{W}^{\mathrm{T}}\mathbf{f} = \widehat{\mathbf{f}}. \tag{3.4}$$

Since the matrix $\mathcal{T}^{(A)}$ is block diagonal, we find that (3.4) can be split into n^2 tridiagonal systems of size $n \times n$ given by

$$T_{j,k}^{(A)}\mathbf{y}_{j,k} = \widehat{\mathbf{f}}_{j,k}, \tag{3.5}$$

where $\mathbf{y}_{j,k} = [y_{1,j,k}, y_{2,j,k}, \ldots, y_{n,j,k}]^{\mathrm{T}}$ and $\widehat{\mathbf{f}}_{j,k} = [\widehat{f}_{1,j,k}, \widehat{f}_{2,j,k}, \ldots, \widehat{f}_{n,j,k}]^{\mathrm{T}}$. The discrete solution $\mathbf{u}$ arising from the fourth-order discretization of (3.1) is then obtained from $\mathbf{u} = \mathcal{W}\overline{P}\mathbf{y}$.

3.1. Solution of recurrence relation. The solution of each tridiagonal system (3.5) reduces to the solution of a three-term recurrence relation with constant coefficients given by

$$\theta_{j,k}^{(A_-)}y_{i-1,j,k} + \theta_{j,k}^{(A_0)}y_{i,j,k} + \theta_{j,k}^{(A_+)}y_{i+1,j,k} = \widehat{\mathbf{f}}_{j,k}, \tag{3.6}$$

with boundary conditions $y_{0,j,k}$ and $y_{n+1,j,k}$ which can be computed from the entries of $\widehat{\mathbf{f}}_{j,k}$. To obtain an explicit expression for $y_{i,j,k}$, we consider for given j and k the roots $\mu_1(j,k)$ and $\mu_2(j,k)$ of the auxiliary equation

$$\theta_{j,k}^{(A_+)}\mu(j,k)^2 + \theta_{j,k}^{(A_0)}\mu(j,k) + \theta_{j,k}^{(A_-)} = 0.$$

These roots are given by

$$\begin{aligned}
\mu_1(j,k) &= \frac{-\theta_{j,k}^{(A_0)} + \sqrt{\left(\theta_{j,k}^{(A_0)}\right)^2 - 4\theta_{j,k}^{(A_-)}\theta_{j,k}^{(A_+)}}}{2\theta_{j,k}^{(A_+)}}, \\
\mu_2(j,k) &= \frac{-\theta_{j,k}^{(A_0)} - \sqrt{\left(\theta_{j,k}^{(A_0)}\right)^2 - 4\theta_{j,k}^{(A_-)}\theta_{j,k}^{(A_+)}}}{2\theta_{j,k}^{(A_+)}}.
\end{aligned} \tag{3.7}$$

In the following we omit the indices j and k from the roots of the auxiliary equation and denote $\mu_1 = \mu_1(j,k)$ and $\mu_2 = \mu_2(j,k)$. The solution of (3.6) is given by [2]

$$y_{i,\,j,\,k} = \sum_{m=1}^{3} W_m(j,k)R_m(i,j,k), \tag{3.8}$$

where

$$
\begin{aligned}
R_1(i,j,k) &= \frac{\mu_1^i - \mu_2^i}{\mu_1^{n+1} - \mu_2^{n+1}},\\
R_2(i,j,k) &= \left(1-\mu_1^i\right) - \left(1-\mu_1^{n+1}\right)\left(\frac{\mu_1^i - \mu_2^i}{\mu_1^{n+1} - \mu_2^{n+1}}\right),\\
R_3(i,j,k) &= \mu_1^i - \mu_1^{n+1}\left(\frac{\mu_1^i - \mu_2^i}{\mu_1^{n+1} - \mu_2^{n+1}}\right).
\end{aligned}
\tag{3.9}
$$

The weight function $W_1(j,k)$ is related to the Dirichlet boundary conditions on the plane $x=1$, $W_3(j,k)$ is related to the boundary conditions on the plane $x=0$, and $W_2(j,k)$ combines the boundary conditions on the planes $y=0$, $y=1$, $z=0$, and $z=1$. Using the relation $R_3(i,j,k) = 1 - R_1(i,j,k) - R_2(i,j,k)$, (3.8) can be written in the form [2, Theorem 3.1]

$$
y_{i,\,j,\,k} = W_3(j,k) + \left[W_1(j,k) - W_3(j,k)\right] R_1(i,j,k) + \left[W_2(j,k) - W_3(j,k)\right] R_2(i,j,k).
\tag{3.10}
$$

The discrete solution $\mathbf{u} = \{u_{i,j,k}\}_{1\le i,j,k\le n}$ given by $\mathbf{u} = \mathcal{W}\overline{P}\mathbf{y}$ can then be expressed in the form

$$
u_{i,\,j,\,k} = \sqrt{\frac{2}{n+1}} \sum_{l=1}^{n} \sin(lj\pi h)\overline{y}_{i,\,l,\,k}, \quad \text{where} \quad \overline{y}_{i,\,j,\,k} = \sqrt{\frac{2}{n+1}} \sum_{l=1}^{n} \sin(lk\pi h) y_{i,\,j,\,l}.
\tag{3.11}
$$

3.2. Study of auxiliary roots. In this section, we study the roots, μ_1 and μ_2, of the auxiliary equation in order to determine the behavior of the discrete solution $\mathbf{u}$. We denote $C_j = \cos j\pi h$ and $C_k = \cos k\pi h$. From (2.5), (2.7), (2.8), and (3.7), we find that the auxiliary roots can be expressed as

$$
\mu_1 = \frac{6+\gamma^2 - C_j - C_k - C_jC_k - \sqrt{\chi}}{1-\gamma+\gamma^2+(1-\gamma)(C_j+C_k)}, \quad \mu_2 = \frac{6+\gamma^2 - C_j - C_k - C_jC_k + \sqrt{\chi}}{1-\gamma+\gamma^2+(1-\gamma)(C_j+C_k)},
$$

where

$$
\chi = 35 + 11\gamma^2 - 2(7+\gamma^2 - C_jC_k)(C_j+C_k) + \gamma^2(C_j^2+C_k^2) + C_jC_k(C_jC_k - 12).
$$

Denoting $\vartheta_{j,k} = C_j + C_k$, we first consider the denominator term in both μ_1 and μ_2 given by

$$
1-\gamma+\gamma^2+(1-\gamma)\vartheta_{j,k} = \left[\gamma - \frac{1}{2}(1+\vartheta_{j,k})\right]^2 + \frac{1}{4}\left[(3-\vartheta_{j,k})\,(\vartheta_{j,k}+1)\right].
$$

For $\vartheta_{j,k} \ge -1$, the denominator term is positive for all $\gamma > 0$. For $\vartheta_{j,k} < -1$, we determine the range of γ for which $1-\gamma+\gamma^2+(1-\gamma)\vartheta_{j,k}$ becomes negative. We see that $1-\gamma+\gamma^2+(1-\gamma)\vartheta_{j,k} = (\gamma-\varsigma_1)\,(\gamma-\varsigma_2)$, where $\varsigma_1 = \frac{1}{2}(1+\vartheta_{j,k} + \sqrt{\vartheta_{j,k}^2 - 2\vartheta_{j,k} - 3}\,)$ and $\varsigma_2 = \frac{1}{2}(1+\vartheta_{j,k} - \sqrt{\vartheta_{j,k}^2 - 2\vartheta_{j,k} - 3}\,)$. Because ς_2 is negative for $\vartheta_{j,k} < -1$, we find that $1-\gamma+\gamma^2+(1-\gamma)\vartheta_{j,k}$ will be negative only for $\gamma \in \widetilde{I}_{\vartheta_{j,k}}$ where the interval $\widetilde{I}_{\vartheta_{j,k}}$ for $\vartheta_{j,k} < -1$ is given by

$$
\widetilde{I}_{\vartheta_{j,k}} = \left(0,\, \frac{1}{2}\big(1+\vartheta_{j,k} + \sqrt{\vartheta_{j,k}^2 - 2\vartheta_{j,k} - 3}\,\big)\right).
$$

The term $6+\gamma^2-C_j-C_k-C_jC_k$ in the numerator is always positive for all j, k and for $\gamma > 0$. To find the range of γ for which the numerator is positive, we first note that

$$\chi = [11 + (C_j + C_k)(C_j + C_k - 2) - 2C_jC_k]\,\gamma^2 + [35 + 2(C_j + C_k)(C_jC_k - 7) + C_jC_k(C_jC_k - 12)]\,. \tag{3.12}$$

Since $-1 < C_j,\, C_k < 1$, the terms $11+(C_j+C_k)(C_j+C_k-2)-2C_jC_k$ and $35+2(C_j+C_k)(C_jC_k-7)+C_jC_k(C_jC_k-12)$ are always positive for any values of the indices j and k. It then follows that μ_2 is positive if its denominator $1-\gamma+\gamma^2+(1-\gamma)(C_j+C_k)$ is positive. This is true when $\gamma > \varsigma_1$. To show that the numerator of μ_1 is positive, we need to show that $6+\gamma^2-C_j-C_k-C_jC_k > \sqrt{\chi}$. Now,

$$\begin{aligned}(6+\gamma^2-C_j-C_k-C_jC_k)^2 - \big[&(11+(C_j+C_k)(C_j+C_k-2)-2C_jC_k)\,\gamma^2 \\ &+ (35+2(C_j+C_k)(C_jC_k-7)+C_jC_k(C_jC_k-12))\big] \\ &= \left[\gamma^2 + \frac{1}{2}\left(1-\vartheta_{j,k}^2\right)\right]^2 + \frac{1}{4}\left(3-\vartheta_{j,k}\right)\left(1+\vartheta_{j,k}\right)^3.\end{aligned}$$

Using a similar argument, we find that the numerator of μ_1 will be positive if

$$\gamma < \frac{1}{2}\left(1+\vartheta_{j,k}+\sqrt{\vartheta_{j,k}^2-2\vartheta_{j,k}-3}\right), \quad \text{or} \quad \gamma > \frac{1}{2}\left(-1-\vartheta_{j,k}+\sqrt{\vartheta_{j,k}^2-2\vartheta_{j,k}-3}\right),$$

as it was shown for (2.14).

The above analysis leads to the following result.

LEMMA 3.1. [illegible] $J = \{(j,k), 1 \le j,k \le n | \vartheta_{j,k} < -1\}$ [illegible] $(j,k) \in J$ [illegible] $\widetilde{I}_{\vartheta_{j,k}}^{(\mu_1)}$ [illegible] $\widetilde{I}_{\vartheta_{j,k}}^{(\mu_2)}$ [illegible]

$$\widetilde{I}_{\vartheta_{j,k}}^{(\mu_1)} = \left(0,\, \frac{1}{2}\left(-1-\vartheta_{j,k}+\sqrt{\vartheta_{j,k}^2-2\vartheta_{j,k}-3}\right)\right)$$

[illegible]

$$\widetilde{I}_{\vartheta_{j,k}}^{(\mu_2)} = \left(0,\, \frac{1}{2}\left(1+\vartheta_{j,k}+\sqrt{\vartheta_{j,k}^2-2\vartheta_{j,k}-3}\right)\right),$$

[illegible] $j,\, k$ [illegible] $1 \le j,\, k \le n$

- $\mu_1(j,k) > 0$ [illegible] $\gamma \notin \bigcup_J \widetilde{I}_{\vartheta_{j,k}}^{(\mu_1)}$
- $\mu_2(j,k) > 0$ [illegible] $\gamma \notin \bigcup_J \widetilde{I}_{\vartheta_{j,k}}^{(\mu_2)}$

For $\vartheta_{j,k} < -1$, we remark that the set $\bigcup_J \widetilde{I}_{\vartheta_{j,k}}^{(\mu_2)} \subset \bigcup_J \widetilde{I}_{\vartheta_{j,k}}^{(\mu_1)}$ and therefore we deduce that the auxiliary roots μ_1 and μ_2 are both positive if $\vartheta_{n,n} < -1$ and

$$\gamma > \frac{1}{2}\left(-1-\vartheta_{n,n}+\sqrt{\left(\vartheta_{n,n}-3\right)\left(\vartheta_{n,n}+1\right)}\right).$$

It is interesting to note that when the mesh size is refined, the lower bound on the range of values of γ approaches the Golden Mean $= \frac{1+\sqrt{5}}{2}$.

The next result states the condition under which the smaller root μ_1 is less than unity.

LEMMA 3.2. [illegible] (j,k) [illegible] $\mu_1(j,k)$ [illegible]

$$\gamma \notin \left(0,\, \frac{1}{2}\left(1+\vartheta_{j,k}+\sqrt{\vartheta_{j,k}^2-2\vartheta_{j,k}-3}\right)\right).$$

[illegible]. We need to show that

$$6+\gamma^2-(C_j+C_k)-C_jC_k-\sqrt{\chi} \;<\; 1-\gamma+\gamma^2+(1-\gamma)(C_j+C_k). \tag{3.13}$$

First, observe that

$$6+\gamma^2-(C_j+C_k)-C_jC_k-(1-\gamma+\gamma^2+(1-\gamma)(C_j+C_k)) = 5+(\gamma-2)(C_j+C_k)-C_jC_k.$$

Also,

$$\begin{aligned}(5+\vartheta_{j,k}(\gamma-2)-C_jC_k)^2-\chi \\ = 2\left(-5+2\vartheta_{j,k}+C_jC_k\right)\left(\gamma^2-(1+\vartheta_{j,k})\gamma+(1+\vartheta_{j,k})\right).\end{aligned} \tag{3.14}$$

To prove the inequality (3.13), it is sufficient to show that the right-hand side term in (3.14) is always negative.

As $2\left(-5+2\vartheta_{j,k}+C_jC_k\right)$ is always negative, we look for the range of γ for which

$$\gamma^2-(1+\vartheta_{j,k})\gamma+(1+\vartheta_{j,k})$$

is positive for all j and k. A similar analysis as in the other cases shows that the right-hand side of (3.14) is always negative when

$$\begin{aligned}\gamma < \varsigma_2 &= \frac{1}{2}\left(1+\vartheta_{j,k}-\sqrt{\vartheta_{j,k}^2-2\vartheta_{j,k}-3}\right), \quad \text{or}\\ \gamma > \varsigma_1 &= \frac{1}{2}\left(1+\vartheta_{j,k}+\sqrt{\vartheta_{j,k}^2-2\vartheta_{j,k}-3}\right).\end{aligned}$$

The same arguments as used above show that the smaller root μ_1 is always less than unity when $\gamma > (1+\vartheta_{j,k}+\sqrt{\vartheta_{j,k}^2-2\vartheta_{j,k}-3}\,)/2$. □

3.3. Analysis of recurrence relation solution. To investigate whether the recurrence relation solution $\mathbf{y} = \{y_{i,j,k}\}_{1\le i,j,k\le n}$ given by (3.10) is oscillatory, we consider the two functions $R_1(i,\,j,\,k)$ and $R_2(i,\,j,\,k)$. Note that $R_1(i,\,j,\,k) = [\frac{1-\left(\frac{\mu_1}{\mu_2}\right)^i}{1-\left(\frac{\mu_1}{\mu_2}\right)^{n+1}}]\mu_2^{i-n-1}$. When $\gamma > (-1-\vartheta_{j,k}+\sqrt{\vartheta_{j,k}^2-2\vartheta_{j,k}-3}\,)/2$, the roots μ_1 and μ_2 are positive, and thus in this range of γ, $0<\mu_1/\mu_2<1$ and $\mu_2>0$. We thus conclude that for given j and k, $R_1(i,\,j,\,k)$ is not an oscillatory function of i in this range. From (3.9), we find that $R_2(i,j,k) = (1-\mu_1^i)-(1-\mu_1^{n+1})R_1(i,j,k)$. Since $0<\mu_1<1$ when $\gamma > (1+\vartheta_{j,k}+\sqrt{\vartheta_{j,k}^2-2\vartheta_{j,k}-3}\,)/2$ and $R_1(i,j,k)$ is nonoscillatory when $\gamma > \frac{1}{2}(-1-\vartheta_{j,k}+\sqrt{\vartheta_{j,k}^2-2\vartheta_{j,k}-3})$, we find that, for given j and k, $R_2(i,j,k)$ is also a nonoscillatory function of i.

We summarize the above in the next theorem.

THEOREM 3.1. [illegible] (3.1) [illegible] $h = 1/(n+1)$ [illegible] $\mathbf{y}$ [illegible] (3.6) [illegible] γ [illegible] γ_c [illegible]

$$\gamma_c = \frac{1}{2}\left(2\cos\pi h - 1 + \sqrt{(2\cos\pi h+3)(2\cos\pi h-1)}\right). \tag{3.15}$$

From (3.15), we see that $\gamma_c \to (1+\sqrt{5})/2$ if $h \to 0$.

3.4. Numerical examples. We illustrate the behavior of the discrete solution $\mathbf{u}$ for some three-dimensional problems. We discretize (3.1) on a uniform mesh with grid size $h = 1/33$.

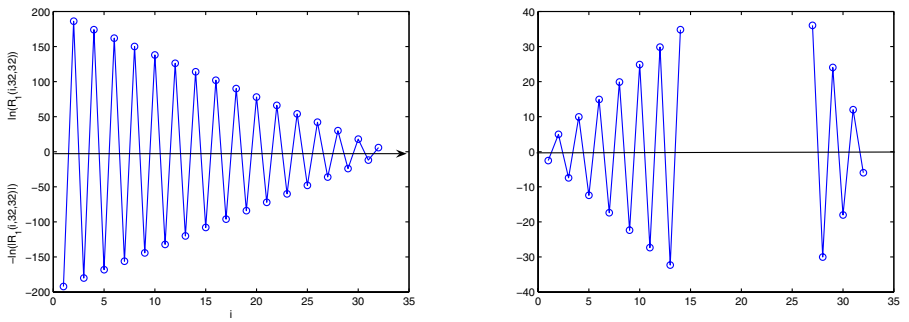
PROBLEM 3.1.

$$u(x, y, z) = \frac{e^{px/2}}{\sinh \sigma} \sin \pi y \sin \pi z \left(2e^{-p/2} \sinh \sigma x + \sinh \sigma(1 - x)\right)$$

$\sigma^2 = 2\pi^2 + p^2/4$ $p > 0$

Since there is no forcing function, the nonzero entries of the right-hand side vector arise from the boundary conditions. Let the Dirichlet boundary conditions on the sides of the unit cube $(0, 1)^3$ be described by the two-dimensional functions f_{z_0}, f_{z_1}, f_{y_0}, f_{y_1}, f_{x_0}, and f_{x_1}. For this problem, the boundary functions $f_{y_0} = f_{y_1} = f_{z_0} = f_{z_1} = 0$. Thus, there is no contribution of the weight W_2 in the recurrence relation solution (3.10). The functions f_{x_0} and f_{x_1} are then given by $f_{x_0}(j, k) = \sin j\pi h \sin k\pi h$ and $f_{x_1}(j, k) = 2 \sin j\pi h \sin k\pi h$.

We consider examples of each of the functions $R_1(i, j, k)$ and $R_2(i, j, k)$ for the case $h = 1/33$. For this mesh size the roots $\mu_2(j, k)$, $1 \le j, k \le n$, are all positive if $\gamma > 0.61648$ and the roots $\mu_1(j, k)$ are all positive if $\gamma > 1.60742$. For $\gamma = 0.6$, we have $\mu_1(32, 32) = -0.08327$ and $\mu_2(32, 32) = -404.561$. Both functions $R_1(i, 32, 32)$ and $R_2(i, 32, 32)$ are thus oscillatory functions of i.

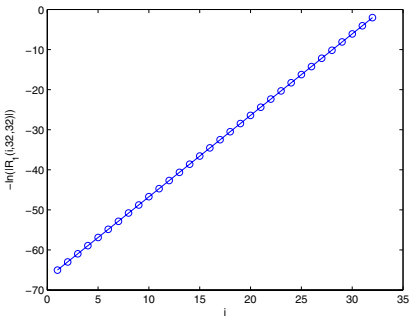


(a) $R_1(i, 32, 32)$, $\gamma = 0.6$. (b) $R_2(i, 32, 32)$, $\gamma = 0.6$.

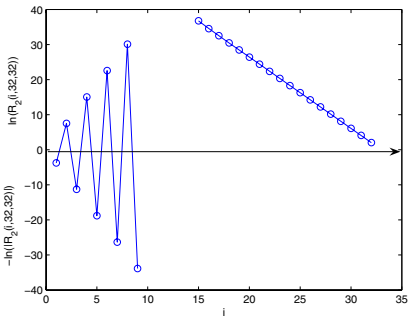
FIG. 3.1. *Functions* $R_1(i, 32, 32)$ *and* $R_2(i, 32, 32)$ *on a logarithmic scale for* $h = 1/33$ *and* $\gamma = 0.6$. *In this example* $\nu_{32,32} = -1.99$.

Figure 3.1 shows the graphs of $R_1(i, 32, 32)$ and $R_2(i, 32, 32)$ on a logarithmic scale when $\gamma = 0.6$. Successive values of $R_1(i, 32, 32)$ alternate in sign. However the large magnitude of the root μ_2 implies that the magnitudes of these oscillations are very small. To show the oscillatory nature of $R_1(i, 32, 32)$, we have plotted $\ln R_1(i, 32, 32)$ when $R_1(i, 32, 32) > 0$ and $-\ln |R_1(i, 32, 32)|$ when $R_1(i, 32, 32) < 0$. The values of $R_2(i, 32, 32)$ oscillate about the value 1. To observe the oscillatory nature of this function, we have plotted $\ln (R_2(i, 32, 32) - 1)$ when $R_2(i, 32, 32) > 1$ and $-\ln (1 - R_2(i, 32, 32))$ when $R_2(i, 32, 32) < 1$. The gap in the plot of $R_2(i, 32, 32)$ corresponds to cases when values of R_2 are practically 1.

In Figure 3.2 we show the plots of the functions $R_1(i, 32, 32)$ and $R_2(i, 32, 32)$

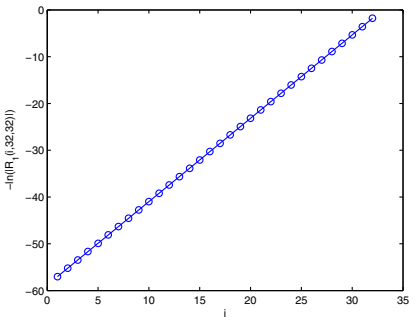


(a) $R_1(i, 32, 32)$, $\gamma = 1.4$.

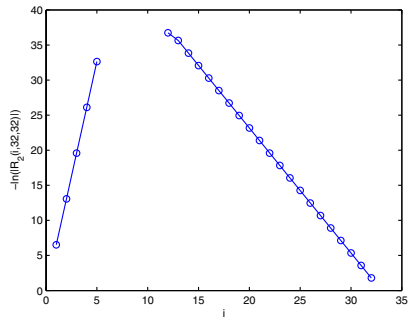


(b) $R_2(i, 32, 32)$, $\gamma = 1.4$.

FIG. 3.2. *Functions* $R_1(i, 32, 32)$ *and* $R_2(i, 32, 32)$ *on a logarithmic scale for* $h = 1/33$ *and* $\gamma = 1.4$. *In this example* $\nu_{32,32} = -1.99$.



(a) $R_1(i, 32, 32)$, $\gamma = 1.62$.



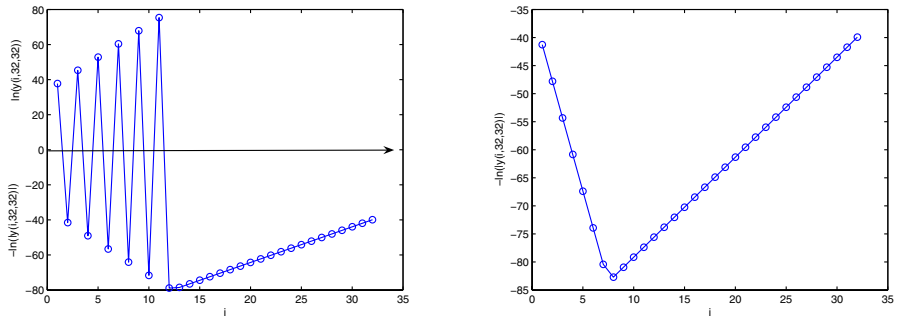
(b) $R_2(i, 32, 32)$, $\gamma = 1.62$.

FIG. 3.3. *Functions* $R_1(i, 32, 32)$ *and* $R_2(i, 32, 32)$ *on a logarithmic scale for* $h = 1/33$ *and* $\gamma = 1.62$. *In this example* $\nu_{32,32} = -1.99$.

for $\gamma = 1.4$. For this value of γ, the function $R_1(i, 32, 32)$ is nonoscillatory, but the theoretical results predict oscillations in $R_2(i, 32, 32)$.

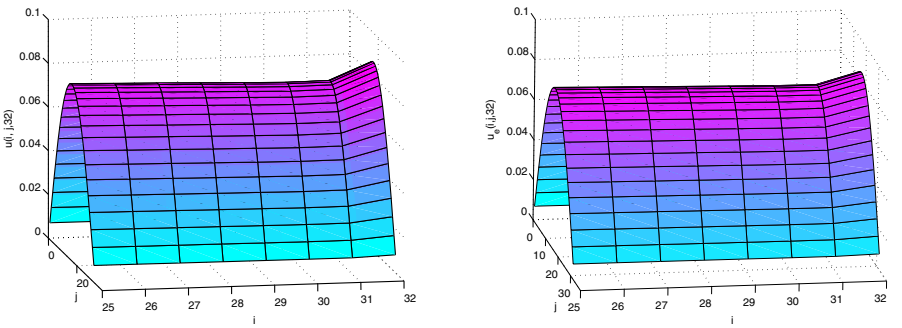
For the case $\gamma = 1.62 > \gamma_c$, our theoretical results predict no oscillations in both R_1 and R_2 and this nonoscillatory property is clearly observed in Figure 3.3. We show in Figure 3.4 the plots of the recurrence relation solution $y(i, 32, 32)$ on a logarithmic scale. We observe oscillations for the case when $\gamma = 1.4 < \gamma_c$. When $\gamma = 1.62 > \gamma_c$, we observe an oscillation-free recurrence relation solution as predicted by Theorem 3.1.

We show in Figures 3.5 and 3.6 plots of the discrete solution $\mathbf{u} = \{u(i, j, k)\}_{1 \le i,j,k \le n}$ and the exact solution $\mathbf{u}_e = \{u_e(i, j, k)\}_{1 \le i,j,k \le n}$ for two different values of γ. For the case $\gamma = 1.4 < \gamma_c$, we do not see any oscillation in the discrete solution $u(i, j, 32)$. This is probably due to the very small magnitude of the oscillations in the functions R_1 and R_2 and the smoothing effect that the transformation (3.11) has on the discrete



(a) $\gamma = 1.4$, $y\,(i, 32, 32)$. (b) $\gamma = 1.62$, $y\,(i, 32, 32)$.

FIG. 3.4. *Plots of the solutions* $y\,(i, 32, 32)$ *on a logarithmic scale when* $n = 32$ *for Problem* 3.1. *The critical value* $\gamma_c = 1.60742$.



(a) $\gamma = 1.4$, $u\,(i, j, 32)$. (b) $\gamma = 1.4$, $u_e\,(i, j, 32)$.

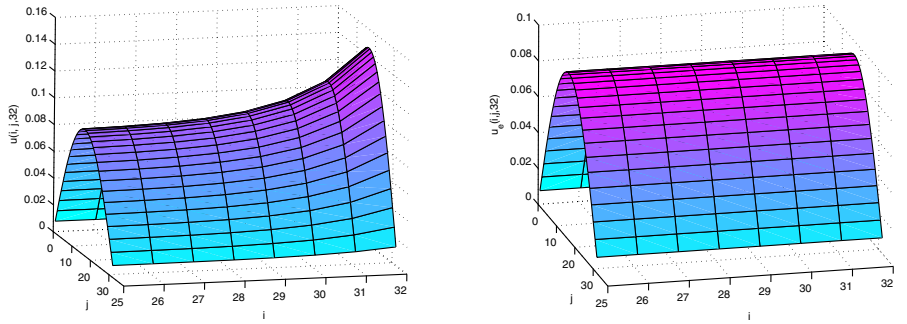
FIG. 3.5. *Plots of the discrete solution* $u\,(i, j, k)$ *and the exact solution* $u_e\,(i, j, k)$ *when* $n = 32$ *for Problem* 3.1.

solution u [2]. Figure 3.6 shows plots of the discrete and exact solutions when $\gamma = 10$. We remark that the discrete solution appears to be slightly diffusive. Similar to one- and two-dimensional problems, this phenomenon can be explained by the addition of some numerical viscosity by the fourth-order approximation scheme [3].

PROBLEM 3.2.

$$f_{x_0} = f_{x_1} = f_{y_0} = f_{y_1} = f_{z_0} = f_{z_1} = 1.$$

$u = 1$



(a) $\gamma = 10$, $u\,(i, j, 32)$. (b) $\gamma = 10$, $u_e\,(i, j, 32)$.

FIG. 3.6. *Plots of the discrete solution* $u\,(i, j, k)$ *and the exact solution* $u_e\,(i, j, k)$ *when* $n = 32$ *for Problem* 3.1.

For this problem, the terms in (3.10) are given as

$$W_1(j,k) = -\frac{4h(\gamma^2 - 3\gamma + 3)}{\theta_{j,k}^{(A_+)}} \left(\sum_{l=1}^{n} \sin j\, l\pi h\right) \left(\sum_{l=1}^{n} \sin k\, l\pi h\right),$$

$$W_3(j,k) = -\frac{4h(\gamma^2 + 3\gamma + 3)}{\theta_{j,k}^{(A_-)}} \left(\sum_{l=1}^{n} \sin j\, l\pi h\right) \left(\sum_{l=1}^{n} \sin k\, l\pi h\right),$$

and

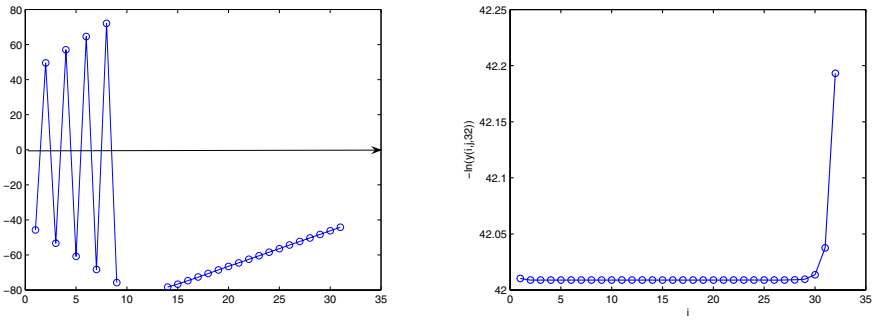
$$W_2(j,k) = \frac{12h}{\theta_{j,k}^{(A_-)} + \theta_{j,k}^{(A_0)} + \theta_{j,k}^{(A_+)}} S(j,\, k)$$

with

$$S(j,\, k) = \left((\sin j\pi h + \sin j\, n\pi h) \sum_{l=1}^{n} \sin k\, l\pi h + (\sin k\pi h + \sin k\, n\pi h) \sum_{l=1}^{n} \sin j\, l\pi h\right).$$

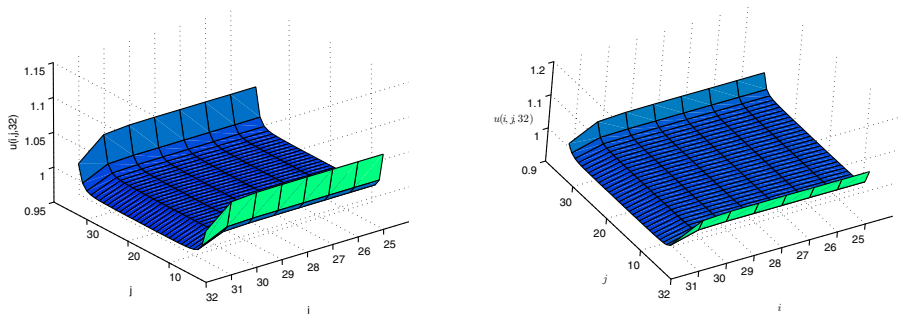
We show in Figure 3.7 plots of the recurrence relation solution for the case $h = 1/33$. For the case $\gamma = 1.4 < \gamma_c$, the recurrence relation solution oscillates. However, the magnitudes of these oscillations are of the order of 10^{-19}. By comparing the first seven significant figures of the first four values of $y(i,\, 32,\, 32)$ which are -5.831129, -5.695544, -5.698597, and -5.698527, we note that the differences between these successive values of $y(i,\, 32,\, 32)$ alternate in sign and therefore there are very tiny oscillations in $y(i,\, 32,\, 32)$. To show this oscillatory nature, we have plotted the successive differences on a logarithmic scale. For the case $\gamma = 1.62 > \gamma_c$ we do not observe any oscillations in the recurrence relation solution.

We show in Figure 3.8 the discrete solution produced by the fourth-order scheme. Recall that the theoretical analysis predicts oscillations in the recurrence relation solution $y(i, j, 32)$ for values of γ in the range $(0,\, \gamma_c)$. Again because of very tiny oscillations in the functions R_1 and R_2, these oscillations are not visible in the plot of the discrete solution $u(i, j, 32)$.



(a) $\gamma = 1.4$, $y\,(i, 32, 32)$. (b) $\gamma = 1.62$, $y\,(i, 32, 32)$.

FIG. 3.7. *Plots of the function* $y(i, j, k)$ *when* $n = 32$ *for Problem* 3.2.



(a) $\gamma = 1.4$, $u\,(i, j, 32)$. (b) $\gamma = 1.62$, $u\,(i, j, 32)$.

FIG. 3.8. *Plots of the discrete solutions* $u\,(i, j, 32)$ *when* $n = 32$ *for Problem* 3.2 *for* $\gamma = 1.4$ *and* $\gamma = 1.62$.

To end this section, we briefly mention a constant coefficient nongrid aligned flow problem. Numerical results described by Gupta and Zhang [5] for the problem (1.1) with $p = 10^3 \cos 35^\circ \cos 45^\circ$, $q = 10^3 \cos 35^\circ \sin 45^\circ$, $r = 10^3 \sin 35^\circ$, and the forcing function f chosen such that the exact solution is $u(x, y, z) = \sin \pi x \sin \pi y \sin \pi z$ indicate that the numerical solution is sufficiently accurate. Although the analysis carried out earlier does not apply here, our computations for this nongrid aligned flow problem indicate that the numerical solution retains the nonoscillatory property.

3.5. Solution accuracy. We consider the case when the convection-diffusion problem (1.1) has variable coefficients, $p(x, y, z)$, $q(x, y, z)$, and $r(x, y, z)$. An important requirement, in order to achieve the fourth-order accuracy, is that the convection coefficients p, q, and r and the forcing function f must be twice continuously differentiable and that the fourth-order derivatives of the solution u must be continuously differentiable.

In [5], Gupta and Zhang carry out accuracy comparisons between the numerical solutions of upwind and fourth-order approximations of four test problems. Here, for illustration on the quality of computed solutions, we did similar computations using Problem 2 from their paper. This convection-diffusion problem (1.1) has variable coefficients with

$$p = -R \sin y \sin z \cos x, \quad q = -R \sin x \sin z \cos y, \quad r = -R \sin x \sin y \cos z.$$

The forcing function f is chosen such that the solution is given by

$$u(x, y, z) = xyz(1-x)(1-y)(1-z)e^{x+y+z}.$$

The discrete solution on a mesh with grid size h is denoted by u^h. For $R = 10, 10^2, 10^3$ we compute the discrete L_∞ norm of the error defined by

$$\|u^h - u\|_\infty = \max_{i,j,k} |u^h(x_i, y_j, z_k) - u(x_i, y_j, z_k)|,$$

and we estimate the order of convergence, ϱ, by computing

$$\varrho = \log_2 \left(\frac{\|u^{2h} - u\|_\infty}{\|u^h - u\|_\infty} \right).$$

The numerical results in Table 3.1 show that the fourth-order compact scheme yields a fourth-order convergence rate when R is small but loses some accuracy when R becomes large. However, the convergence order is at least 3 compared to first-order accurate solutions for the upwind scheme.

TABLE 3.1
Maximum errors and order of convergence for fourth-order and upwind schemes.

	FOS			Upwind		
	$R = 10^1$	$R = 10^2$	$R = 10^3$	$R = 10^1$	$R = 10^2$	$R = 10^3$
$n = 15$	4.77(−06)	3.94(−05)	2.17(−04)	6.31(−03)	2.32(−02)	2.98(−02)
$n = 31$	2.96(−07)	2.59(−06)	2.41(−05)	3.15(−03)	1.23(−02)	1.60(−02)
ϱ	4.00	3.93	3.17	1.00	0.92	0.90

Yavneh [11] analyzed a fourth-order compact scheme for a two-dimensional constant-coefficient convection-diffusion problem and he showed that, while the compact scheme does not suffer from a cross-stream artificial viscosity, it does include a streamwise artificial viscosity that is inversely proportional to the natural viscosity. Here we use similar arguments to discuss the accuracy of the fourth-order scheme for the three-dimensional problem.

Rescaling the convection-diffusion problem (1.1) by letting $t = \max(p, q, r)$, $\epsilon = 1/t$, $\bar{p} = \epsilon p$, $\bar{q} = \epsilon q$, $\bar{r} = \epsilon r$, and $\bar{f} = \epsilon f$, we obtain

$$-\epsilon \Delta u + (\bar{p}, \bar{q}, \bar{r})^{\mathrm{T}} \cdot \nabla u = \bar{f} \tag{3.16}$$

in the unit cube with the same boundary conditions. We denote the convection operator in the characteristic direction ξ by $D_\xi = (\bar{p}, \bar{q}, \bar{r})^{\mathrm{T}} \cdot \nabla$. On a uniform grid with mesh size h, the fourth-order compact scheme for (3.16) can be written in the form

TABLE 3.2
Maximum error and convergence order for $p = 10$, 50, and 100.

	$n = 15$	$n = 31$	Convergence order
$p = 10$	3.57(−04)	2.26(−05)	3.98
$p = 50$	4.31(−02)	4.42(−03)	3.26
$p = 100$	1.82(−01)	3.66(−02)	2.31

$$\left(-\epsilon\Delta^h + D_\xi^h - \frac{h^2}{12\epsilon}D_{\xi\xi}^h\right)u^h = \left(I^h + \frac{h^2}{12}\bar{\Delta}^h - \frac{h^2}{12\epsilon}\bar{D}_\xi^h\right)\bar{f}^h, \tag{3.17}$$

where I^h is the grid-h identity. Denoting the square of the convection operator D_ξ by $D_{\xi\xi}$, the stencils of the discrete operators in (3.17) are given by

$$\Delta^h = \frac{1}{6h^2}\left(\begin{array}{ccc|ccc|ccc} 0 & 1 & 0 & 1 & 2 & 1 & 0 & 1 & 0 \\ 1 & 2 & 1 & 2 & -24 & 2 & 1 & 2 & 1 \\ 0 & 1 & 0 & 1 & 2 & 1 & 0 & 1 & 0 \end{array}\right) = \Delta + \frac{h^2}{12}\Delta^2 + O(h^4),$$

$$\begin{aligned} D_\xi^h &= \frac{1}{12h}\left(\begin{array}{ccc|ccc|ccc} 0 & \bar{q}-\bar{r} & 0 & -\bar{p}+\bar{q} & 2\bar{q} & \bar{p}+\bar{q} & 0 & \bar{q}+\bar{r} & 0 \\ -\bar{p}-\bar{r} & -2\bar{r} & \bar{p}-\bar{r} & -2\bar{p} & 0 & 2\bar{p} & -\bar{p}+\bar{r} & \bar{r} & \bar{p}+\bar{r} \\ 0 & -\bar{q}-\bar{r} & 0 & -\bar{p}-\bar{q} & -2\bar{q} & \bar{p}-\bar{q} & 0 & -\bar{q}+\bar{r} & 0 \end{array}\right) \\ &= D_\xi + \frac{h^2}{6}\Delta D_\xi + O(h^4), \end{aligned}$$

$$\begin{aligned} D_{\xi\xi}^h &= \frac{1}{h^2}\left(\begin{array}{ccc|ccc|ccc} 0 & -\bar{q}\bar{r} & 0 & -\bar{p}\bar{q} & 2\bar{q}^2 & \bar{p}\bar{q} & 0 & \bar{q}\bar{r} & 0 \\ \bar{p}\bar{r} & 2\bar{r}^2 & -\bar{p}\bar{r} & 2\bar{p}^2 & -4(\bar{p}^2+\bar{q}^2+\bar{r}^2) & 2\bar{p}^2 & -\bar{p}\bar{r} & 2\bar{r}^2 & \bar{p}\bar{r} \\ 0 & \bar{q}\bar{r} & 0 & \bar{p}\bar{q} & 2\bar{q}^2 & -\bar{p}\bar{q} & 0 & -\bar{q}\bar{r} & 0 \end{array}\right) \\ &= D_{\xi\xi} + O(h^2), \end{aligned}$$

$$\bar{\Delta}^h = \frac{1}{h^2}\left(\begin{array}{ccc|ccc|ccc} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & -6 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{array}\right) = \Delta + O(h^2),$$

$$\bar{D}_\xi^h = \frac{1}{h^2}\left(\begin{array}{ccc|ccc|ccc} 0 & 0 & 0 & 0 & q & 0 & 0 & 0 & 0 \\ 0 & -r & 0 & -p & 0 & p & 0 & r & 0 \\ 0 & 0 & 0 & 0 & -q & 0 & 0 & 0 & 0 \end{array}\right) = D_\xi + O(h^2).$$

Using the commutativity of Δ and D_ξ and substituting the differential operators and truncation terms in (3.17) lead to

$$\left(1 + \frac{h^2}{12}\Delta - \frac{h^2}{12\epsilon}D_\xi\right)\left(-\epsilon\Delta u^h + D_\xi u^h - \bar{f}^h\right) = O\left(\left(1 + \epsilon + \epsilon^{-1}\right)h^4\right). \tag{3.18}$$

Equation (3.18) shows that the fourth-order scheme becomes less accurate in the convection-dominated case. This is shown by the results in Table 3.2 for problem (3.1). However, when γ becomes large, the overall error is dominated by the error in the boundary layer and this is a common problem with discretization schemes on uniform grids if there are not enough grid points inside the boundary layer [1, 5].

4. Summary. Analytical expressions for the eigenvalues of the matrix arising from a fourth-order discretization of a model three-dimensional constant-coefficient convection-diffusion problem have been derived. It is shown that the matrix is positive definite if the cell-Reynolds number exceeds a certain value which depends on the size of the matrix. A recent method based on Fourier analysis for obtaining an analytical expression for the discrete solution of two-dimensional convection-diffusion problems has been extended to the three-dimensional case. One interesting result shown is that as the mesh size is refined, the recurrence relation solution is nonoscillatory for cell-Reynolds numbers exceeding the Golden Mean. For the convection-dominated case with cell-Reynolds number γ less than the critical value γ_c, in the numerical examples described, the discrete solutions do not present any visible oscillations even though these might be anticipated because of experience with the second-order central scheme.

We showed that although some accuracy is lost for convection-dominated problems, an advantage of the high-order compact scheme over the upwind scheme is that it allows coarser grids and thus smaller linear systems. For variable coefficient problems, we mentioned the regularity requirements on the convection coefficients and the forcing function in order to achieve fourth-order accuracy. Finally, we note that, for two-dimensional problems with thin boundary layers, the deterioration of computed accuracy due to an insufficient number of nodes inside the boundary layer has been successfully treated by employing the graded mesh techniques [10]. Similar techniques can be employed for three-dimensional problems.

Acknowledgments. We thank the referees for the careful reading of the manuscript and their helpful suggestions which considerably improved the work. We also thank the associate editor for his helpful remarks on the manuscript.

REFERENCES

[1] H. C. Elman and A. Ramage, *An analysis of smoothing effects of upwinding strategies for the convection-diffusion equation*, SIAM J. Numer. Anal., 40 (2002), pp. 254–281.

[2] H. C. Elman and A. Ramage, *A characterisation of oscillations in the discrete two-dimensional convection-diffusion equation*, Math. Comp., 72 (2003), pp. 263–288.

[3] A. Gopaul and M. Bhuruth, *Analysis of algebraic systems arising from fourth-order compact discretizations of convection-diffusion equations*, Numer. Methods Partial Differential Equations, 18 (2002), pp. 155–178.

[4] C. Greif, *Analysis of Cyclic Reduction for the Numerical Solution of Three-Dimensional Convection-Diffusion Equations*, Ph.D. thesis, The University of British Columbia, Vancouver, BC, Canada, 1998.

[5] M. M. Gupta and J. Zhang, *High accuracy multigrid solution of the 3D convection-diffusion equation*, Appl. Math. Comput., 113 (2000), pp. 249–274.

[6] M. M. Gupta, J. Kouatchou, and J. Zhang, *A compact multigrid solver for convection-diffusion equations*, J. Comput. Phys., 132 (1997), pp. 123–129.

[7] M. K. Jain, R. K. Jain, and R. K. Mohanty, *Fourth-order finite difference method for three-dimensional elliptic equations with nonlinear first-derivative terms*, Numer. Methods Partial Differential Equations, 8 (1992), pp. 575–591.

[8] A. L. Pardhanani, W. F. Spotz, and G. F. Carey, *A stable multigrid strategy for convection-diffusion using high-order compact discretization*, Electron. Trans. Numer. Anal., 6 (1997), pp. 211–223.

[9] A. Ramage, *A Note on Parameter Choice and Iterative Convergence for Stabilised Discretisations of Advection-Diffusion Problems in Three Dimensions*, Technical report, Mathematics Research Report 32/98, University of Strathclyde, Glasgow, Scotland, 1998.

[10] W. F. Spotz and G. F. Carey, *Formulation and experiments with high-order compact schemes for nonuniform grids*, Internat. J. Numer. Methods Heat Fluid Flow, 8 (1998), pp. 288–303.
[11] I. Yavneh, *Analysis of a fourth-order compact scheme for convection-diffusion*, J. Comput. Phys., 133 (1997), pp. 361–364.
[12] J. Zhang, *An explicit fourth-order compact finite difference scheme for three-dimensional convection-diffusion equation*, Comm. Numer. Methods Engrg., 14 (1998), pp. 209–218.

SIAM J. SCI. COMPUT.
Vol. 28, No. 6, pp. 2095–2113
© 2006 Society for Industrial and Applied Mathematics

AN AUGMENTED LAGRANGIAN-BASED APPROACH TO THE OSEEN PROBLEM*

MICHELE BENZI† AND MAXIM A. OLSHANSKII‡

Abstract. We describe an effective solver for the discrete Oseen problem based on an augmented Lagrangian formulation of the corresponding saddle point system. The proposed method is a block triangular preconditioner used with a Krylov subspace iteration like BiCGStab. The crucial ingredient is a novel multigrid approach for the (1,1) block, which extends a technique introduced by Schöberl for elasticity problems to nonsymmetric problems. Our analysis indicates that this approach results in fast convergence, independent of the mesh size and largely insensitive to the viscosity. We present experimental evidence for both isoP2-P0 and isoP2-P1 finite elements in support of our conclusions. We also show results of a comparison with two state-of-the-art preconditioners, showing the competitiveness of our approach.

Key words. Navier–Stokes equations, finite element, iterative methods, multigrid, preconditioning

AMS subject classifications. 65F10, 65N22, 65F50

DOI. 10.1137/050646421

1. Introduction.

We consider the numerical solution of the steady Navier–Stokes equations governing the flow of a Newtonian, incompressible viscous fluid. Let $\Omega \subset \mathbb{R}^d$ $(d = 2, 3)$ be a bounded, connected domain with a piecewise smooth boundary $\partial\Omega$. Given a force field $\mathbf{f} : \Omega \to \mathbb{R}^d$ and boundary data $\mathbf{g} : \partial\Omega \to \mathbb{R}^d$, the problem is to find a velocity field $\mathbf{u} : \Omega \to \mathbb{R}^d$ and a pressure field $p : \Omega \to \mathbb{R}$ such that

$$-\nu\Delta\mathbf{u} + (\mathbf{u}\cdot\nabla)\,\mathbf{u} + \nabla p = \mathbf{f} \quad \text{in } \Omega, \tag{1.1}$$

$$\operatorname{div}\mathbf{u} = 0 \quad \text{in } \Omega, \tag{1.2}$$

$$\mathbf{u} = \mathbf{g} \quad \text{on } \partial\Omega, \tag{1.3}$$

where $\nu > 0$ is the kinematic viscosity coefficient (inversely proportional to the Reynolds number Re), Δ is the Laplace operator in $\mathbb{R}^d$, ∇ denotes the gradient, and div is the divergence. To determine p uniquely we may impose some additional condition, such as

$$(p,\, 1) = \int_\Omega p\, dx = 0.$$

Equation (1.1) represents conservation of momentum, while (1.2) represents the incompressibility condition, or mass conservation. Due to the presence of the convective term $(\mathbf{u}\cdot\nabla)\,\mathbf{u}$ in the momentum equations, the Navier–Stokes system is nonlinear. It can be linearized in various ways. A widely used linearization scheme is the one

*Received by the editors November 30, 2005; accepted for publication (in revised form) May 30, 2006; published electronically December 5, 2006.
http://www.siam.org/journals/sisc/28-6/64642.html

†Department of Mathematics and Computer Science, Emory University, Atlanta, GA 30322 (benzi@mathcs.emory.edu). The work of this author was supported in part by the National Science Foundation grant DMS-0511336.

‡Department of Mechanics and Mathematics, Moscow State M. V. Lomonosov University, Moscow 119899, Russia (Maxim.Olshanskii@mtu-net.ru). The work of this author was supported in part by the Russian Foundation for Basic Research and the Netherlands Organization for Scientific Research grants NWO-RFBR 047.016.008, RFBR 05-01-00864, RFBR-DFG 06-01-04000.

based on Picard's iteration; see, e.g., [10, section 7.2.2]. Starting with an initial guess $\mathbf{u}^{(0)}$ (with $\operatorname{div}\mathbf{u}^{(0)} = 0$) for the velocity field, Picard's iteration constructs a sequence of approximate solutions $(\mathbf{u}^{(k)}, p^{(k)})$ by solving the linear [illegible]:

$$-\nu\Delta\mathbf{u}^{(k)} + (\mathbf{u}^{(k-1)} \cdot \nabla)\,\mathbf{u}^{(k)} + \nabla p^{(k)} = \hat{\mathbf{f}} \quad \text{in } \Omega, \tag{1.4}$$

$$\operatorname{div}\mathbf{u}^{(k)} = 0 \quad \text{in } \Omega, \tag{1.5}$$

$$\mathbf{u}^{(k)} = \mathbf{g} \quad \text{on } \partial\Omega \tag{1.6}$$

($k = 1, 2, \dots$). Note that no initial pressure needs to be specified. Under certain conditions on ν (which should not be too small) and $\hat{\mathbf{f}}$ (which should not be too large in an appropriate norm), the steady Navier–Stokes equations (1.1)–(1.3) have a unique solution $(\mathbf{u}_*, p_*)$, and the iterates $(\mathbf{u}^{(k)}, p^{(k)})$ converge to it as $k \to \infty$ for any choice of the initial velocity $\mathbf{u}^{(0)}$; see [10, Chapter 7] and the references therein.

Hence, at each Picard iteration one needs to solve an Oseen problem of the form

$$-\nu\Delta\mathbf{u} + (\mathbf{w} \cdot \nabla)\,\mathbf{u} + \nabla p = \hat{\mathbf{f}} \quad \text{in } \Omega, \tag{1.7}$$

$$\operatorname{div}\mathbf{u} = 0 \quad \text{in } \Omega, \tag{1.8}$$

$$\mathbf{u} = \mathbf{g} \quad \text{on } \partial\Omega \tag{1.9}$$

with a known, divergence-free coefficient $\mathbf{w}$. Discretization of (1.7)–(1.9) using LBB-stable finite elements [10, 26] results in a linear system of the form

$$\begin{pmatrix} A & B^T \\ B & O \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} f \\ 0 \end{pmatrix}, \qquad \text{or} \qquad \mathcal{A}\,\mathbf{x} = \mathbf{b}, \tag{1.10}$$

in which u represents the discrete velocities and p the discrete pressure. Here $A = \operatorname{diag}(A_1, \dots, A_d)$ is a block diagonal matrix, where each block corresponds to a discrete convection-diffusion operator with the appropriate boundary conditions. Note that A is nonsymmetric. The rectangular matrix B^T represents the discrete gradient operator, while B represents its adjoint, the (negative) divergence operator. The solvability of this system is discussed, e.g., in [2, section 3.2]; here we assume that the coefficient matrix in (1.10) is nonsingular.

Linear systems of the form (1.10) are often referred to as [illegible]. In recent years, a great deal of effort has been invested in solving systems of this form. Most of the work has been aimed at developing effective preconditioning techniques; see [10], and [2] for an extensive survey. In spite of these efforts, there is still considerable interest in preconditioning techniques that are truly robust, i.e., techniques which result in convergence rates that are largely independent of problem parameters such as mesh size and viscosity. In this paper we describe a promising approach based on an augmented Lagrangian (AL) formulation. The success of this method crucially depends on the availability of a robust multigrid solver for the (1,1) block (submatrix) in the AL formulation of the saddle point system; we develop such a method by building on previous work by Schöberl [29], together with appropriate smoothers for convection-dominated flows. Rather than as a solver, this multigrid iteration will be used to define a block preconditioner for the outer iteration on the coupled saddle point system. We will show that this approach is especially appropriate for discretizations based on discontinuous pressure approximations, but can be used to construct preconditioners for other discretizations using continuous pressures. As an example of the finite element (FE) method with discontinuous pressures we will use the isoP2-P0 pair. In this case, our numerical experiments demonstrate a robust

behavior of the solver with respect to h and ν for some typical wind vector functions $\mathbf{w}$ in (1.7). Further, the isoP2-P1 FE pair is used for the continuous pressure-based approximation. For this case numerical experiments show h-independent convergence rates, with mild dependence on ν when the viscosity becomes very small.

The remainder of the paper is organized as follows. In section 2 we present the (standard) abstract formulation of the Oseen problem and discuss the FE approximations. In section 3 we describe the AL approach and the resulting block preconditioner and briefly discuss related approaches that can be found in the literature. Section 4 is devoted to a study of the spectrum of the preconditioned system, assuming exact solves for the (1,1) block. The multigrid solver for the (1,1) block is described in section 5. Numerical experiments on some two-dimensional (2D) model problems are presented in section 6. Here we also present a comparison with two of the best available preconditioning techniques, showing that our method is quite competitive in terms of convergence rates, robustness, and efficiency. Some concluding remarks are given in section 7.

2. Finite element method. For simplicity, we assume the boundary conditions in (1.9) to be Dirichlet homogeneous (i.e., $\mathbf{g} = \mathbf{0}$). The weak formulation of the Oseen problem reads as follows: Given $\mathbf{f} \in \mathbf{H}^{-1}(\Omega)$, find $\mathbf{u} \in \mathbf{H}_0^1(\Omega)$ and $p \in L_0^2(\Omega)$ such that

$$\begin{aligned}
&\mathcal{L}(\mathbf{u}, p; \mathbf{v}, q) \;=\; \langle \mathbf{f}, \mathbf{v} \rangle \qquad \forall\ \mathbf{v} \in \mathbf{H}_0^1(\Omega),\ q \in L_0^2(\Omega),\\
&\mathcal{L}(\mathbf{u}, p; \mathbf{v}, q) := \nu(\nabla \mathbf{u}, \nabla \mathbf{v}) + ((\mathbf{w} \cdot \nabla)\, \mathbf{u}, \mathbf{v}) - (p, \operatorname{div} \mathbf{v}) + (q, \operatorname{div} \mathbf{u}).
\end{aligned}$$

Given conforming FE spaces $\mathbb{V}_h \subset \mathbf{H}_0^1(\Omega)$ and $\mathbb{Q}_h \subset L_0^2(\Omega)$, the Galerkin FE discretization of (1.7)–(1.9) is based on the following weak formulation: Find $\{\mathbf{u}_h, p_h\} \in \mathbb{V}_h \times \mathbb{Q}_h$ such that

$$\mathcal{L}(\mathbf{u}_h, p_h; \mathbf{v}_h, q_h) = \langle \mathbf{f}, \mathbf{v}_h \rangle \qquad \forall\ \mathbf{v}_h \in \mathbb{V}_h,\ q_h \in \mathbb{Q}_h. \tag{2.1}$$

There are several critical issues associated with the Galerkin FE method for the Oseen problem. One is the compatibility of $\mathbb{V}_h$ and $\mathbb{Q}_h$, i.e., the validation of the LBB (inf-sup) stability condition, which guarantees that the FE velocity space is "rich enough" relative to the FE pressure space. This ensures well-posedness and full approximation order for the FE linear problem. For the numerical experiments in the paper we used the isoP2-P0 FE pair (piecewise constant pressure and piecewise linear continuous velocity on two-times finer triangulation) and the isoP2-P1 FE pair (piecewise linear pressure and piecewise linear continuous velocity on two-times finer triangulation). Both pairs are LBB stable: There exists a mesh-independent constant $\mu(\Omega)$ such that

$$\inf_{q_h \in \mathbb{Q}_h} \sup_{\mathbf{v}_h \in \mathbb{V}_h} \frac{(q_h, \operatorname{div} \mathbf{v}_h)}{\|\nabla \mathbf{v}_h\| \|q_h\|} = \mu_h \geq \mu(\Omega) > 0. \tag{2.2}$$

We note that the chosen FE pairs have stable higher order variants P2-P0 and P2-P1 (the latter is also known as Taylor–Hood element). All considerations and conclusions in the paper remain valid for these higher order finite elements. Piecewise linear velocity-based elements were used solely for the ease of implementation.

Another potential source of instabilities in (2.1) is the presence of dominating convection terms. This necessitates stabilization of the discrete system if the mesh is not sufficiently fine to resolve the stiff behavior of the system. We outline below the SUPG-type method we used. Many more details on the family of SUPG stabilization

methods can be found in, e.g., [6, 28, 31]. Using (2.1) as the starting point, a weighted residual for the FE solution multiplied by a solution-dependent test function is added:

$$\mathcal{L}(\mathbf{u}_h, p_h; \mathbf{v}_h, q_h) + \sum_{\tau \in T_h} \sigma_\tau (-\nu \Delta \mathbf{u}_h + \mathbf{w}\cdot\nabla \mathbf{u}_h + \nabla p_h - \mathbf{f}, \mathbf{w}\cdot\nabla \mathbf{v}_h)_\tau = (\mathbf{f}, \mathbf{v}_h) \qquad \forall\ \mathbf{v}_h \in \mathbb{V}_h,\ q_h \in \mathbb{Q}_h. \tag{2.3}$$

The "new" term in (2.3) is evaluated elementwise for each element τ of a triangulation T_h. The parameters σ_τ are element- and problem-dependent. The general idea behind the choice of σ_τ is to add almost no additional stabilization terms in regions of small mesh Reynolds numbers, so as to recover optimality of the Galerkin method, but to add them in regions of large mesh Reynolds numbers. Several recipes for the particular choice of the stabilization parameters can be found in the literature. We use the one from [22]:

$$\sigma_\tau = \bar{\sigma} \frac{h_\tau}{\|\mathbf{w}\|_{L_2(\tau)}} \frac{2Re_\tau}{1 + Re_\tau}, \qquad Re_\tau := \frac{\|\mathbf{w}\|_{L_2(\tau)} h_\tau}{\nu}. \tag{2.4}$$

This formula for the parameter with $\bar{\sigma} \in [0.2, 1]$ was successfully used in many numerical experiments in [31]. In our experiments we take $\bar{\sigma} = 0.3$.

Remark 2.1. The particular choice of stabilization parameters σ_τ can be quite important for the discrete solution accuracy. On the other hand, using high order finite elements [20] or sufficiently fine meshes may dispense with the need to stabilize the problem. We found, however, that in order to have an efficient solver it is still important to add stabilization to the [illegible] in the multigrid method (see details in section 5). Additionally, the efficiency of the iterative solver described below was found to be essentially independent of the particular choice of $\bar{\sigma} = O(1)$ or of different formulae for σ_τ known from the literature for convection-diffusion problems (see, e.g., [11]). We can also assume that any pressure-dependent part in the stabilization term enters the nonlinear residual in the Picard iterations, thus preserving the saddle point structure (1.10) of the problem.

3. Augmented Lagrangian formulation. Some of the most effective solvers for the Oseen problem found in the literature are based on block triangular preconditioners of the form

$$\mathcal{P} = \begin{pmatrix} \hat{A} & B^T \\ O & \hat{S} \end{pmatrix}, \tag{3.1}$$

where $\hat{A}$ and $\hat{S}$ are approximations to A and to the Schur complement $S = -BA^{-1}B^T$, respectively; see, e.g., [2, 10]. The approximation $\hat{A}$ often consists of an inexact solver for linear systems with coefficient matrix A, such as one or more iterations of an appropriate multigrid solver for convection-diffusion equations. The search for good approximations $\hat{S}$ to the Schur complement (or its inverse) has generated much research in recent years [10]. Note that $S = -BA^{-1}B^T$ is a dense matrix. When ν is relatively large (moderate Reynolds numbers), then S is well conditioned and can be well approximated by the pressure mass matrix M_p, provided that the discretization is LBB-stable. When ν is small, however, finding good and inexpensive approximations to S is not easy. While the best available methods exhibit h-independent convergence of the preconditioned iterations [10], some degradation in the rate of convergence is

observed as $\nu \to 0$. For large Reynolds numbers, the number of iterations can be high.

Here we follow an approach that bypasses the need for a good approximation to the dense matrix S. Suppose we replace the original problem (1.10) with the "regularized" one,

$$\begin{pmatrix} A & B^T \\ B & -\frac{1}{\gamma}W \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} f \\ 0 \end{pmatrix}, \tag{3.2}$$

where $\gamma > 0$ is a parameter and W is a positive definite matrix. The solution $[u(\gamma), p(\gamma)]$ of (3.2) tends to the solution of (1.10) for $\gamma \to \infty$. Observing that the Schur complement of the (1,1) block in the coefficient matrix of (3.2) (sometimes referred to as the "primal" or "velocity" Schur complement) is $A + \gamma B^T W^{-1} B$, we expect

$$\mathcal{P}(\gamma) = \begin{pmatrix} A + \gamma B^T W^{-1} B & B^T \\ O & -\frac{1}{\gamma}W \end{pmatrix} \tag{3.3}$$

(or, in practice, some inexact version of it) to be an effective preconditioner for the regularized problem (3.2). Note that if W is diagonal, or block diagonal with small blocks, the matrix $A + \gamma B^T W^{-1} B$ is also going to be sparse. Our choice of W is discussed in the next section.

There is, however, a well-known difficulty with this approach. The matrix in the regularized problem is close to the original one only for γ large, and for large γ the (1,1) block $A + \gamma B^T W^{-1} B$ becomes increasingly ill conditioned (note that $B^T B$ has a null space of dimension $n - m$ for $n = \dim \mathbb{V}_h$, $m = \dim \mathbb{Q}_h$). Hence, finding an effective inexact solver for the (1,1) block becomes difficult. It is therefore preferable to keep the value of γ moderate, say, $\gamma = O(1)$ or, for reasons of scaling, $\gamma \approx \|\mathbf{w}\|$. This is the idea behind the classical AL approach. In its simplest form, this method consists of replacing the original system (1.10) with the equivalent

$$\begin{pmatrix} A + \gamma B^T W^{-1} B & B^T \\ B & O \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} f \\ 0 \end{pmatrix}, \qquad \text{or} \qquad \widehat{\mathcal{A}}\,\mathbf{x} = \mathbf{b}. \tag{3.4}$$

Clearly, this system has precisely the same solution as the original one (1.10). We refer to [13] for an extensive discussion of AL techniques and their applications. We propose to precondition the AL system (3.4) with the block triangular preconditioner

$$\mathcal{P} = \begin{pmatrix} \hat{A}_\gamma & B^T \\ O & \hat{S} \end{pmatrix}, \tag{3.5}$$

where the action of $\hat{A}_\gamma^{-1}$ is computed via an appropriate (inexact) solver for linear systems involving the matrix $A + \gamma B^T W^{-1} B$,

$$\hat{A}_\gamma \approx A + \gamma B^T W^{-1} B, \tag{3.6}$$

and $\hat{S}$ is implicitly defined through its inverse,

$$\hat{S}^{-1} := -\nu \hat{M}_p^{-1} - \gamma W^{-1}. \tag{3.7}$$

Here $\hat{M}_p^{-1}$ denotes an approximate solve with the pressure mass matrix. This choice of $\hat{S}$ is motivated by Lemma 4.1 below and will be analyzed in the next section. The

pressure mass matrix M_p is diagonal for FE pairs based on piecewise constant pressure approximations. For higher order pressure approximations M_p is no longer diagonal but it is spectrally equivalent to a diagonal matrix $\bar{M}_p$; see [34]. Thus, linear systems with M_p can be solved easily, either exactly or approximately.

It follows from the identity

$$\mathcal{P}^{-1} = \begin{pmatrix} \hat{A}_\gamma^{-1} & O \\ O & I_m \end{pmatrix} \begin{pmatrix} I_n & B^T \\ O & -I_m \end{pmatrix} \begin{pmatrix} I_n & O \\ O & -\hat{S}^{-1} \end{pmatrix} \tag{3.8}$$

that the action of the preconditioner on a given vector requires one application of $\hat{A}_\gamma^{-1}$, one of $\hat{S}^{-1}$, and one sparse matrix-vector product with B^T. The question of the adequacy of the augmented Schur complement preconditioner (3.7) will be addressed in sections 4 and 6. Besides this, the major issue is the definition of $\hat{A}_\gamma^{-1}$, the approximate (1,1) block solver. Developing such a solver is a nontrivial task. Note that the introduction of the additional term $\gamma B^T W^{-1} B$ in the (1,1) block of the saddle point matrix introduces a coupling between the components of the velocity vector, which is originally avoided in the Oseen linearization compared to the Newton one. For vanishing convection the (1,1) block essentially reduces to the one for the classical elasticity problem. For moderate and high Reynolds numbers, however, it becomes increasingly nonsymmetric. It is nevertheless possible to develop efficient iterative solvers for the (1,1) block, at least for some choices of the FE discretization. In section 5 we describe an effective multigrid method for the (1,1) block.

We conclude this section with a brief discussion of related work. An AL approach to the Oseen problem can already be found in [13, Chapter 2], in the context of the classical Uzawa's method. However, the crucial aspect of how to efficiently solve linear systems involving $A + \gamma B^T W^{-1} B$ was not discussed there. Among more recent papers, we mention [7], where a preconditioner based on (3.2) is applied to (symmetric) linear incompressible elasticity problems. The inexact solution of systems involving (3.2) is accomplished by a "primal" Schur complement approach, leading to an overall preconditioning strategy which resembles the approach taken here. There are, however, important differences. Our preconditioner is applied to the AL formulation (3.4), whereas the authors of [7] work with the original saddle point formulation. A more important difference is the fact that the problem tackled in [7] is symmetric; thus, the authors are able to leverage existing techniques for linear elasticity problems when developing an efficient solver for the (1,1) block. We mention that a related approach has been used in [17] to construct block preconditioners for saddle point systems arising from the curl-curl formulation of the time-harmonic Maxwell's equations, also a symmetric problem.

The paper [14] applies a preconditioning technique similar to the one in [7] to Stokes and linearized Navier–Stokes problems. The numerical evidence reported in [14] indicates that as the Reynolds number increases, the rate of convergence of the inner iterative solver used for the (1,1) block (a two-level overlapping additive Schwarz preconditioner with a multiplicative coarse grid correction) tends to deteriorate as the mesh is refined. The deterioration of this solver is already noticeable for a 2D driven cavity problem with Reynolds number $Re = 400$, a fairly easy problem (close to Stokes flow). Furthermore, no analysis of the preconditioner is given. On the positive side, the solver in [14] appears to be well suited for massively parallel architectures, an aspect which we do not address in this paper.

Finally, adding the product $\gamma(\operatorname{div} \mathbf{u}_h, \operatorname{div} \mathbf{v}_h)$ on the left-hand side of the FE method formulation (2.1) is well known in the literature on stabilized FE methods

for fluid problems and sometimes called "the ∇ div-stabilization." Using this stabilization may change the FE solution. However, the effect on the iterative solution of (1.10) is similar to the one resulting from adding $\gamma B^T W^{-1} B$ (with $W = M_p$) to the (1,1) block of the linear algebraic system. This linear algebra aspect of the ∇ div-stabilization was studied for the Stokes problem in [24] and for the linearized rotation form of the Navier–Stokes problem in [25]. In particular, numerical experiments in [25] demonstrate ν-independent convergence of the preconditioned BiCGstab method if $\gamma = O(1)$ and if exact solves are used for systems involving the (1,1) block. However, in [24] and [25] the question of finding a robust solver or preconditioner for the (1,1) block with respect to the ratio ν/γ was left open.

4. Analysis of the preconditioner. It is well known that characterizing the rate of convergence of nonsymmetric preconditioned iterations can be a difficult task. In particular, eigenvalue information alone may not be sufficient to give meaningful estimates of the convergence rate of a method like preconditioned GMRES [16]. The situation is even more complicated for a method like BiCGStab, for which virtually no convergence theory exists. Nevertheless, experience shows that for many linear systems arising in practice, a well-clustered spectrum (away from zero) usually results in rapid convergence of the preconditioned iteration.

In this section we show that for the preconditioner (3.5), the eigenvalues of the preconditioned matrix $\mathcal{M} = \mathcal{P}^{-1}\widehat{\mathcal{A}}$ are enclosed in a rectangular region contained in the right half-plane $\Re(z) > 0$, the sides of which do not depend on the mesh size h and for sufficiently large γ do not depend on ν. Here we assume exact solves for the (1,1) block, i.e., $\hat{A}_\gamma = A + \gamma B^T W^{-1} B$. Hence, at least for this "ideal" version of the preconditioner, the eigenvalues of the preconditioned matrix are bounded away from zero, independently of h and ν. Our analysis makes use of the following simple lemma, which is a straightforward consequence of [12, Exercise 12.12] or [15, Proposition 2.1].

LEMMA 4.1. Let $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{m\times n}$ $(m \le n)$, and let $\gamma \in \mathbb{R}$ be such that A, $A + \gamma B^T W^{-1} B$, $BA^{-1}B^T$, and $B(A + \gamma B^T W^{-1} B)^{-1} B^T$ are nonsingular. Then

$$\left[B(A + \gamma B^T W^{-1} B)^{-1} B^T\right]^{-1} = \left(BA^{-1}B^T\right)^{-1} + \gamma W^{-1}. \tag{4.1}$$

We note that the conditions of Lemma 4.1 are satisfied if we assume that B has full row rank and that A is positive real (i.e., the symmetric part of A is positive definite); see [2, section 3]. These conditions are satisfied, in particular, for the matrices considered in this paper.

Recall that $\mathcal{P}^{-1}\widehat{\mathcal{A}}$ and $\widehat{\mathcal{A}}\mathcal{P}^{-1}$ have the same eigenvalues. A simple calculation shows that

$$\widehat{\mathcal{A}}\mathcal{P}^{-1} = \begin{pmatrix} I_n & O \\ B(A + \gamma B^T W^{-1} B)^{-1} & -B(A + \gamma B^T W^{-1} B)^{-1} B^T \hat{S}^{-1} \end{pmatrix}. \tag{4.2}$$

Hence, the preconditioned matrix has the eigenvalue 1 of multiplicity n, with the remaining eigenvalues λ_i $(1 \le i \le m)$ being solutions of the generalized eigenproblem

$$B(A + \gamma B^T W^{-1} B)^{-1} B^T p = -\lambda \hat{S} p.$$

Setting $W = M_p$ and $\hat{M}_p = M_p$, Lemma 4.1 implies

$$(\nu + \gamma)\lambda_i^{-1} = \gamma + \mu_i^{-1},$$

where μ_i satisfies the generalized eigenproblem

$$BA^{-1}B^T q = \mu M_p\, q. \tag{4.3}$$

Hence, the nonunit eigenvalues of $\mathcal{P}^{-1}\widehat{\mathcal{A}}$ are given by

$$\lambda_i = \frac{\gamma+\nu}{\gamma+\mu_i^{-1}}, \quad 1 \le i \le m. \tag{4.4}$$

It has been shown in [8] that the eigenvalues of (4.3) are enclosed in a rectangle contained in the half-plane $\Re(z) > 0$; using this result and the relation (4.4), we can conclude that the same is true of the eigenvalues of $\mathcal{P}^{-1}\widehat{\mathcal{A}}$. If we denote by a_i and b_i the real and imaginary parts of μ_i, respectively, easy manipulations result in the following expressions for the real and the imaginary parts of λ_i:

$$\Re(\lambda_i) = \frac{(\nu+\gamma)(a_i+\gamma(a_i^2+b_i^2))}{(\gamma a_i+1)^2+(\gamma b_i)^2} \qquad \text{and} \qquad \Im(\lambda_i) = \frac{(\nu+\gamma)b_i}{(\gamma a_i+1)^2+(\gamma b_i)^2}. \tag{4.5}$$

The following result is an immediate consequence of (4.5).

THEOREM 4.2. [illegible] $W = M_p$ [illegible] $\mathcal{P}^{-1}\widehat{\mathcal{A}}$ [illegible] 1 [illegible] n [illegible] m [illegible] λ_i [illegible] (4.4) [illegible] $\mu_i = a_i + \mathrm{i}\, b_i$ [illegible] (4.3) [illegible]

$$1 - \frac{1}{\gamma a_i + 1} \le \Re(\lambda_i) < 1 + \nu\gamma^{-1} \quad \text{and} \quad |\Im(\lambda_i)| < (1+\nu\gamma^{-1}) \min\left\{\gamma b_i, \frac{1}{\gamma b_i}\right\}.$$

[illegible] 1 [illegible] $\gamma \to \infty$

The bounds in Theorem 4.2, while independent of h, will generally depend on ν in an implicit way through a_i and b_i. For the case of the Galerkin FE method (2.1) (with no stabilization), the results in [8] provide us with the following estimates for the eigenvalues of (4.3):

$$c\,\nu \le a_i \le C\,\nu^{-1}, \qquad |b_i| \le C_1 \nu^{-1} \tag{4.6}$$

for some h- and ν-independent positive constants c, C, C_1 (here we assume $0 < \nu \le 1$). Together with the estimates from Theorem 4.2, this result implies that it is sufficient to set $\gamma = O(\nu^{-1})$ to ensure that all nonunit eigenvalues of $\mathcal{P}^{-1}\widehat{\mathcal{A}}$ are contained in a box $[\mathtt{a}, \mathtt{A}] \times [-\mathtt{b}, \mathtt{b}]$, $\mathtt{a} > 0$, in the complex plane, with $\mathtt{a}, \mathtt{A}, \mathtt{b}$ independent of ν and h. We note that this theoretical result is of somewhat limited use, since in practice we found that it is unnecessary to use large values of γ (!), or to solve linear systems involving the (1,1) block "exactly" (the latter being perhaps less surprising).

[illegible] 4.3. In numerical experiments we found that using $\gamma \approx \|\mathbf{w}\|$ already provides ν- and h-independent convergence of the Krylov subspace method for a wide range of values of ν and h (it should be noted that we also use streamline diffusion-type velocity stabilization). Furthermore, assume that $\hat{A}_\gamma^{-1}$ is not a solver but a preconditioner for the (1,1) block and that its efficiency deteriorates with $\nu/\gamma \to 0$. Then the optimal value of γ in the Krylov subspace method will be quite modest for $\nu \ll 1$ (see Table 6.2 in section 6).

[illegible] 4.4. The result of the theorem remains valid if one considers $W = \bar{M}_p$ and with $\bar{M}_p$ replacing M_p in (4.3). Following the arguments in [8], it can be easily verified that (4.6) still holds. In general, W is not necessarily an approximation to a mass matrix or identity; see the discussion in [15] in conjunction with other saddle point problems. In this paper, however, we do not pursue other choices of W.

5. A multigrid method for the (1,1) block. In this section we develop a multigrid method to (approximately) solve the system of equations for the discrete velocity at each outer iteration. For the purpose of analysis in this section we consider $W = M_p$; however, in the implementation it is more practical to set $W = \bar{M}_p$. By a simple scaling argument one can assume $\|\mathbf{w}\|_\infty = 1$; thus we can let $\gamma = 1$. The system one needs to solve is

$$A\,u + B^T M_p^{-1} B\,u = f. \tag{5.1}$$

To define an FE counterpart of (5.1) we introduce the orthogonal projector $P_h : L_0^2 \to \mathbb{Q}_h$. The corresponding FE problem takes the form

$$\begin{aligned}\nu(\nabla \mathbf{u}_h, \nabla \mathbf{v}_h) + \sum_{\tau \in T_h} \sigma_\tau (\mathbf{w}\cdot\nabla \mathbf{u}_h, \mathbf{w}\cdot\nabla \mathbf{v}_h)_\tau + ((\mathbf{w}\cdot\nabla)\,\mathbf{u}_h, \mathbf{v}_h) \\ + (P_h \operatorname{div} \mathbf{u}_h, \operatorname{div} \mathbf{v}_h) = (f, \mathbf{v}_h) \qquad \forall\ \mathbf{v}_h \in \mathbb{V}_h.\end{aligned} \tag{5.2}$$

An ideal multigrid method for solving (5.1) should be robust with respect to ν and different types of wind $\mathbf{w}$ and have optimal computational complexity. There are several difficulties in developing such a method. First, note that matrix B has a kernel of dimension $n - m$. In the case of $\nu \ll 1$ and if $\mathbf{w}$ is small in part of Ω, some vectors from $\ker(B)$ may nearly vanish after multiplication by $A + B^T M_p^{-1} B$; see Figure 5.1 for an example of locally supported functions from $\ker(B)$ for a particular FE pair. Thus, common relaxation iterations like Gauss–Seidel could fail to smooth out oscillatory components from $\ker(B)$ that may be present in the error. At the same time, if a local Reynolds number is large, the problem has convection-dominated character at least [illegible] B. Building robust multigrid methods for convection-dominated problems is a well-known challenge; see the overview in [23].

To define the necessary components of the multigrid method, let us consider two pairs of finite element spaces $\mathbb{V}_h \times \mathbb{Q}_h$ and $\mathbb{V}_H \times \mathbb{Q}_H$, corresponding to fine and coarse grids, respectively.

It was shown in [27, 10, 23] that a stable direct discretization on the coarse grid is the preferred choice in geometric multigrid methods for convection-diffusion problems. Additionally, in the case of elasticity-type problems ($\mathbf{w} = \mathbf{0}$ in (2.1)), stability of the coarse grid problem means that velocity and auxiliary pressure approximations on the coarser grid remain compatible, so as to avoid the so-called locking phenomenon [1]. Thus, to build a matrix for a coarse grid problem in our multigrid method we consider the FE formulation

$$\begin{aligned}\nu(\nabla \mathbf{u}_H, \nabla \mathbf{v}_H) + \sum_{\tau \in T_H} \sigma_\tau (\mathbf{w}\cdot\nabla \mathbf{u}_H, \mathbf{w}\cdot\nabla \mathbf{v}_H)_\tau + ((\mathbf{w}\cdot\nabla)\,\mathbf{u}_H, \mathbf{v}_H) \\ + (P_H \operatorname{div} \mathbf{u}_H, \operatorname{div} \mathbf{v}_H) = (f, \mathbf{v}_H) \qquad \forall\ \mathbf{v}_H \in \mathbb{V}_H,\end{aligned}$$

where the new set of parameters σ_τ corresponds to the coarse grid and P_H projects on $\mathbb{Q}_H$. We further define the problem-dependent "energy" norms on $\mathbf{H}_0^1$,

$$\|\mathbf{v}\|_h^2 = \nu(\nabla \mathbf{v}, \nabla \mathbf{v}) + (P_h \operatorname{div} \mathbf{v}, \operatorname{div} \mathbf{v}) \quad \text{and} \quad \|\mathbf{v}\|_H^2 = \nu(\nabla \mathbf{v}, \nabla \mathbf{v}) + (P_H \operatorname{div} \mathbf{v}, \operatorname{div} \mathbf{v}),$$

and set $a_h(\mathbf{u}, \mathbf{v}) = \nu(\nabla \mathbf{u}, \nabla \mathbf{v}) + (P_h \operatorname{div} \mathbf{u}, \operatorname{div} \mathbf{v})$. Consider the standard prolongation operator $p : \mathbb{V}_H \to \mathbb{V}_h$ induced by the embedding $\mathbb{V}_H \subset \mathbb{V}_h$. This is a common choice in a multigrid method for solving elliptic problems. However, it appears not

to work well for (5.1) in the case of small ν. The reason is that $\mathbf{v}_H \in \ker(P_H \operatorname{div})$ does not yield in general $\mathbf{v}_H \in \ker(P_h \operatorname{div})$. Therefore, the standard prolongation is not uniformly stable in the energy norm; i.e., the constant C in the estimate

$$\|p\,\mathbf{v}_H\|_h \le C\,\|\mathbf{v}_H\|_H \qquad \forall\ \mathbf{v}_H \in \mathbb{V}_H \tag{5.3}$$

could grow unboundedly for $\nu \to 0$. In [29] it is suggested that to enforce the stability of the prolongation one can make a correction of $\mathbf{v}_H$ on a fine grid in a way involving solution of local problems. To present this idea in a general setting we consider the L^2-orthogonal decomposition

$$\mathbb{Q}_h = \mathbb{Q}_H \oplus \tilde{\mathbb{Q}}_h. \tag{5.4}$$

We prove the following lemma.

LEMMA 5.1. [illegible] $\tilde{\mathbb{V}}_h \subset \mathbb{V}_h$ [illegible] c [illegible] h [illegible]

$$\inf_{\tilde{q}_h \in \tilde{\mathbb{Q}}_h} \sup_{\tilde{\mathbf{v}}_h \in \tilde{\mathbb{V}}_h} \frac{(\tilde{q}_h, \operatorname{div} \tilde{\mathbf{v}}_h)}{\|\nabla \tilde{\mathbf{v}}_h\| \|\tilde{q}_h\|} \ge c > 0 \tag{5.5}$$

[illegible] $\tilde{\mathbb{V}}_h \subset \ker(P_H \operatorname{div})$ [illegible] $\mathbf{u}_H$ [illegible] $\tilde{\mathbf{u}}_h$ [illegible]

$$a_h(\tilde{\mathbf{u}}_h, \tilde{\mathbf{v}}_h) = a_h(\mathbf{u}_H, \tilde{\mathbf{v}}_h) \qquad \forall\ \tilde{\mathbf{v}}_h \in \tilde{\mathbb{V}}_h. \tag{5.6}$$

[illegible] $p : \mathbb{V}_H \to \mathbb{V}_h$ [illegible]

$$p\,\mathbf{u}_H = \mathbf{u}_H - \tilde{\mathbf{u}}_h. \tag{5.7}$$

[illegible] p [illegible] (5.3) [illegible] C [illegible] h [illegible] ν [illegible]. Define the following bilinear form on $\mathbf{H}_0^1 \times L_0^2$:

$$\mathcal{B}(\mathbf{u}, p; \mathbf{v}, q) := \nu(\mathbf{u}, \mathbf{v}) + (p, \operatorname{div} \mathbf{v}) + (q, \operatorname{div} \mathbf{u}) - (p, q).$$

For given $\mathbf{u}_H$ denote

$$p_H = P_H \operatorname{div} \mathbf{u}_H. \tag{5.8}$$

Define $\tilde{\mathbf{u}}_h \in \tilde{\mathbb{V}}_h$ and $\tilde{p}_h \in \tilde{\mathbb{Q}}_h$ by solving

$$\mathcal{B}(\tilde{\mathbf{u}}_h, \tilde{p}_h; \tilde{\mathbf{v}}_h, \tilde{q}_h) = \mathcal{B}(\mathbf{u}_H, p_H; \tilde{\mathbf{v}}_h, \tilde{q}_h) \quad \forall\ \tilde{\mathbf{v}}_h \in \tilde{\mathbb{V}}_h,\ \tilde{q}_h \in \tilde{\mathbb{Q}}_h. \tag{5.9}$$

This problem is well-posed thanks to (5.5). Taking $\tilde{\mathbf{v}}_h = 0$ in (5.9) we get

$$(\operatorname{div}(\mathbf{u}_H - \tilde{\mathbf{u}}_h), \tilde{q}_h) = (p_H - \tilde{p}_h, \tilde{q}_h) \quad \forall\ \tilde{q}_h \in \tilde{\mathbb{Q}}_h. \tag{5.10}$$

Also, since (5.8) implies $(p_H - \operatorname{div} \mathbf{u}_H, q_H) = 0$ and the assumptions $\tilde{\mathbb{V}}_h \subset \ker(P_H \operatorname{div})$ and $\tilde{\mathbb{Q}}_h \perp \mathbb{Q}_H$ imply $(\tilde{p}_h, q_H) = (\operatorname{div} \tilde{\mathbf{u}}_h, q_H) = 0$ for any $q_H \in \mathbb{Q}_H$, we get

$$(\operatorname{div}(\mathbf{u}_H - \tilde{\mathbf{u}}_h), q_H) = (p_H - \tilde{p}_h, q_H) \quad \forall\ q_H \in \mathbb{Q}_H. \tag{5.11}$$

Since $\mathbb{Q}_h = \mathbb{Q}_H \oplus \tilde{\mathbb{Q}}_h$, relations (5.10) and (5.11) yield

$$P_h \operatorname{div}(\mathbf{u}_H - \tilde{\mathbf{u}}_h) = p_H - \tilde{p}_h. \tag{5.12}$$

Using (5.12) and (5.9) with $\tilde{q}_h = 0$ we can see that $\mathbf{u}_H$ and $\tilde{\mathbf{u}}_h$ found from (5.9) satisfy

$$\nu(\nabla\tilde{\mathbf{u}}_h, \nabla\tilde{\mathbf{v}}_h) + (P_h \operatorname{div}\tilde{\mathbf{u}}_h, \operatorname{div}\tilde{\mathbf{v}}_h) = \nu(\nabla\mathbf{u}_H, \nabla\tilde{\mathbf{v}}_h) + (P_h \operatorname{div}\mathbf{u}_H, \operatorname{div}\tilde{\mathbf{v}}_h) \quad \forall\ \tilde{\mathbf{v}}_h \in \tilde{\mathbb{V}}_h.$$

This is the same as (5.6).

Thanks to the stability assumption (5.5) the bilinear form $\mathcal{B}$ is stable with respect to the norm $(\nu\|\mathbf{u}\|^2 + \nu^{-1}\|p\|^2)^{-\frac{1}{2}}$ (this can be easily verified by rescaling the problem to a Stokes problem with the penalty term $\nu(p,q)$). Thus, for the solution of the problem (5.9) we obtain the estimate

$$\begin{aligned}(5.13)\quad (\nu\|\tilde{\mathbf{u}}_h\|^2 + \nu^{-1}\|\tilde{p}_h\|^2)^{-\frac{1}{2}} &\le C \sup_{\tilde{\mathbf{v}}_h, \tilde{q}_h \in \mathbb{V}_h \times \mathbb{Q}_h} \frac{\mathcal{B}(\tilde{\mathbf{u}}_h, \tilde{p}_h; \tilde{\mathbf{v}}_h, \tilde{q}_h)}{(\nu\|\tilde{\mathbf{v}}_h\|^2 + \nu^{-1}\|\tilde{q}_h\|^2)^{-\frac{1}{2}}} \\ &= C \sup_{\tilde{\mathbf{v}}_h, \tilde{q}_h \in \mathbb{V}_h \times \mathbb{Q}_h} \frac{\mathcal{B}(\mathbf{u}_H, p_H; \tilde{\mathbf{v}}_h, \tilde{q}_h)}{(\nu\|\tilde{\mathbf{v}}_h\|^2 + \nu^{-1}\|\tilde{q}_h\|^2)^{-\frac{1}{2}}} \le 2C\,\nu\|\mathbf{u}_H\|.\end{aligned}$$

Using the triangle inequality and (5.13) we get

$$\nu\|\mathbf{u}_H - \tilde{\mathbf{u}}_h\|^2 + \|p_H - \tilde{p}_h\|^2 \le C\,((\nu + \nu^2)\|\mathbf{u}_H\|^2 + \|p_H\|^2).$$

Thanks to (5.8) and (5.12) the last inequality is equivalent to

$$\nu\|\mathbf{u}_H - \tilde{\mathbf{u}}_h\|^2 + \|P_h \operatorname{div}(\mathbf{u}_H - \tilde{\mathbf{u}}_h)\|^2 \le C\,(\nu\|\mathbf{u}_H\|^2 + \|P_H \operatorname{div}\mathbf{u}_H\|^2),$$

which is (5.3) for the prolongation defined in (5.7). □

Note that for isoP2-P0 or P2-P0 finite elements, the spaces $\tilde{\mathbb{Q}}_h$ and the appropriate $\tilde{\mathbb{V}}_h$ are easy to describe. Indeed, let T_H be the coarse grid (pressure) triangulation. Then

$$\begin{aligned}\tilde{\mathbb{Q}}_h &= \left\{\tilde{q}_h \in \mathbb{Q}_h : \int_\tau \tilde{q}_h\,dx = 0 \quad \forall\ \tau \in T_H\right\}, \\ \tilde{\mathbb{V}}_h &= \left\{\tilde{\mathbf{v}}_h \in \mathbb{V}_h : \tilde{\mathbf{v}}_h|_{\partial\tau} = 0 \quad \forall\ \tau \in T_H\right\}.\end{aligned}$$

The assumptions of Lemma 5.1 can be readily checked. For this choice of the subspace $\tilde{\mathbb{V}}_h$, solving (5.6) requires the solution of N_p independent problems of small dimension (for regular subdivisions the dimension is 6 in two dimensions, and 9 in three dimensions), where N_p is the number of elements in T_H. On the other hand, for isoP2-P1 or P2-P1 finite elements we are unable to find a simple choice for $\tilde{\mathbb{V}}_h$ leading to easily solved local problems (5.6). Therefore, in this case we compute the correction $\tilde{\mathbf{v}}_h$ in (5.7) by solving locally (5.6) on each element $\tau \in T_H$, taking for $\tilde{\mathbb{V}}_h$ all $\mathbf{v}_h \in \mathbb{V}_h$ vanishing on $\partial\tau$. From experiments we saw that such a prolongation still improves the robustness of the multigrid method in the case of continuous pressure discretization compared to the canonical prolongation ($\tilde{\mathbf{u}}_h = 0$). Moreover, some further improvement in this case was observed if the $\mathbf{w}$-dependent terms were included in the definition of the bilinear form a_h in (5.6). However, the observed improvement in this case was not as dramatic as for isoP2-P0 elements; see also Remark 5.2.

Typically the restriction operator $r : \mathbb{V}_h \to \mathbb{V}_H$ is taken to be the adjoint of the prolongation. This choice is also convenient for theoretical analysis [29]. However, an operator adjoint to (5.7) is not easily computable. Therefore, as a restriction we use the L^2-orthogonal projection from $\mathbb{V}_h$ to $\mathbb{V}_H$. It is well known that the L^2-orthogonal projection on $\mathbb{V}_H$ is stable in the H^1-norm (see, e.g., [4]) under some assumptions on T_H. Although we did not find in the literature any results which would allow us

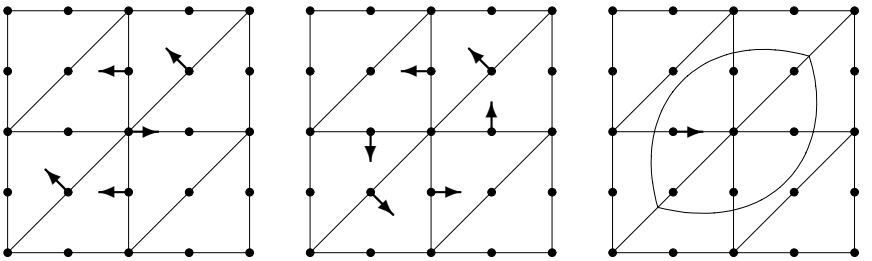


FIG. 5.1. *Basis of weakly divergence-free functions for isoP2-P0 and P2-P0. Pressures from $\mathbb{Q}_h$ are constants on each triangle; dots denote velocity nodes.*

to conclude the stability of the L^2-orthogonal projection with respect to the norms $\|\cdot\|_h$, $\|\cdot\|_H$ if $\nu \to 0$, this choice works very well as the restriction operator in our multigrid method.

Finally, we describe the smoother. As already mentioned, an effective smoother should effectively eliminate the oscillatory components in the kernel of $P_h \operatorname{div}$. For piecewise constant pressure-based FE pairs one finds that $\mathbf{u}_h \in \ker(P_h \operatorname{div})$ if and only if for any $\tau \in T_h$ the total flux for $\mathbf{u}_h$ through $\partial\tau$ equals zero. Thus, the kernel of $P_h \operatorname{div}$ can be described explicitly. For example, basis functions for $\ker(P_h \operatorname{div})$ in the case of P2-P0 finite elements are shown in Figure 5.1 [5, 29] (these are actually three of nine basis functions "captured" inside the oval on the right). Note that all basis functions have local support. Therefore an efficient smoother can be built as a block relaxation procedure, where degrees of freedom (DOFs) supporting each basis function from $\ker(P_h \operatorname{div})$ are organized in one block. The blocks can be overlapping. This is described more formally below.

We used a block Gauss–Seidel method, which can be written as a sequential subspace correction method (in the terminology of Xu [36]) based on the decomposition $\mathbb{V}_h = \sum_{i=0}^{l} \mathbb{V}_i$, $\mathbb{V}_i \subset \mathbb{V}_h$, where the sum is not necessarily a direct sum. Let A_i be the stiffness matrix for (5.2) on $\mathbb{V}_i$. Then one relaxation iteration consists of the following steps: Let $u^0 = u^{old}$, then compute

$$u^{i+1} = u^i - p_i A_i^{-1} r_i((A + B^T M_p^{-1} B)\, u^i - f) \quad \text{for } i = 0, \dots, l, \tag{5.14}$$

and put $u^{new} = u^{l+1}$. Restriction r_i in (5.14) is the simple nodal one and $p_i = r_i^T$.

For isoP2-P0 or P2-P0 finite elements the natural choice is to gather nodal DOFs for velocity inside ovals similar to the one in Figure 5.1. To be more precise, let s_i, $i = 1, \dots, l$, be the set of all interior vertices of T_h. For each i define $\boldsymbol{\tau}^i$ as a union of all triangles $\tau \in T_h$ sharing s_i. Then

$$\mathbb{V}_i := \{\mathbf{v}_h \in \mathbb{V}_h : \mathbf{v}_h = 0 \quad \text{in } \Omega \setminus \boldsymbol{\tau}^i\}.$$

We note that in the case of piecewise linear pressure FE pairs (e.g., isoP2-P1) it is harder to describe the kernel of $P_h \operatorname{div}$. Thus, in this case we simply use the same choice of $\mathbb{V}^i$ to define the relaxation step (5.14). As we will see, the resulting smoother remains quite effective.

As usual for Gauss–Seidel type methods, the performance of the iterations (5.14) depends on the ordering of blocks of unknowns. Since the definition of $\mathbb{V}^i$ is solely based on the choice of a node s_i, the relaxation method (5.14) is uniquely defined by the ordering of nodes in T_h. It is well known that smoothing iterations for convection-dominated problems are sensitive to the numbering of unknowns and that the fastest

convergence is achieved for the streamwise ordering of nodes; see, e.g., [3, 18, 23]. Finding appropriate orderings may be a difficult task, especially for complex three-dimensional (3D) flow fields $\mathbf{w}$. However, our numerical experiments suggest that the multigrid method with the above-described block Gauss–Seidel smoothing applied to (5.1) is not very sensitive to the numbering of nodes. Because of this, in our implementation we used a lexicographical ordering of the blocks of unknowns. One smoothing iteration consists of two relaxation steps (5.14): one with left-right/top-bottom ordering, the other with top-bottom/left-right ordering.

We conclude the section with some remarks about the implementation and performance of the multigrid scheme.

REMARK 5.2. While the multigrid method described above turns out to be very robust for (iso)P2-P0 finite elements, in the case of (iso)P2-P1 discretization some degradation of convergence is still observed as $\frac{\nu}{\gamma} \to 0$. This results in a large number of outer iterations. One way to improve convergence is to increase the number of smoothing steps on each level as $\frac{\nu}{\gamma} \to 0$. Another, more efficient way (followed in the numerical experiments; see section 6) is to take somewhat smaller γ for $\nu \ll 1$, keeping the number of smoothing steps constant and low. Doing this, one ensures that the cost of one outer iteration does not increase. At the same time one sacrifices slightly the quality of the Schur complement preconditioner in (3.7): Indeed, for $\gamma = 0$ the preconditioner from (3.7) is poor for small ν (see [8]). Nevertheless, the total number of outer iterations stays quite low for all reasonable values of ν (see section 6).

REMARK 5.3. In general we found the W-cycle to be more robust than the V-cycle. For the problems we tested, the V-cycle needs approximately twice or three times as many smoothing steps as the W-cycle to demonstrate similar convergence rates. Thus, the results in the next section are those obtained with the W-cycle implementation. However, for problems with locally refined grids using the V-cycle instead may be preferable in order to preserve the optimal complexity of the multigrid method.

REMARK 5.4. The dimension of the subspace $\mathbb{V}_i$ equals 14 in the 2D case (two velocity components for each node inside the oval in Figure 5.1). In the 3D case and for a regular subdivision we get $\dim \mathbb{V}_i = 39$. It is a common practice in using block smoothers for flow problems (such as Vanka smoothing [33]) that instead of applying A_i^{-1} in (5.14) one considers a simple, e.g., diagonal or triangular, approximation of A_i [31, section 2.4.1]. This helps reduce significantly the computational cost of one smoothing iteration. In this paper we solve subproblems with A_i exactly. A multigrid scheme based on inexact block inverses will be studied elsewhere.

6. Numerical experiments. In this section we present results for both isoP2-P0 and isoP2-P1 elements. We focus on two commonly used model problems on the unit square. In the first problem the wind function is constant and is given by

$$\mathbf{w} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}. \tag{6.1}$$

The second problem is a classical recirculating flow problem. Here the wind is a "rotating vortex," i.e., the vector field

$$\mathbf{w} = \begin{pmatrix} 4(2y-1)(1-x)x \\ -4(2x-1)(1-y)y \end{pmatrix}. \tag{6.2}$$

In the experiments we use our preconditioner with the Krylov subspace method BiCGStab [32]. Inexact solves involving the (1,1) block consist of a single W(1,1)-cycle with the multigrid method described in the previous section, that is, one W-cycle

TABLE 6.1
Results for AL approach, isoP2-P0 finite elements.

Mesh size h	Viscosity ν				
	1	0.1	0.01	10^{-3}	10^{-4}
w is constant wind (6.1)					
1/16	7	5	5	6	6
1/32	7	5	6	7	8
1/64	5	5	6	5	7
1/128	5	5	5	5	6
w is rotating vortex (6.2)					
1/16	5	5	6	10	15
1/32	4	4	5	10	21
1/64	4	4	5	9	18
1/128	4	5	5	7	14

Number of preconditioned BiCGStab iterations
($\hat{A}_\gamma^{-1}$ is one W(1,1)-cycle, $\gamma = 1$).

with one presmoothing and one postsmoothing step on each level. Each smoothing step consists of two iterations of block Gauss–Seidel relaxation with the alternating ordering described at the end of the previous section. The number of levels is such that the coarsest grid problem corresponds to $h = \frac{1}{2}$. The coarsest grid problem is solved exactly.

We set $W = \bar{M}_p$ in all experiments. Solves involving the (2,2) block are inexpensive. The action of $\hat{S}^{-1}$ on a vector is computed via (3.7). For isoP2-P0 elements this amounts to a diagonal scaling operation. For the case of isoP2-P1 elements, the pressure mass matrix is not diagonal. Linear systems involving M_p are approximately solved by performing a fixed number (fifteen) of stationary Richardson-type relaxations with parameter $\alpha = 1.75$. Note that the condition number of M_p is bounded by a constant independent of h (and of ν). Since we use a fixed number of stationary iterations for both the (1,1) and the (2,2) blocks, the overall preconditioner remains constant from one outer iteration to the next. Hence, we can use a standard Krylov subspace method such as BiCGStab for the outer iteration. In all runs we stop the outer iteration once the 2-norm of the initial residual has been reduced by at least six orders of magnitude, and we always use the zero vector as the initial guess.

The results for isoP2-P0 elements are reported in Table 6.1. For both choices of the wind function, the convergence rate is h-independent. For the constant wind problem, the convergence rate is also independent of ν, even for values of ν as small as 10^{-4}. For the recirculating flow problem, the results are virtually the same except for an increase in the number of iterations for the smallest value of the viscosity, $\nu = 10^{-4}$. Even in this extreme case, however, the convergence is quite fast.

Results for the isoP2-P1 discretization are reported in Table 6.2. The results are still quite good. For the smallest values of the viscosity we found that γ must be reduced in order to retain fast convergence (see Remarks 4.3 and 5.2 for a discussion). The results are, in all cases, h-independent; some increase in the iteration count occurs for the smallest values of ν, but overall the convergence properties remain excellent.

6.1. Other methods. In this subsection we report the results obtained when solving the same model problems with the [illegible] (PCD) preconditioner developed in [21] and with a full [illegible] method as a preconditioner (see, e.g., [19, 33]). Both techniques are based on the original, nonaugmented

TABLE 6.2
Results for AL approach, isoP2-P1 finite elements.

Mesh size h	Viscosity ν				
	1	0.1	0.01	10^{-3}	10^{-4}
	Parameter γ				
	1	1	1	0.1	0.02
w is constant wind (6.1)					
1/16	6	6	7	8	24
1/32	7	6	10	8	22
1/64	7	6	8	7	19
1/128	7	6	9	9	18
w is rotating vortex (6.2)					
1/16	6	6	7	13	25
1/32	5	6	9	11	32
1/64	4	5	10	11	37
1/128	4	4	10	12	34

Number of preconditioned BiCGStab iterations
($\hat{A}_\gamma^{-1}$ is one W(1,1)-cycle).

formulation ($\gamma = 0$). The PCG preconditioner is built upon separately solving velocity and pressure subproblems, while Vanka's approach is a geometric multigrid method for a coupled problem (1.10); see the review paper [35] discussing this class of methods.

The PCD method is widely regarded as one of the best available preconditioners in the class of block solvers for the steady Oseen problem, at least in the case of continuous pressure approximations. The preconditioner is of the form (3.1), where $\hat{S}$, the approximate pressure Schur complement, is defined through the action of its inverse:

$$\hat{S}^{-1} = M_p^{-1} F_p A_p^{-1}. \tag{6.3}$$

In (6.3) M_p denotes the pressure mass matrix, F_p denotes a convection-diffusion operator acting on the pressure space, and A_p represents the pressure Laplacian; for the latter two operators, care should be taken to enforce the appropriate boundary conditions. For the given model problems we used Neumann conditions on the entire boundary. In the case of piecewise linear pressure approximations F_p was constructed following [10, page 348]. For piecewise constant pressure elements we set $A_p := BM_u^{-1}B^T$ (so-called mixed approximation of the pressure Poisson problem) and $F_p := r_{u\to p}\, A_x\, r_{p\to u}$, where A_x is the x-subblock of A, $r_{u\to p}$ and $r_{p\to u}$ are suitable mappings from $\mathbb{Q}_h$ to $\mathbb{V}_h$, and vice versa.

Note that in two dimensions each application of the PCD preconditioner requires solving two convection-diffusion problems for the velocities and one Poisson-type problem for the pressure. These solves are usually performed inexactly in order to reduce costs. In our implementation we used [illegible] solves for both the (1,1) block and for the inverses appearing in (6.3).

As recommended in [10], we use the PCD preconditioner with BiCGStab(2) [30]. This Krylov subspace method requires four matrix-vector multiplies and four applications of the preconditioner per iteration. In order to facilitate the comparison with the results obtained with BiCGStab and our own preconditioner, we report in Table 6.3 the number of "BiCGStab-equivalent iterations" (i.e., the number of BiCGStab(2)

TABLE 6.3
Results for PCD preconditioner, isoP2-P0/isoP2-P1 finite elements.

Mesh size h	Viscosity ν				
	1	0.1	0.01	10^{-3}	10^{-4}
w is constant wind (6.1)					
1/16	6 / 12	8 / 16	12 / 24	30 / 34	100 / 80
1/32	6 / 10	10 / 16	14 / 24	24 / 28	86 / 92
1/64	6 / 10	8 / 14	16 / 24	22 / 32	64 / 66
1/128	6 / 10	8 / 12	16 / 26	24 / 36	64 / 58
w is rotating vortex (6.2)					
1/16	6 / 8	10 / 12	30 / 40	> 400 / 188	
1/32	6 / 8	10 / 12	30 / 40	> 400 / 378	
1/64	4 / 6	8 / 12	26 / 40	> 400 / > 400	
1/128	4 / 6	8 / 10	22 / 44	228 / > 400	

Number of preconditioned BiCGStab-equivalent iterations (exact solves).

iterations multiplied by 2). We also experimented with BiCGStab, but the results were not better.

It appears from these results that the PCD preconditioner is generally less robust and effective than the AL-based preconditioner, especially for small ν. We note, however, that PCD preconditioners can be built on a basis of various solvers for the (1,1) block and for the pressure Poisson problem; in particular, algebraic multigrid methods can be used, while the efficient implementation of the AL-based preconditioner described in this paper requires a hierarchy of grids to be available, as usual for a geometric multigrid method. Also, it should be mentioned that the PCD preconditioner results in preconditioned operators for which some of the eigenvalues have an imaginary part growing like $O(\nu^{-1})$; see [9]. It is well known that BiCG-type methods tend to perform poorly for such problems. Thus, it is possible that better results could be obtained for small ν if PCD preconditioning is used with GMRES or with BiCGStab(ℓ) with larger values of ℓ.

The Vanka preconditioner is built on a geometric multigrid method with block Gauss–Seidel smoothing for system (1.10). One relaxation step consists of l sequential updates:

$$\begin{pmatrix} u^{i+1} \\ p^{i+1} \end{pmatrix} = \begin{pmatrix} u^{i} \\ p^{i} \end{pmatrix} - r_i^T \mathcal{C}_i^{-1} r_i \left(\mathcal{A} \begin{pmatrix} u^{i} \\ p^{i} \end{pmatrix} - \begin{pmatrix} f \\ 0 \end{pmatrix} \right) \quad \text{for } i = 0, \dots, l. \tag{6.4}$$

Restriction r_i is the simple nodal one. Entries of matrices $\mathcal{C}_i$ are those from $\mathcal{A}$ corresponding to unknowns in the ith block. For approximations based on discontinuous pressure, unknowns in one block consist of all pressure and velocity DOFs tied to one cell (triangle). For isoP2-P0 elements this results in a matrix of the form

$$\mathcal{C}_i = \begin{pmatrix} A_i & B_i^T \\ B_i & 0 \end{pmatrix}, \tag{6.5}$$

where the block A_i is 12×12 and B_i is 1×12. Since for a regular subdivision in two dimensions the number of cells is approximately twice the number of vertices and the dimension of A_i in (5.14) is 14×14, we conclude that for isoP2-P0 elements one smoothing step in Vanka's method is more computationally expensive than in the multigrid method from section 5.

TABLE 6.4
Vanka multigrid preconditioner, isoP2-P0/isoP2-P1 finite elements.

Mesh size h	Viscosity ν				
	1	0.1	0.01	10^{-3}	10^{-4}
w is constant wind (6.1)					
1/16	4 / 6	4 / 6	3 / 10	4 / 18	5 / 20
1/32	4 / 6	4 / 6	3 / 8	4 / 22	4 / 33
1/64	4 / 6	4 / 6	4 / 8	3 / 21	5 / 56
1/128	4 / 5	4 / 5	4 / 6	3 / 14	4 / 64
w is rotating vortex (6.2)					
1/16	4 / 4	5 / 5	5 / 7	8 / 19	13 / 30
1/32	4 / 4	4 / 5	6 / 6	10 / 20	17 / 38
1/64	4 / 4	4 / 4	5 / 5	8 / 18	40 / 59
1/128	4 / 4	4 / 4	4 / 5	9 / 13	> 400 / 67

Number of preconditioned BiCGStab iterations.

For approximations based on the continuous pressure it is recommended in [19] that unknowns in a block should consist of all the pressure and velocity DOFs tied to one vertex. For isoP2-P1 elements this results in a matrix of the same structure as (6.5). However, now each matrix $\mathcal{C}_i$ has dimension 39×39 for a regular 2D triangulation (196×196 in three dimensions). This makes one relaxation step considerably more expensive than in (5.14). To facilitate comparison we include in a block only velocity DOFs in the corresponding vertex and next to it (i.e., exactly the same as in the multigrid method described in section 5). Taking into account one pressure unknown in the vertex, this results in a 15×15 matrix $\mathcal{C}_i$. Thus the computational cost of one smoothing step is nearly the same as in the multigrid method used in the AL-based preconditioner. One problem with this reduction is that now the $\mathcal{C}_i$ matrix for isoP2-P1 is LBB-unstable and the relaxation fails to smooth the error in the pressure. As typical for stabilized methods, the solution is to replace the zero (2,2) "block" in (6.5) (its dimension is actually 1×1) by the matrix $-\sigma(\mathbf{x})I$, where $\sigma(\mathbf{x})$ is an element-dependent stabilization parameter. Numerical experiments reveal that multigrid convergence is very sensitive to the value of $\sigma(\mathbf{x})$. An ad hoc choice which we found to be close to optimal for various values of h and ν is $\sigma(\mathbf{x}) = (\nu/2 + 2\|\mathbf{w}(\mathbf{x})\|h^2)^{-1}$, where $\|\mathbf{w}(\mathbf{x})\|$ is the Euclidean norm of vector $\mathbf{w}(\mathbf{x})$ in the corresponding vertex.

Results for the Vanka multigrid preconditioner and both FE pairs are shown in Table 6.4. Similar to the multigrid used in the AL-based method, one smoothing iteration in Vanka consists of two relaxation steps with alternating ordering directions.

From the results reported in the table, it appears that the Vanka-type multigrid preconditioner gives good results for a wide range of values of h and ν. For very small values of the viscosity, however, the performance of the method may in some cases deteriorate, and the preconditioned iteration may even fail to converge.

7. Conclusions. The paper presents a new preconditioning technique for finite element discretizations of the Oseen problem arising from Picard linearizations of the steady Navier–Stokes equations. The approach is based on an AL formulation of the resulting saddle point system. The preconditioner is block triangular and requires an efficient approximate solver for the (1,1) block; the second block is essentially a scaled mass matrix and is easy to invert. The (1,1) block can be seen as the discretization of a nonconventional, nonsymmetric operator consisting of a linear elasticity operator plus a convection term. We have developed an effective inexact solver for this opera-

tor by combining a multigrid method of Schöberl (originally designed for symmetric problems) with a suitable smoother. We have shown that assuming exact solves for the (1,1) block, the preconditioner clusters the eigenvalues in a small rectangular region of the complex plane around the point $(1, 0)$, and that the sides of this rectangle are independent of both h and ν. Our numerical experiments on some 2D model problems indicate that in practice, a single W(1,1)-cycle is sufficient to achieve fast, h-independent convergence of the outer BiCGStab iteration. Furthermore, the rate of convergence appears to be essentially ν-independent, except for a slight degradation for extremely low values of ν. The preconditioner is especially effective for isoP2-P0 and P2-P0 elements, but excellent performance is also observed for discretizations using piecewise linear pressure approximations. In the latter case, a good choice of the parameter γ is important for $\nu \ll 1$.

Finally, we mention that while in this paper we have focused on the steady 2D case, our approach can be extended to the solution of unsteady and 3D problems.

Acknowledgments. We would like to thank Clark Dohrmann for his helpful comments on an earlier version of the paper. Thanks also to David Silvester, Andy Wathen, and an anonymous referee for their careful reading of the manuscript. This work was done while the second author was visiting the Department of Mathematics and Computer Science at Emory University during fall 2005. The hospitality and support of Emory University are gratefully acknowledged.

REFERENCES

[1] I. Babuška and M. Suri, *Locking effects in the finite element approximation of elasticity problems*, Numer. Math., 62 (1992), pp. 439–463.

[2] M. Benzi, G. H. Golub, and J. Liesen, *Numerical solution of saddle point problems*, Acta Numer., 14 (2005), pp. 1–137.

[3] J. Bey and G. Wittum, *Downwind numbering: Robust multigrid for convection-diffusion problems*, Appl. Numer. Math., 23 (1997), pp. 177–192.

[4] J. H. Bramble, J. E. Pasciak, and O. Steinbach, *On the stability of the L^2 projection in $H^1(\Omega)$*, Math. Comp., 71 (2001), pp. 147–156.

[5] F. Brezzi and M. Fortin, *Mixed and Hybrid Finite Element Methods*, Springer, New York, 1991.

[6] A. N. Brooks and T. J. R. Hughes, *Streamline upwind/Petrov–Galerkin formulation for convection dominated flows with particular emphasis on the incompressible Navier–Stokes equations*, Comput. Methods Appl. Mech. Engrg., 32 (1982), pp. 199–259.

[7] C. R. Dohrmann and R. B. Lehoucq, *A primal-based penalty preconditioner for elliptic saddle point systems*, SIAM J. Numer. Anal., 44 (2006), pp. 270–282.

[8] H. Elman and D. Silvester, *Fast nonsymmetric iterations and preconditioning for the Navier–Stokes equations*, SIAM J. Sci. Comput., 17 (1996), pp. 33–46.

[9] H. C. Elman, D. J. Silvester, and A. J. Wathen, *Performance and analysis of saddle point preconditioners for the discrete steady-state Navier–Stokes equations*, Numer. Math., 90 (2002), pp. 665–688.

[10] H. C. Elman, D. J. Silvester, and A. J. Wathen, *Finite Elements and Fast Iterative Solvers: With Applications in Incompressible Fluid Dynamics*, Numer. Math. Sci. Comput., Oxford University Press, Oxford, UK, 2005.

[11] B. Fischer, A. Ramage, D. J. Silvester, and A. J. Wathen, *On parameter choice and iterative convergence for stabilised discretisations of advection-diffusion problems*, Comput. Methods Appl. Mech. Engrg., 179 (1999), pp. 179–195.

[12] R. Fletcher, *Practical Methods of Optimization*, 2nd ed., Wiley & Sons, Chichester, UK, 1987.

[13] M. Fortin and R. Glowinski, *Augmented Lagrangian Methods: Applications to the Numerical Solution of Boundary-Value Problems*, Stud. Math. Appl. 15, North–Holland, Amsterdam, New York, Oxford, 1983.

[14] D. K. Gartling and C. R. Dohrmann, *Quadratic finite elements and incompressible viscous flow*, Comput. Methods Appl. Mech. Engrg., 195 (2006), pp. 1692–1708.

[15] G. H. Golub and C. Greif, *On solving block-structured indefinite linear systems*, SIAM J. Sci. Comput., 24 (2003), pp. 2076–2092.

[16] A. Greenbaum, *Iterative Methods for Solving Linear Systems*, Frontiers Appl. Math. 17, SIAM, Philadelphia, 1997.

[17] C. Greif and D. Schötzau, *Preconditioners for the discretized time-harmonic Maxwell equations in mixed form*, Numer. Linear Algebra Appl., to appear.

[18] W. Hackbusch and T. Probst, *Downwind Gauss–Seidel smoothing for convection dominated problems*, Numer. Linear Algebra Appl., 4 (1997), pp. 85–102.

[19] V. John, *Higher order finite element methods and multigrid solvers in a benchmark problem for the 3D Navier–Stokes equations*, Internat. J. Numer. Methods Fluids, 40 (2002), pp. 775–798.

[20] V. John and G. Matthies, *Higher order finite element discretizations in a benchmark problem for incompressible flows*, Internat. J. Numer. Methods Fluids, 37 (2001), pp. 885–903.

[21] D. Kay, D. Loghin, and A. Wathen, *A preconditioner for the steady-state Navier–Stokes equations*, SIAM J. Sci. Comput., 24 (2002), pp. 237–256.

[22] G. Lube, *Stabilized Galerkin finite element methods for convection dominated and incompressible flow problems*, in Numerical Analysis and Mathematical Modelling, Banach Center Publ. 29, Polish Academy of Sciences, Warsaw, 1994, pp. 85–104.

[23] M. A. Olshanskii and A. Reusken, *Convergence analysis of a multigrid method for a convection-dominated model problem*, SIAM J. Numer. Anal., 42 (2004), pp. 1261–1291.

[24] M. A. Olshanskii and A. Reusken, *Grad-div stabilization for Stokes equations*, Math. Comp., 73 (2004), pp. 1699–1718.

[25] M. A. Olshanskii, *A low order Galerkin finite element method for the Navier–Stokes equations of steady incompressible flow: A stabilization issue and iterative methods*, Comput. Methods Appl. Mech. Engrg., 191 (2002), pp. 5515–5536.

[26] A. Quarteroni and A. Valli, *Numerical Approximation of Partial Differential Equations*, Springer Ser. Comput. Math. 23, Springer, Berlin, 1994.

[27] A. Ramage, *A multigrid preconditioner for stabilised discretisations of advection-diffusion problems*, J. Comput. Appl. Math., 110 (1999), pp. 187–203.

[28] H. G. Roos, M. Stynes, and L. Tobishka, *Numerical Methods for Singularly Perturbed Equations: Convection-Diffusion and Flow Problems*, Springer, Berlin, 1996.

[29] J. Schöberl, *Multigrid methods for a parameter dependent problem in primal variables*, Numer. Math., 84 (1999), pp. 97–119.

[30] G. L. G. Sleijpen and D. R. Fokkema, *BiCGStab(ℓ) for linear equations involving unsymmetric matrices with complex spectrum*, Electron. Trans. Numer. Anal., 1 (1993), pp. 11–32.

[31] S. Turek, *Efficient Solvers for Incompressible Flow Problems. An Algorithmic and Computational Approach*, Lect. Notes Comput. Sci. Eng. 6, Springer, Berlin, Heidelberg, New York, 1999.

[32] H. A. van der Vorst, *Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 13 (1992), pp. 631–644.

[33] S. P. Vanka, *Block-implicit multigrid solution of Navier–Stokes equations in primitive variables*, J. Comput. Phys., 65 (1986), pp. 138–158.

[34] A. Wathen, *Realistic eigenvalues bounds for the Galerkin mass matrix*, IMA J. Numer. Anal., 7 (1987), pp. 449–457.

[35] P. Wesseling and C. W. Oosterlee, *Geometric multigrid with applications to computational fluid dynamics*, J. Comput. Appl. Math., 128 (2001) pp. 311–334.

[36] J. Xu, *Iterative methods by space decomposition and subspace correction*, SIAM Rev., 34 (1992), pp. 581–613.

SIAM J. SCI. COMPUT.
Vol. 28, No. 6, pp. 2114–2138
© 2006 Society for Industrial and Applied Mathematics

AN ADAPTIVE WEM ALGORITHM FOR SOLVING ELLIPTIC BOUNDARY VALUE PROBLEMS IN FAIRLY GENERAL DOMAINS*

S. BERRONE† AND T. KOZUBEK‡

Abstract. In this paper, we introduce a simple adaptive wavelet element algorithm similar to the Cohen–Dahmen–DeVore algorithm [A. Cohen, W. Dahmen, and R. DeVore, *Math. Comp.*, 70 (2001), pp. 27–75]. The main difference is that we do not assume knowledge of the many constants appearing therein. The algorithm is easy to implement and applicable to a large class of problems in fairly general domains. The efficiency is illustrated by several two-dimensional numerical examples and compared with an adaptive finite element method.

Key words. adaptive wavelet and finite element methods, elliptic operator equations, rates of convergence

AMS subject classifications. 65T60, 42C40, 65N30, 65N50, 65N55

DOI. 10.1137/04062014X

1. Introduction.

Wavelets have become an efficient tool for solving many problems arising in theory and in practice. Let us mention, for example, signal analysis, compression of information, geometry representation, and numerical analysis. In this paper we deal with wavelets used for the solution of elliptic boundary value problems in $\mathbb{R}^d$, $d = 1, 2, 3$. Adaptivity is a key ingredient in modern numerical computation which helps to improve significantly the performance of the method. When dealing with the finite element method (FEM), adaptivity occurs in the context of grid refinement, in contrast to the function selection used in wavelet discretization methods such as the wavelet element method (WEM). A detailed discussion of the adaptive (nonlinear) refinement advantages with respect to the uniform (linear) refinement can be found in [12, 23].

We introduce an adaptive WEM algorithm applicable to a large class of problems defined in fairly general domains. The key ideas of our algorithm follow the fundamental Cohen–Dahmen–DeVore approach to wavelet adaptivity for PDEs [2, 12, 13], but we do not assume knowledge of the many constants appearing therein. Industrial applications of the Cohen–Dahmen–DeVore algorithm in all its aspects still seem to be quite hard. Here we propose an adaptive algorithm based on the same ideas, but whose practical application is easier. For the proposed algorithm we do not provide any proof of either convergence or optimality, as in [12]. Our goal is rather (i) CG, BiCG, GMRES (ii) (iii) Ω $\mathbb{R}^d$; (iv)

*Received by the editors October 24, 2005; accepted for publication (in revised form) May 31, 2006; published electronically December 5, 2006. This research was supported by the European project Breaking Complexity, HPRN-CT-2002-00286, by grants IAA1075402 and 1ET400300415 of the Grant Agency of the Czech Academy of Sciences, and by the Italian funds MIUR PRIN 2004 "Adattività e avanzamento in tempo nei modelli numerici alle derivate parziali" and INDAM-GNCS-2005 "Metodi numerici per lo studio di problemi evolutivi multiscala."

http://www.siam.org/journals/sisc/28-6/62014.html

†Dipartimento di Matematica, Politecnico di Torino, C.so Duca degli Abruzzi, 24, 10129 Torino, Italy (sberrone@calvino.polito.it).

‡Department of Applied Mathematics, VSB-Technical University of Ostrava, 17. listopadu 15, 708 33 Ostrava, Czech Republic (tomas.kozubek@vsb.cz).

To construct multiresolution systems on fairly general domains in $\mathbb{R}^d$, we use a WEM construction described in [10, 11, 4], but the proposed algorithm itself does not depend on the type of the used wavelet basis [20, 26] nor on the WEM construction [22, 14, 10]. The key point of the WEM is the splitting of the domain into subdomains in such a way that each of them can be mapped to a reference hypercube. Bases on the reference domain in $\mathbb{R}^d$ are constructed by the tensor product of scaling and wavelet functions defined on the unit interval. In our implementation wavelet bases on the interval are formed by biorthogonal B-spline wavelets with optimal localization introduced in [26]. To obtain a globally continuous biorthogonal wavelet basis, one has to enforce matching conditions across interelement boundaries. The construction enables one to fulfill additional important features such as symmetry, vanishing moments, and a minimal support.

Problem setting and general requirements on the wavelet bases are briefly mentioned in section 2. For space reasons we assume some knowledge of these aspects, referring for more details to [16, 17, 18] and the references therein. In section 3, we introduce the adaptive WEM algorithm and describe all ingredients in details. In section 4, we illustrate the applicability of this adaptive WEM method to quite general situations; some numerical examples involving singularities and two-dimensional complex geometries requiring mapping and matching are considered. Conclusions are summarized in section 5. Finally, in the appendix, we compare some results obtained by the proposed algorithm with results obtained using a well-assessed adaptive finite element code [30, 1]. This comparison does not attempt to claim that one method outperforms the other, but aims at comparing some aspects of the solution process of the two adaptive methods.

2. General settings. In this section, we introduce a class of elliptic boundary value problems which are the focus of our interest. Finally, using such basis, we show how to transform the problem into an equivalent l_2-problem.

2.1. Problem setting. Let Ω be a bounded domain in $\mathbb{R}^d$, $d = 1, 2, 3$, with the Lipschitz boundary $\partial\Omega$. Further, let $H := H(\Omega)$ be a Hilbert space and $\mathcal{A} : H \to H'$ an elliptic operator of the second order, where H' is a dual to H and $H \hookrightarrow L^2(\Omega) \hookrightarrow H'$. Due to simplicity of our presentation we will be concerned with the following abstract pure Dirichlet boundary value problem: for given $f \in H' := H^{-1}(\Omega)$ find $u \in H := H_0^1(\Omega)$ such that

$$\begin{cases} \mathcal{A}u = f & \text{in } \Omega, \\ u = 0 & \text{on } \partial\Omega. \end{cases} \tag{2.1}$$

The corresponding weak form to (2.1) is

$$\begin{cases} \text{Find } u \in H \text{ such that} \\ a(u, v) = \langle f, v \rangle \quad \forall v \in H, \end{cases} \tag{2.2}$$

where $a(.\,,.) : H \times H \to \mathbb{R}$ is the continuous bilinear form defined by $a(u, v) := \langle \mathcal{A}u, v \rangle$ and $\langle .\,,. \rangle$ denotes the standard duality pairing between H and H'.

We assume that problem (2.2) is well-posed, i.e., there are positive finite constants $c_{\mathcal{A}}$, $C_{\mathcal{A}}$ such that $c_{\mathcal{A}}\|v\|_H \le \|\mathcal{A}v\|_{H'} \le C_{\mathcal{A}}\|v\|_H$, $v \in H$. This immediately implies the existence of a unique solution to (2.2).

2.2. General requirements on the wavelet bases. We are interested in solving problem (2.2) using the [illegible], i.e., a Galerkin method with a suitably chosen wavelet basis for H in $\boldsymbol{\psi} = \{\psi_\lambda : \lambda \in \mathcal{J}\}$, where $\mathcal{J}$ is an infinite index set, and an adaptive procedure for the basis functions selection. In the following, we list the relevant properties of the wavelet basis $\boldsymbol{\psi}$.

For any $v \in L^2(\Omega)$ we have a [illegible] expansion $v = \sum_{\lambda\in\mathcal{J}} v_\lambda\psi_\lambda = \mathbf{v}^T\boldsymbol{\psi}$ which induces an isomorphism between H and l_2. More precisely, there is a scaling diagonal matrix $\mathbb{D} = \operatorname{diag}(d_\lambda : \lambda \in \mathcal{J})$ and positive constants c_ψ, C_ψ such that $c_\psi\|\mathbf{v}\|_{l_2} \le \|\mathbf{v}^T\mathbb{D}^{-1}\boldsymbol{\psi}\|_H \le C_\psi\|\mathbf{v}\|_{l_2}$. For example, when H is the Sobolev space $H^t(I)$ with $I = [0,1]$, a canonical choice of d_λ is $d_\lambda = 2^{t|\lambda|}$; in general for $H = H^1(\Omega)$ we choose $d_\lambda = \sqrt{a(\psi_\lambda,\psi_\lambda)}$ [19, 21, 27, 28]. We assume that basis functions have [illegible] and are [illegible] with $\operatorname{diam}(\operatorname{supp}\psi_\lambda) \sim 2^{-|\lambda|}$, where $\lambda \in \mathcal{J}$ and $|\lambda|$ is the level of the basis function ψ_λ (see [20, 26, 10]).

2.3. An equivalent l_2-problem. Given a basis with the mentioned properties, we can transform problem (2.2) over a function space H into an algebraic system of equations over the corresponding sequence space. More precisely, we obtain the following equivalent l_2-problem: $\mathbb{A}\mathbf{x} = \mathbf{f}$, where $\mathbb{A} = \mathbb{D}^{-1}\langle\mathcal{A}\boldsymbol{\psi},\boldsymbol{\psi}\rangle\mathbb{D}^{-1} = \left(d_\lambda^{-1}d_\mu^{-1}\langle\mathcal{A}\psi_\mu,\psi_\lambda\rangle\right)_{\lambda,\mu\in\mathcal{J}}$ is an infinite matrix and $\mathbf{f} = \mathbb{D}^{-1}\langle f,\boldsymbol{\psi}\rangle = \left(d_\lambda^{-1}\langle f,\psi_\lambda\rangle\right)_{\lambda\in\mathcal{J}}$ is an infinite vector. It is well known [12, 13, 16, 18] that if $u = \mathbf{x}^T\mathbb{D}^{-1}\boldsymbol{\psi}$ is the scaled wavelet representation of the solution to (2.2), the problem $\mathbb{A}\mathbf{x} = \mathbf{f}$ is an equivalent problem in l_2 to $\mathcal{A}u = f$. In addition, there are positive constants $c_{\mathbb{A}}$, $C_{\mathbb{A}}$ such that $c_{\mathbb{A}}\|\mathbf{v}\|_{l_2} \le \|\mathbb{A}\mathbf{v}\|_{l_2} \le C_{\mathbb{A}}\|\mathbf{v}\|_{l_2}$, $\mathbf{v} \in l_2$, with lower and upper estimates $c_{\mathbb{A}} \ge c_\psi^2 c_{\mathcal{A}}$ and $C_{\mathbb{A}} \le C_\psi^2 C_{\mathcal{A}}$, respectively.

Our infinitely dimensional problem $\mathbb{A}\mathbf{x} = \mathbf{f}$ can be now approximated by a finite-dimensional approximation

$$\mathbb{A}_{\Lambda\times\Lambda}\mathbf{x}_\Lambda = \mathbf{f}_\Lambda, \tag{2.3}$$

where $\mathbf{f}_\Lambda = \left(d_\lambda^{-1}\langle f,\psi_\lambda\rangle\right)_{\lambda\in\Lambda}$, $\mathbb{A}_{\Lambda\times\Lambda} = \left(d_\lambda^{-1}d_\mu^{-1}\langle\mathcal{A}\psi_\mu,\psi_\lambda\rangle\right)_{\lambda,\mu\in\Lambda}$, $\Lambda \subset \mathcal{J}$, and the approximation of the solution u is given by

$$u_\Lambda = \sum_{\mu\in\Lambda} d_\mu^{-1}x_\mu\psi_\mu = \mathbf{x}_\Lambda^T\mathbb{D}_\Lambda^{-1}\boldsymbol{\psi}_\Lambda. \tag{2.4}$$

With $\mathbb{A}$ positive definite, all finite sections $\mathbb{A}_{\Lambda\times\Lambda}$ have $\|\mathbb{A}_{\Lambda\times\Lambda}\|_2 \sim 1$, $\|\mathbb{A}_{\Lambda\times\Lambda}^{-1}\|_2 \sim 1$, and $1 \sim \operatorname{cond}_2(\mathbb{A}_{\Lambda\times\Lambda}) \le \operatorname{cond}_2(\mathbb{A}) < \infty$ [19, 21, 27].

3. An adaptive WEM algorithm. In this section, we present our adaptive algorithm; particular settings of the parameters are presented in section 4.

We start initializing all parameters of the algorithm and choosing an appropriate solver of the resulting algebraic systems. The algorithm consists of an outer adaptive iterative process, where all iterations consist of several steps. At iteration n, we have three sets of basis functions, whose sets of indices will be denoted by Λ_n, $\tilde{\Lambda}_n$, and $\overline{\Lambda}_n$. More precisely, Λ_n is the set of indices of the active basis functions, whereas $\tilde{\Lambda}_n$ consists of indices corresponding to the functions that can be activated in this iteration. In the definition of this set one can include essential information on the data of the problem to deal with particular situations. Finally, $\overline{\Lambda}_n$ is the set of indices

of basis functions after a thresholding step in which we remove from Λ_n the indices of those basis functions whose coefficients are less significant. We name the sets Λ_n, $\tilde{\Lambda}_n$ and the corresponding functions [illegible] and [illegible], respectively.

The first step of each iteration consists in enlarging the set $\overline{\Lambda}_{n-1}$ to get Λ_n. In the first iteration $n = 0$ of the adaptive procedure, we add the indices of all scaling functions at the starting scaling level to $\overline{\Lambda}_{-1} = \emptyset$. We obtain the active set Λ_0. In the next iterations, we use the set $\tilde{\Lambda}_{n-1}$ which is constructed by the indices of suitably chosen basis functions whose supports interact with the supports of the functions in $\overline{\Lambda}_{n-1}$. We compute the coefficients of the residual function corresponding to the solution of the previous outer iteration with respect to the dual basis functions whose primal functions are in $\tilde{\Lambda}_{n-1}$. Then, to get Λ_n, we add to $\overline{\Lambda}_{n-1}$ a set of suitably chosen indices $\delta\Lambda_{n-1} \subset \tilde{\Lambda}_{n-1}$ chosen in order to catch as much as possible the bulk of the residuum with the minimum number of new basis functions. We use all active functions corresponding to $\Lambda_n = \overline{\Lambda}_{n-1} \cup \delta\Lambda_{n-1}$ to assemble the stiffness matrix $\mathbb{A}_{\Lambda_n \times \Lambda_n}$ and the right-hand side vector $\mathbf{f}_{\Lambda_n}$. Using a chosen iterative method for the inner iterations, we improve the current approximation of the solution coefficients $\mathbf{x}_{\Lambda_n}$ by approximately solving the corresponding linear algebraic system $\mathbb{A}_{\Lambda_n \times \Lambda_n} \mathbf{x}_{\Lambda_n} = \mathbf{f}_{\Lambda_n}$. Further, we remove the indices of all negligible functions corresponding to the smallest (less significant) absolute entries of the coefficient vector $\mathbf{x}_{\Lambda_n}$ from Λ_n to keep $\overline{\Lambda}_n$ as small as possible. We check an appropriate stopping criterion for the outer iteration in terms of the reduction of the residuum, and if it is not satisfied, we repeat all of the above-mentioned steps. Otherwise, we accept the actual approximation of the coefficient vector as the final solution and stop iterations.

3.1. Computation of $\mathbb{A}_{\Lambda\times\Lambda}$ and $\mathbf{f}_\Lambda$. When we have chosen our finite set of indices Λ the solution of the linear system (2.3) gives the approximation u_Λ of the solution u by (2.4). An efficient approximation of the integrals involved in the definitions of the elements $a_{\lambda,\mu}$ of the stiffness matrix and f_λ of the right-hand side vector plays a fundamental role [8, 7]. In this subsection, we recall some results of [8] which are independent of the algorithm presented there for the approximation of the integrals. We introduce a relative precision $\mathrm{tol}_{int} < 1$ in computing the elements of the stiffness matrix and the right-hand side vector. Let us denote by $\mathbb{A}^*_{\Lambda\times\Lambda}$ and $\mathbf{f}^*_\Lambda$ the numerical approximations of the stiffness matrix and the right-hand side vector, respectively. Set $\mathbb{E}_\Lambda = \mathbb{A}_{\Lambda\times\Lambda} - \mathbb{A}^*_{\Lambda\times\Lambda}$, $\mathbf{h}_\Lambda = \mathbf{f}_\Lambda - \mathbf{f}^*_\Lambda$ and define

$$i(\lambda,\mu) = \begin{cases} 1 & \text{if } |\operatorname{supp}\psi_\lambda \cap \operatorname{supp}\psi_\mu| > 0, \\ 0 & \text{otherwise.} \end{cases}$$

PROPOSITION 3.1. [illegible] λ [illegible] μ [illegible] $l^1(\mathbb{N})$ [illegible] $\delta = (\delta_l)_{l\in\mathbb{N}}$ [illegible] $l^1(\mathbb{N})$ [illegible] $\mathbb{E}_\Lambda = (e_{\lambda,\mu})_{\lambda,\mu\in\Lambda}$ [illegible]

$$|e_{\lambda,\mu}| \precsim \mathrm{tol}_{int}\, i(\lambda,\mu)\delta_{||\lambda|-|\mu||} 2^{-d/2||\lambda|-|\mu||} \|\mathbb{A}_{\Lambda\times\Lambda}\|_2. \tag{3.1}$$

[illegible] $\|\mathbb{E}_\Lambda\|_2 \precsim \mathrm{tol}_{int}\|\mathbb{A}_{\Lambda\times\Lambda}\|_2$

$\mathbb{A}$ being positive definite, we simply have $\|\mathbb{E}_\Lambda\|_2 \precsim \mathrm{tol}_{int}$.

PROPOSITION 3.2. [illegible] λ [illegible] $l^2(\mathbb{N})$ [illegible] $\delta = (\delta_l)_{l\in\mathbb{N}}$ [illegible] $l^2(\mathbb{N})$ [illegible] $\mathbf{h}_\Lambda = (h_\lambda)_{\lambda\in\Lambda}$ [illegible]

$$|h_\lambda| \precsim \mathrm{tol}_{int}\delta_{|\lambda|} 2^{-d/2|\lambda|} \|\mathbf{f}_\Lambda\|_2. \tag{3.2}$$

$\|\mathbf{h}_\Lambda\|_2 \precsim \mathrm{tol}_{int}\|\mathbf{f}_\Lambda\|_2$.

In each iteration n we verify (3.2) with the approximation of $\|\mathbf{f}_\Lambda\|_2$ given by $\|\mathbf{f}_{\Lambda_n}\|_2$. In our adaptive algorithm we usually form a sequence of enlarging sets Λ_n, $n = 0, 1, \ldots$, so $\|\mathbf{f}_{\Lambda_n}\|_2$ is increasing and we do not need to recompute any previously computed elements of $\mathbf{f}_{\Lambda_k}$, $k = 0, \ldots, n-1$.

Under the assumptions of the previous propositions and for our finite-dimensional l_2-problem we can conclude that there is $\mathrm{tol}_{int_0} > 0$ such that for all $0 < \mathrm{tol}_{int} \leq \mathrm{tol}_{int_0}$ the following estimate holds:

$$\|\mathbf{x}_\Lambda - \mathbf{x}^*_\Lambda\|_2 \precsim \mathrm{tol}_{int}\|\mathbf{x}_\Lambda\|_2,$$

where $\mathbf{x}^*_\Lambda$ is the coefficient vector computed from the approximated linear system $\mathbb{A}^*_{\Lambda\times\Lambda}\mathbf{x}^*_\Lambda = \mathbf{f}^*_\Lambda$ and $u^*_\Lambda = (\mathbf{x}^*_\Lambda)^T\mathbb{D}^{-1}_\Lambda\boldsymbol{\psi}_\Lambda$ is the corresponding approximate solution.

In the following it will be useful to evaluate the effect of the approximation in computing the integrals on the norm of the residuum. Let us define $\mathcal{J}_n = \Lambda_n \cup \tilde{\Lambda}_n$, $\mathbf{r}_{\mathcal{J}_n} = \mathbf{f}_{\mathcal{J}_n} - \mathbb{A}_{\mathcal{J}_n\times\Lambda_n}\mathbf{x}_{\Lambda_n}$, and $\mathbf{r}^*_{\mathcal{J}_n} = \mathbf{f}^*_{\mathcal{J}_n} - \mathbb{A}^*_{\mathcal{J}_n\times\Lambda_n}\mathbf{x}^*_{\Lambda_n}$. According to the results of the previous propositions and properties such as the well conditioning and stability of the l_2-problem, we have

$$\|\mathbf{r}_{\mathcal{J}_n} - \mathbf{r}^*_{\mathcal{J}_n}\|_2 \precsim \mathrm{tol}_{int}\|\mathbf{f}_{\mathcal{J}_n}\|_2.$$

In our algorithm we want to find a solution $\mathbf{x}_\Lambda$ which satisfies $\|\mathbf{f}_\mathcal{J} - \mathbb{A}_{\mathcal{J}\times\Lambda}\mathbf{x}_\Lambda\|_{l_2} \leq \varepsilon\|\mathbf{f}_\mathcal{J}\|_{l_2}$. Therefore we propose to choose the parameter tol_{int} such that $\mathrm{tol}_{int}\|\mathbf{f}_{\mathcal{J}_n}\|_2$ is negligible with respect to $\varepsilon\|\mathbf{f}_{\mathcal{J}_n}\|_2$, i.e., $\mathrm{tol}_{int} \ll \varepsilon$.

Different approaches for computing entries of stiffness matrix and right-hand side can be found in [2, 3].

3.2. Initialization part. We choose an appropriate solver of the resulting linear algebraic systems, such as , and set its stopping tolerance tol_{grad}, which is an upper bound for the ratio between the final algebraic residuum and the algebraic residuum corresponding to the initial guess of the current outer iteration (see (3.3)). The choice of tol_{int} has to be done in such a way that the approximation of the integrals gives a negligible contribution in the approximation of the bulk of the residuum. Further, we set all remaining parameters: relative activating and thresholding tolerances tol_{on} and tol_{off} involved in adding and removing indices of basis functions to and from the active set Λ, respectively. Finally, we set the relative target accuracy ε of the adaptive process, i.e., the final reduction factor of the total residuum.

3.3. Activating new basis functions. In each iteration $n > 0$ we want to catch and reduce the bulk of the residuum $\mathbf{r} = \mathbf{f}_\mathcal{J} - \mathbb{A}_{\mathcal{J}\times\overline{\Lambda}_{n-1}}\overline{\mathbf{x}}_{\overline{\Lambda}_{n-1}}$ and improve the solution with its components corresponding to the most significant entries of the residuum. In practice, we choose those significant components of the residuum whose indices belong to the activable set $\tilde{\Lambda}_{n-1}$. This procedure is realized by sorting the absolute values of the entries of the vector $\tilde{\mathbf{r}}_{\tilde{\Lambda}_{n-1}} = \tilde{\mathbf{f}}_{\tilde{\Lambda}_{n-1}} - \mathbb{A}_{\tilde{\Lambda}_{n-1}\times\overline{\Lambda}_{n-1}}\overline{\mathbf{x}}_{\overline{\Lambda}_{n-1}}$ in decreasing order, summing successively the squares of the largest ones and adding the corresponding indices to $\delta\Lambda_{n-1}$, until the target percentage of $\|\tilde{\mathbf{r}}_{\tilde{\Lambda}_{n-1}}\|_2^2$ is reached, i.e., while $\sum_{\lambda\in\delta\Lambda_{n-1}}|\tilde{r}_\lambda|^2 \leq \mathrm{tol}^2_{on}\|\tilde{\mathbf{r}}_{\tilde{\Lambda}_{n-1}}\|_2^2$. Then we set $\Lambda_n = \overline{\Lambda}_{n-1} \cup \delta\Lambda_{n-1}$ and extend the old approximation of the coefficient vector $\overline{\mathbf{x}}_{\overline{\Lambda}_{n-1}}$ by zeros at the positions corresponding to indices $\lambda \in \delta\Lambda_{n-1}$ to get the initial coefficient vector $\mathbf{x}^{\mathrm{init}}_{\Lambda_n}$. In the first outer iteration $n = 0$ we have $\overline{\Lambda}_{-1} = \emptyset$, $\overline{\mathbf{x}}_{\overline{\Lambda}_{-1}} = 0$ and $\delta\Lambda_{-1}$ is the set of indices

of all scaling functions at the starting level j_0 greater than or equal to the minimum construction level of basis functions.

FUNCTION $[\Lambda_n, \delta\Lambda_{n-1}] =$ Activate$(\overline{\Lambda}_{n-1}, \tilde{\Lambda}_{n-1}, \tilde{\mathbf{r}}_{\tilde{\Lambda}_{n-1}}, \mathrm{tol}_{on})$.

(i) $N := \operatorname{card} \tilde{\Lambda}_{n-1}$ $\tilde{\mathbf{r}}_{\tilde{\Lambda}_{n-1}}$ $\lambda^* := \lambda^*(\tilde{\mathbf{r}}_{\tilde{\Lambda}_{n-1}}) = (\lambda_1^*, \lambda_2^*, \dots, \lambda_N^*)$ $\tilde{\mathbf{r}}^*_{\tilde{\Lambda}_{n-1}} = (|\tilde{r}_{\lambda_1^*}|, |\tilde{r}_{\lambda_2^*}|, \dots, |\tilde{r}_{\lambda_N^*}|)$ $\tilde{\mathbf{r}}_{\tilde{\Lambda}_{n-1}}$ $\|\tilde{\mathbf{r}}_{\tilde{\Lambda}_{n-1}}\|_2^2$

(ii) $k = 1, 2, \dots$ $\sum_{l=1}^{k} |\tilde{r}_{\lambda_l^*}|^2$ k $\mathrm{tol}_{on}^2 \|\tilde{\mathbf{r}}_{\tilde{\Lambda}_{n-1}}\|_2^2$ k $\delta\Lambda_{n-1} := \{\lambda_l^*;\ l = 1, \dots, k\}$ $\Lambda_n := \overline{\Lambda}_{n-1} \cup \delta\Lambda_{n-1}$

FUNCTION $\mathbf{x}^{\mathrm{init}}_{\Lambda_n} =$ Extend$(\overline{\mathbf{x}}_{\overline{\Lambda}_{n-1}}, \overline{\Lambda}_{n-1}, \delta\Lambda_{n-1})$.

(i) $\mathbf{x}^{\mathrm{init}}_{\Lambda_n}$ $\overline{\mathbf{x}}_{\overline{\Lambda}_{n-1}}$ $\overline{\Lambda}_{n-1}$ ψ_μ $\mu \in \delta\Lambda_{n-1}$

3.4. Assembling $\mathbf{f}_{\Lambda_n}$ and $\mathbb{A}_{\Lambda_n \times \Lambda_n}$. For a given Λ_n we need to assemble the right-hand side vector $\mathbf{f}_{\Lambda_n}$ and the stiffness matrix $\mathbb{A}_{\Lambda_n \times \Lambda_n}$. In the computation of $a_{\lambda,\mu}$ and f_λ we use numerical quadrature formulas that allow us to control the approximation of the elements as in Propositions 3.1 and 3.2.

Computing the entries of $\mathbf{f}_{\Lambda_n}$. We form a sequence $\{\mathbf{f}^{(J)}_{\Lambda_n}\}_{J \in \mathbb{N}}$ of approximations to $\mathbf{f}_{\Lambda_n}$ with increasing accuracy to satisfy the test criterion (3.2):

1. $J > \max|\lambda| + 2$ $\mathbf{f}^{(J)}_{\Lambda_n}$
2. $\mathbf{f}^{(J-1)}_{\Lambda_n}$ $J - 1$
3. $\mathbf{h}^{(J)}_{\Lambda_n} := \mathbf{f}^{(J)}_{\Lambda_n} - \mathbf{f}^{(J-1)}_{\Lambda_n}$ $\mathbf{f}^{(J+1)}_{\Lambda_n} := \mathbf{f}^{(J)}_{\Lambda_n}$
4. $h^{(J)}_\lambda$ $\mathbf{h}^{(J)}_{\Lambda_n}$
   - $|h^{(J)}_\lambda| > \mathrm{tol}_{int} \delta_{|\lambda|} 2^{-d/2|\lambda|} \|\mathbf{f}^{(J)}_{\Lambda_n}\|_2$, $f^{(J+1)}_\lambda$ $\mathbf{f}^{(J+1)}_{\Lambda_n}$ $J + 1$
5. $h^{(J)}_\lambda$ $\lambda \in \Lambda_n$ $J = J + 1$ 3 $\mathbf{f}_{\Lambda_n} := \mathbf{f}^{(J+1)}_{\Lambda_n}$

We set $\delta_{|\lambda|} = 1/|\lambda|$ in (3.2), and the level J is chosen in such a way that we are able to detect all singularities in f.

3.3. The choice $J > \max|\lambda| + 2$ in the previous algorithm is not optimal. Nevertheless, in many applications in which the maximal level needed by the approximation process is not too large, the evaluation of the right-hand side on a fine fixed grid leads to a faster approximation of the integrals than an adaptive method like the one presented for the entries of the stiffness matrix. An algorithm similar to the one proposed for the stiffness matrix can be applied for the right-hand side too.

Computing the entries of $\mathbb{A}_{\Lambda_n \times \Lambda_n}$. In this case, we apply a composite Gauss quadrature formula to compute $a_{\lambda,\mu}$. To satisfy (3.1) for all entries we can use a modification of the above-mentioned adaptive scheme applying a sequence of Gauss quadrature formulas on each dyadic cube in the intersection of the supports of test and trial functions [5] to satisfy the test criterion (3.1), where $\delta_{||\lambda|-|\mu||} = 1/(||\lambda|-|\mu||^2)$. In this approach only dyadic cubes of level $\max\{|\lambda|, |\mu|\}$ are involved in the computation of the integrals.

For $n > 0$, the entries $a_{\lambda,\mu}$ and f_λ with $\lambda \in \delta\Lambda_{n-1} \subset \tilde{\Lambda}_{n-1}$ and $\mu \in \overline{\Lambda}_{n-1}$ have been computed and stored in the previous iteration during the computation of the residuum $\tilde{\mathbf{r}}_{\tilde{\Lambda}_{n-1}}$. Thus we compute only the remaining entries.

FUNCTION $[\mathbb{A}_{\Lambda_n\times\Lambda_n}, \mathbf{f}_{\Lambda_n}] = \text{Assemble}(\overline{\Lambda}_{n-1}, \delta\Lambda_{n-1})$.

(i) [illegible] $a_{\lambda,\mu}$ [illegible] $(\lambda,\mu) \in \{\{\overline{\Lambda}_{n-1}\times\delta\Lambda_{n-1}\}\cup\{\delta\Lambda_{n-1}\times\delta\Lambda_{n-1}\}\}$

(ii) [illegible] $\mathbb{A}_{\Lambda_n\times\Lambda_n}$ [illegible] $\mathbf{f}_{\Lambda_n}$ [illegible]

3.5. Iterative improvement of $\mathbf{x}_{\Lambda_n}$. For each outer iteration we successively improve the current approximation of the coefficient vector $\mathbf{x}^{\text{init}}_{\Lambda_n}$ by approximately solving the linear algebraic system $\mathbb{A}_{\Lambda_n\times\Lambda_n}\mathbf{x}_{\Lambda_n} = \mathbf{f}_{\Lambda_n}$ using the chosen iterative method. We stop iterations when the following relative stopping criterion is matched:

$$\|\mathbf{r}_{\Lambda_n}\|_2^2 \le \text{tol}^2_{grad}\|\mathbf{f}_{\Lambda_n} - \mathbb{A}_{\Lambda_n\times\Lambda_n}\mathbf{x}^{\text{init}}_{\Lambda_n}\|_2^2, \tag{3.3}$$

where $\mathbf{r}_{\Lambda_n} = \mathbf{f}_{\Lambda_n} - \mathbb{A}_{\Lambda_n\times\Lambda_n}\mathbf{x}_{\Lambda_n}$ is the residual vector corresponding to the active set Λ_n. The parameter tol_{grad} is a given relative stopping tolerance. The idea of the iterative improvement is to solve in each outer iteration the corresponding algebraic system to a moderate accuracy, in order to avoid oversolving the linear system when the discretization error is still large.

FUNCTION $[\mathbf{x}_{\Lambda_n}, \mathbf{r}_{\Lambda_n}] = \text{GradMethod}(\mathbb{A}_{\Lambda_n\times\Lambda_n}, \mathbf{f}_{\Lambda_n}, \mathbf{x}^{\text{init}}_{\Lambda_n}, tol_{grad})$. [illegible] symmetric positive definite matrices [illegible] GradMethod [illegible] steepest descent CG [illegible] non-symmetric matrices [illegible] GradMethod [illegible] BiCG GMRES [illegible]

3.6. Removing negligible basis functions. Some basis functions ψ_λ, $\lambda \in \Lambda_n$, can have negligible energy contributions $|x_\lambda|^2$ to the solution, i.e., their coefficients are less significant so we remove them from Λ_n to get $\overline{\Lambda}_n$, in order to keep Λ_{n+1} as small as possible. The number of basis functions, which we may remove, is related to the total energy $\|\mathbf{x}_{\Lambda_n}|_{\delta\Lambda_{n-1}}\|_2^2$ of newly added functions ψ_μ, $\mu \in \delta\Lambda_{n-1}$. We call this process [illegible], and it can be realized by sorting absolute values of the entries of $\mathbf{x}_{\Lambda_n}$ in decreasing order, successively summing (starting from the bottom of the sorted vector) squares of the smallest ones, while this sum is less than $\text{tol}^2_{off}\|\mathbf{x}_{\Lambda_n}|_{\delta\Lambda_{n-1}}\|_2^2$. We remove these entries from $\mathbf{x}_{\Lambda_n}$ and their indices from Λ_n [illegible]. The particular realization is similar to the routine Activate and appears as follows.

FUNCTION $[\overline{\mathbf{x}}_{\overline{\Lambda}_n}, \overline{\Lambda}_n] = \text{Threshold}(\mathbf{x}_{\Lambda_n}, \Lambda_n, \delta\Lambda_{n-1}, \text{tol}_{off})$.

(i) [illegible] $N := \text{card}\,\Lambda_n$ [illegible] $\mathbf{x}_{\Lambda_n}$ [illegible] $\lambda^* := \lambda^*(\mathbf{x}_{\Lambda_n}) = (\lambda_1^*, \lambda_2^*, \dots, \lambda_N^*)$ [illegible] $\mathbf{x}^*_{\Lambda_n} = (|x_{\lambda_1^*}|, |x_{\lambda_2^*}|, \dots, |x_{\lambda_N^*}|)$ [illegible] $\|\mathbf{x}_{\Lambda_n}|_{\delta\Lambda_{n-1}}\|_2^2$

(ii) [illegible] $k = 1, 2, \dots,$ [illegible] $\sum_{l=1}^{k}|x^*_{\lambda_{N-l+1}}|^2$ [illegible] k [illegible] $\text{tol}^2_{off}\|\mathbf{x}_{\Lambda_n}|_{\delta\Lambda_{n-1}}\|_2^2$ [illegible] k [illegible] $\overline{\Lambda}_n := \{\lambda_l^*;\ l = 1, \dots, N-k\} \cup \{$ [illegible] $\lambda_m^*,\ m > N-k,$ [illegible] $\}$ [illegible] $\overline{\mathbf{x}}_{\overline{\Lambda}_n} := \mathbf{x}_{\Lambda_n}|_{\overline{\Lambda}_n}$

3.7. Construction of the activable set $\tilde{\Lambda}_n$. We start with the empty set $\tilde{\Lambda}_n$. For all $\lambda \in \overline{\Lambda}_n$, the indices μ of all functions ψ_μ constructed at the level $|\mu| = |\lambda| + 1$ whose supports have nonempty intersections with the support of ψ_λ and $\mu \notin \overline{\Lambda}_n$

are added to $\tilde{\Lambda}_n$. The number of activable functions ψ_μ, $\mu \in \tilde{\Lambda}_n$, can be many times greater than the number of active functions corresponding to $\overline{\Lambda}_n$. Therefore it is convenient to introduce a spatial truncation rule to keep $\tilde{\Lambda}_n$ small. We use contracted support instead of considering intersections with the whole support of ψ_λ, using a contraction factor c. Typical choices are $c = 1/4$ or $c = 1/8$. Another possibility is to select only adjacent wavelets with respect to the positions of their central dyadic nodes, i.e., 2 functions in one dimension, 8 in two dimensions and 26 in three dimensions.

FUNCTION $\tilde{\Lambda}_n$ = ConstructActivableSet($\overline{\Lambda}_n$).

(i) [illegible] $\tilde{\Lambda}_n := \emptyset$

(ii) [illegible] $\lambda \in \overline{\Lambda}_n$ [illegible] $\tilde{\Lambda}_n$ [illegible] μ [illegible] ψ_μ [illegible]
$|\mu| = |\lambda| + 1$ [illegible] $\mu \notin \overline{\Lambda}_n$ [illegible]

Var. 1. $\operatorname{supp}\psi_\mu$ [illegible] $\operatorname{supp}\psi_\lambda$ [illegible] $c \in (0, 1]$ [illegible]

Var. 2. ψ_μ [illegible]

[illegible]

3.8. Assembling the residual vector $\tilde{\mathbf{r}}_{\tilde{\Lambda}_n}$. For given $\tilde{\Lambda}_n$ we assemble the residual vector $\tilde{\mathbf{r}}_{\tilde{\Lambda}_n}$ using the activable functions ψ_λ, $\lambda \in \tilde{\Lambda}_n$. The entries $\tilde{r}_\lambda$ are computed by

$$\tilde{r}_\lambda = \tilde{f}_\lambda - d_\lambda^{-1}\langle \mathcal{A}\overline{u}_{\overline{\Lambda}_n}, \psi_\lambda\rangle \quad \forall \lambda \in \tilde{\Lambda}_n, \tag{3.4}$$

where $\tilde{f}_\lambda = d_\lambda^{-1}\langle f, \psi_\lambda\rangle$ and $\overline{u}_{\overline{\Lambda}_n}$ is given by

$$\overline{u}_{\overline{\Lambda}_n} = \sum_{\mu \in \overline{\Lambda}_n} d_\mu^{-1}\overline{x}_\mu \psi_\mu = \overline{\mathbf{x}}_{\overline{\Lambda}_n}^T \mathbb{D}_{\overline{\Lambda}_n}^{-1} \boldsymbol{\psi}_{\overline{\Lambda}_n}. \tag{3.5}$$

We substitute the expansion (3.5) of $\overline{u}_{\overline{\Lambda}_n}$ into (3.4) and we obtain

$$\tilde{r}_\lambda = \tilde{f}_\lambda - \sum_{\forall \mu \in \overline{\Lambda}_n} d_\lambda^{-1} d_\mu^{-1} \overline{x}_\mu \langle \mathcal{A}\psi_\mu, \psi_\lambda\rangle \quad \forall \lambda \in \tilde{\Lambda}_n,$$

or in matrix form, $\tilde{\mathbf{r}}_{\tilde{\Lambda}_n} = \tilde{\mathbf{f}}_{\tilde{\Lambda}_n} - \mathbb{A}_{\tilde{\Lambda}_n \times \overline{\Lambda}_n} \overline{\mathbf{x}}_{\overline{\Lambda}_n}$.

To compute the entries of $\tilde{\mathbf{f}}_{\tilde{\Lambda}_n}$ and $\mathbb{A}_{\tilde{\Lambda}_n \times \overline{\Lambda}_n}$ we use the same quadrature formulas as for computing the entries of $\mathbf{f}_{\Lambda_n}$ and $\mathbb{A}_{\Lambda_n \times \Lambda_n}$. The resulting routine appears as follows.

FUNCTION $[\tilde{\mathbf{r}}_{\tilde{\Lambda}_n}, \tilde{\mathbf{f}}_{\tilde{\Lambda}_n}]$ = Residual($\tilde{\Lambda}_n, \overline{\Lambda}_n, \overline{\mathbf{x}}_{\overline{\Lambda}_n}$).

(i) [illegible] $\tilde{f}_\lambda$ $a_{\lambda,\mu}$ [illegible] $\tilde{r}_\lambda$ [illegible] $\lambda \in \tilde{\Lambda}_n$ [illegible] $\mu \in \overline{\Lambda}_n$ [illegible]
[illegible]

3.9. Stopping criterion. A natural stopping criterion of the outer loop is based on the achievement of the sufficient (relative) reduction of the total residuum $\|\mathbf{f}_\mathcal{J} - \mathbb{A}_{\mathcal{J}\times\Lambda_n}\mathbf{x}_{\Lambda_n}\|_{l_2}$. If we admit that the sets $\tilde{\Lambda}_n$ catch the bulk of the residuum in each iteration, $\|\mathbf{f}_\mathcal{J} - \mathbb{A}_{\mathcal{J}\times\Lambda_n}\mathbf{x}_{\Lambda_n}\|_{l_2}$ is well approximated by $\sqrt{\|\mathbf{r}_{\Lambda_n}\|_2^2 + \|\tilde{\mathbf{r}}_{\tilde{\Lambda}_n}\|_2^2}$ and $\|\mathbf{f}_\mathcal{J}\|_{l_2}$ by $\sqrt{\|\mathbf{f}_{\Lambda_n}\|_2^2 + \|\tilde{\mathbf{f}}_{\tilde{\Lambda}_n}\|_2^2}$ for n sufficiently large. Our final stopping criterion is as follows:

if $(\|\mathbf{r}_{\Lambda_n}\|_2^2 + \|\tilde{\mathbf{r}}_{\tilde{\Lambda}_n}\|_2^2 \le \varepsilon^2 (\|\mathbf{f}_{\Lambda_n}\|_2^2 + \|\tilde{\mathbf{f}}_{\tilde{\Lambda}_n}\|_2^2))$,
then $\mathbf{x}(\varepsilon) := \overline{\mathbf{x}}_{\overline{\Lambda}_n}$, $\Lambda(\varepsilon) := \overline{\Lambda}_n$ and stop the outer iterations.

3.10. ADWEM solver. In this subsection, we summarize the above-mentioned ingredients of the algorithm by the calling sequence of the functions corresponding to the previous different steps.

ADWEM SOLVER.
$\Lambda_{-1} = \overline{\Lambda}_{-1} = \emptyset$ $\overline{\mathbf{x}}_{\overline{\Lambda}_{-1}} = 0$ $\delta\Lambda_{-1} = \{indices\ of\ all\ scaling\ functions\}$
For $n = 0, 1, \dots$ **do**
If $(n = 0)$ then $\Lambda_0 = \delta\Lambda_{-1}$ else $[\Lambda_n, \delta\Lambda_{n-1}] = \text{Activate}(\overline{\Lambda}_{n-1}, \tilde{\Lambda}_{n-1}, \tilde{\mathbf{r}}_{\tilde{\Lambda}_{n-1}}, \text{tol}_{on})$.
$\mathbf{x}^{\text{init}}_{\Lambda_n} = \text{Extend}(\overline{\mathbf{x}}_{\overline{\Lambda}_{n-1}}, \overline{\Lambda}_{n-1}, \delta\Lambda_{n-1})$.
$[\mathbb{A}_{\Lambda_n \times \Lambda_n}, \mathbf{f}_{\Lambda_n}] = \text{Assemble}(\overline{\Lambda}_{n-1}, \delta\Lambda_{n-1})$.
$[\mathbf{x}_{\Lambda_n}, \mathbf{r}_{\Lambda_n}] = \text{GradMethod}(\mathbb{A}_{\Lambda_n \times \Lambda_n}, \mathbf{f}_{\Lambda_n}, \mathbf{x}^{\text{init}}_{\Lambda_n}, \text{tol}_{grad})$ until $(\|\mathbf{r}_{\Lambda_n}\|_2^2 \le \text{tol}^2_{grad} \|\mathbf{f}_{\Lambda_n} - \mathbb{A}_{\Lambda_n \times \Lambda_n} \mathbf{x}^{\text{init}}_{\Lambda_n}\|_2^2)$.
$[\overline{\mathbf{x}}_{\overline{\Lambda}_n}, \overline{\Lambda}_n] = \text{Threshold}(\mathbf{x}_{\Lambda_n}, \Lambda_n, \delta\Lambda_{n-1}, \text{tol}_{off})$.
$\tilde{\Lambda}_n = \text{ConstructActivableSet}(\overline{\Lambda}_n)$.
$[\tilde{\mathbf{r}}_{\tilde{\Lambda}_n}, \tilde{\mathbf{f}}_{\tilde{\Lambda}_n}] = \text{Residual}(\tilde{\Lambda}_n, \overline{\Lambda}_n, \overline{\mathbf{x}}_{\overline{\Lambda}_n})$.
If $\|\mathbf{r}_{\Lambda_n}\|_2^2 + \|\tilde{\mathbf{r}}_{\tilde{\Lambda}_n}\|_2^2 \le \varepsilon^2 (\|\mathbf{f}_{\Lambda_n}\|_2^2 + \|\tilde{\mathbf{f}}_{\tilde{\Lambda}_n}\|_2^2)$ then $\mathbf{x}(\varepsilon) := \overline{\mathbf{x}}_{\overline{\Lambda}_n}$ $\Lambda(\varepsilon) := \overline{\Lambda}_n$
end.

3.4. In the previous description we use a fixed tolerance tol_{grad}. Nevertheless it is possible to use a changing tolerance. We propose the following rule for choosing tol_{grad} at each iteration: given tol_{grad_0}, define

$$\text{tol}_{grad_n} := \min \left\{ \text{tol}_{grad_0}, \mathcal{C} \frac{\|\mathbf{f}_{\Lambda_{n-1}} - \mathbb{A}_{\Lambda_{n-1} \times \Lambda_{n-1}} \mathbf{x}_{\Lambda_{n-1}}\|_2}{\|\mathbf{f}_{\Lambda_n} - \mathbb{A}_{\Lambda_n \times \Lambda_n} \mathbf{x}^{\text{init}}_{\Lambda_n}\|_2} \right\}, \qquad \mathcal{C} \in (0, 1).$$

For this choice, the residuum norm $\|\mathbf{r}_{\Lambda_n}\|_2$ decreases minimally with the rate $\mathcal{C}$ with increasing outer iteration n.

4. Numerical examples. In this section, we illustrate the behavior of the introduced adaptive algorithm. In Example 1, the computational domain Ω is a curved quadrilateral and the forcing function f induces in the solution two Gaussian peaks of different heights, slopes, and positions (see Figures 4.1 and 4.2). We expect the adaptive algorithm to feel at first the higher peak and, after reaching a good approximation of this peak, the algorithm should reveal the smaller one. For this case, we illustrate sensitivity to the ADWEM parameters tol_{off}, tol_{on}, and the support contraction c. In Example 2, we take the same geometry and split it into two subdomains with different mappings. The particular geometries are depicted in Figures 4.4 and 4.10, respectively. With these examples we want to investigate the effects of mapping and matching on the adaptive method.

The remaining two examples correspond to an L-shaped domain. These cases are interesting because f is everywhere infinitely differentiable and a singularity of u is generated by the shape of the domain (see Figures 4.14 and 4.15). The last example is aimed at showing the difference in terms of rates of convergence between the linear and the nonlinear approximations, respectively, depending on the Sobolev and the Besov regularities. Further numerical results and comparisons can be found in [6].

In our numerical experiments we use piecewise linear and piecewise quadratic biorthogonal B-splines, with $L = 2$, $\tilde{L} = 2$ and $L = 3$, $\tilde{L} = 5$, respectively. We denote by L and $\tilde{L}$ the exactness order of the primal and dual basis, respectively, i.e., polynomials up to the degree $L - 1$ and $\tilde{L} - 1$ are reproduced exactly [26]. If

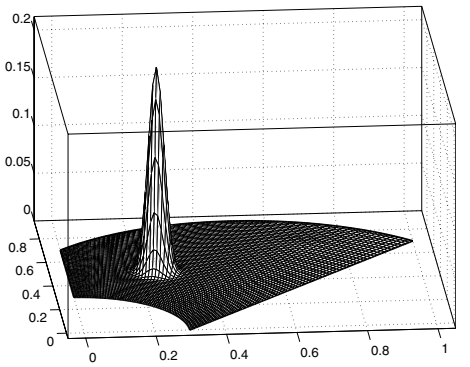


FIG. 4.1. *Example* 1. *Exact solution* u_{ex} *(linear scale).*

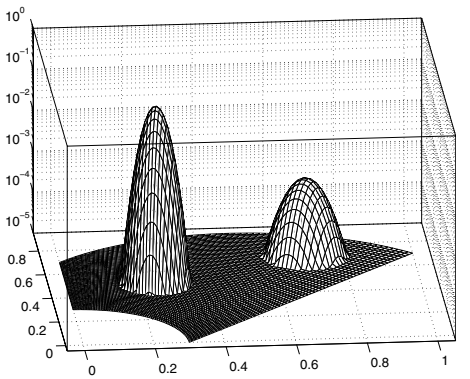


FIG. 4.2. *Example* 1. *Exact solution* u_{ex} *(log scale).*

not specified, the starting scaling level j_0 is equal to 4. The operator we consider is elliptic and self-adjoint; therefore we replace GradMethod by CG. We set the stopping tolerance $\text{tol}_{grad_0} = 1e-4$ and $\mathcal{C} = 0.9$. Other parameters of the adaptive algorithm are the relative activating and thresholding tolerances $\text{tol}_{on} = 0.99$ and $\text{tol}_{off} = 1e-3$ and the relative target accuracy ε of the adaptive process ($\varepsilon = 1e-3$ for Examples 1 and 2 and $\varepsilon = 5e-2$ for Examples 3 and 4). In computing the entries $a_{\lambda,\mu}$, f_λ we use the relative tolerance $\text{tol}_{int} = 1e-6$. We set $c = 0.125$ in the relation that selects the activable functions (see the routine ConstructActivableSet).

1. Let $\Omega \subset \mathbb{R}^2$ be the curved quadrilateral depicted in Figure 4.4. The corresponding mapping $F : \hat{\Omega} \to \Omega$ is chosen in such a way that it maps the reference unit square $\hat{\Omega}$ to the set

$$\Omega = \left\{(x,y) \in \mathbb{R}^2 \, : x > 0, y > x - 0.3, x^2 + y^2 > 0.09, (x-0.5)^2 + (y-0.2)^2 < 0.5\right\}.$$

Let us consider the problem

$$\begin{cases} -\triangle u = f & \text{in } \Omega, \\ u = 0 & \text{on } \partial\Omega \end{cases} \tag{4.1}$$

with $f = -\triangle u_{ex}$ and

$$u_{ex}(x,y) = x(y - x + 0.3)(0.5 - (x-0.5)^2 - (y-0.2)^2)(x^2 + y^2 - 0.09)\ g,$$

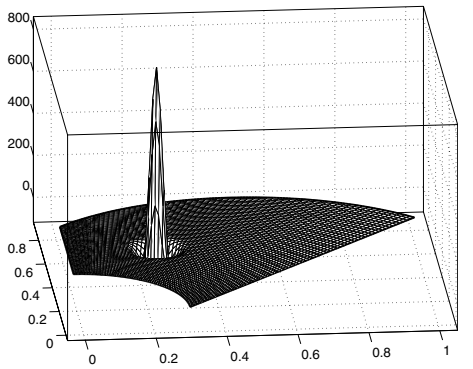


FIG. 4.3. *Example* 1. *Right-hand side f (linear scale).*

TABLE 4.1
Example 1. *ADWEM parameters sensitivity* ($\text{tol}_{off} = 0.001$, $c = 0.125$).

	$\text{tol}_{on} = 0.7$		$\text{tol}_{on} = 0.9$		$\text{tol}_{on} = 0.99$	
Outer iter. n	$\#\Lambda_n$	$\delta_{H^1(\Omega)}$	$\#\Lambda_n$	$\delta_{H^1(\Omega)}$	$\#\Lambda_n$	$\delta_{H^1(\Omega)}$
0	225	6.274e-1	225	6.274e-1	225	6.274e-1
1	226	5.288e-1	230	3.804e-1	241	3.374e-1
2	229	4.015e-1	250	2.170e-1	298	1.722e-1
3	234	3.135e-1	328	1.213e-1	497	8.676e-2
4	245	2.323e-1	567	6.764e-2	1245	4.289e-2
5	268	1.726e-1	1278	3.769e-2	4103	1.953e-2
6	313	1.301e-1	3353	2.024e-2	15161	3.463e-3

TABLE 4.2
Example 1. *ADWEM parameters sensitivity* ($\text{tol}_{off} = 0.01$, $c = 0.125$).

	$\text{tol}_{on} = 0.7$		$\text{tol}_{on} = 0.9$		$\text{tol}_{on} = 0.99$	
Outer iter. n	$\#\Lambda_n$	$\delta_{H^1(\Omega)}$	$\#\Lambda_n$	$\delta_{H^1(\Omega)}$	$\#\Lambda_n$	$\delta_{H^1(\Omega)}$
0	225	6.274e-1	225	6.274e-1	225	6.274e-1
1	226	5.288e-1	230	3.804e-1	241	3.374e-1
2	229	4.015e-1	250	2.170e-1	298	1.722e-1
3	234	3.135e-1	328	1.213e-1	497	8.676e-2
4	245	2.323e-1	567	6.764e-2	1244	4.289e-2
5	268	1.726e-1	1278	3.769e-2	4101	1.954e-2
6	313	1.301e-1	3353	2.024e-2	15152	3.462e-3

where $g = 20e^{-1000r_1^2} + 0.02e^{-300r_2^2}$ with $r_1^2 = (x - 0.25)^2 + (y - 0.5)^2$ and $r_2^2 = (x - 0.7)^2 + (y - 0.7)^2$. The plots of f in the linear scale and u_{ex} in the linear and logarithmic scales are plotted in Figures 4.3, 4.1, and 4.2, respectively.

Sensitivity of ADWEM algorithm with respect to the parameters tol_{off}, tol_{on}, and support contraction c considered are shown in Tables 4.1–4.4. As expected, larger values of tol_{on} imply a faster activation of new basis functions and a parallel reduction of the error. On the other hand, for steady problems the thresholding step is in practice less important, and this can be observed from the smooth dependence of the number of basis functions and of the error on the parameter tol_{off}. The number of active functions $N = \#\Lambda_n$ grows slowly as c increases, since the parameter c mainly affects the number of activable basis functions at each iteration than the number of active basis functions.

TABLE 4.3
Example 1. *ADWEM parameters sensitivity* ($\mathrm{tol}_{off} = 0.1$, $c = 0.125$).

	$\mathrm{tol}_{on} = 0.7$		$\mathrm{tol}_{on} = 0.9$		$\mathrm{tol}_{on} = 0.99$	
Outer iter. n	$\#\Lambda_n$	$\delta_{H^1(\Omega)}$	$\#\Lambda_n$	$\delta_{H^1(\Omega)}$	$\#\Lambda_n$	$\delta_{H^1(\Omega)}$
0	225	6.274e-1	225	6.274e-1	225	6.274e-1
1	226	5.288e-1	230	3.804e-1	237	3.393e-1
2	229	4.015e-1	250	2.170e-1	287	1.736e-1
3	234	3.135e-1	327	1.215e-1	470	8.790e-2
4	245	2.323e-1	559	6.787e-2	1159	4.355e-2
5	268	1.726e-1	1248	3.821e-2	3733	2.006e-2
6	313	1.301e-1	3226	2.074e-2	13599	4.303e-3

TABLE 4.4
Example 1. *ADWEM parameters sensitivity* ($\mathrm{tol}_{off} = 0.001$, $\mathrm{tol}_{on} = 0.99$).

	$c = 0.125$		$c = 0.5$		$c = 1.0$	
Outer iter. n	$\#\Lambda_n$	$\delta_{H^1(\Omega)}$	$\#\Lambda_n$	$\delta_{H^1(\Omega)}$	$\#\Lambda_n$	$\delta_{H^1(\Omega)}$
0	225	6.274e-1	225	6.274e-1	225	6.274e-1
1	241	3.374e-1	241	3.374e-1	241	3.374e-1
2	298	1.722e-1	299	1.721e-1	299	1.721e-1
3	497	8.676e-2	499	8.669e-2	500	8.667e-2
4	1245	4.289e-2	1271	4.272e-2	1282	4.265e-2
5	4103	1.953e-2	4239	1.938e-2	4324	1.930e-2
6	15161	3.463e-3	15756	3.170e-3	16205	2.969e-3

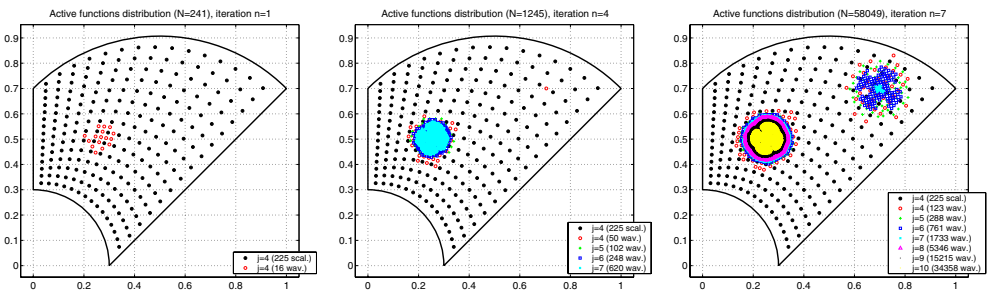


FIG. 4.4. *Example* 1. *Distribution of active functions,* $n = 1, 4, 7$, $N = \#\Lambda_n$.

In Figure 4.4, we successively plot the distributions of active basis functions in outer iterations $n = 1, 4, 7$ for the choice $L = 2$, $\tilde{L} = 2$. We observe that the adaptive algorithm recognizes at first the deficiency corresponding to the higher peak and activates new basis functions near its center. When the adaptive process reaches a good approximation of the higher peak (iteration 4) it reveals another deficiency corresponding to the smaller peak and starts to add new basis functions there as well. The approximate solution u_{Λ_n} and the difference $|u_{\Lambda_n} - u_{ex}|$ are depicted in the logarithmic scale in Figures 4.5 and 4.6, respectively, for $n = 1, 4, 7$. The terms involved in the stopping criterion of the adaptive algorithm are depicted in Figure 4.7.

Finally, we plot the relative errors

$$\delta_{L^2} = \frac{\|u_{\Lambda_n} - u_{ex}\|_{L^2(\Omega)}}{\|u_{ex}\|_{L^2(\Omega)}}, \quad \delta_{H^1} = \frac{\|u_{\Lambda_n} - u_{ex}\|_{H^1(\Omega)}}{\|u_{ex}\|_{H^1(\Omega)}}$$

of the approximate solution u_{Λ_n} in Figure 4.8 with respect to the number of active functions $N = \#\Lambda_n$. The above-mentioned L^2- and H^1-norms are evaluated using

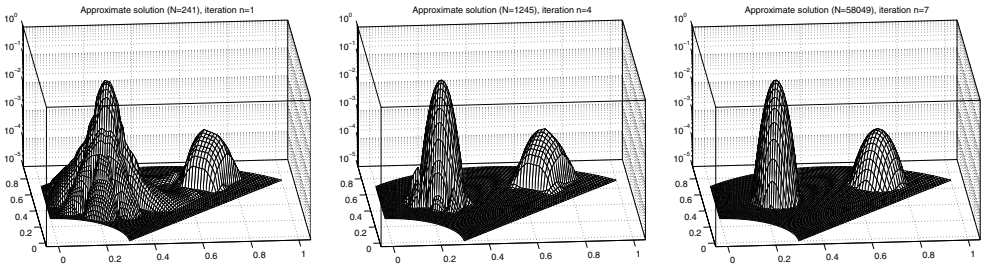


FIG. 4.5. *Example* 1. *Computed solution* u_{Λ_n}, $n = 1, 4, 7$, $N = \#\Lambda_n$.

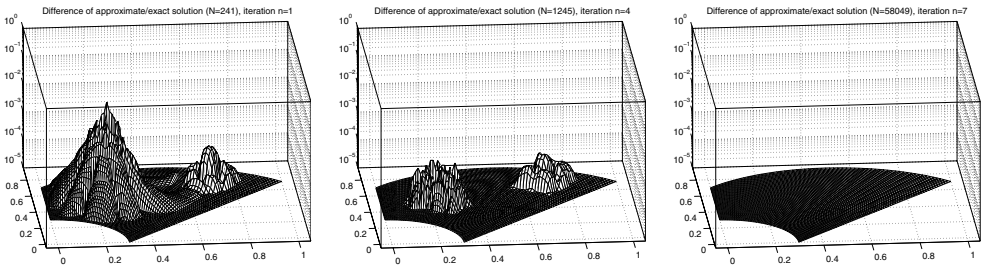


FIG. 4.6. *Example* 1. *Error of the computed solution* $|u_{\Lambda_n} - u_{ex}|$, $n = 1, 4, 7$, $N = \#\Lambda_n$.

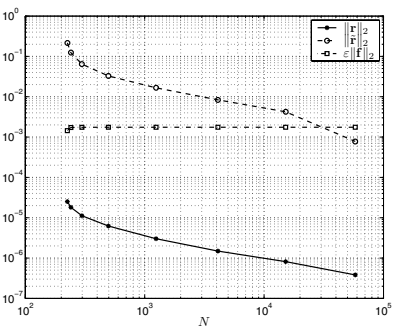


FIG. 4.7. *Example* 1. *Terms in the stopping criterion.*

the Simpson composite formula at the fixed very fine grid of level 13. For comparison purposes we depict the relative errors of the best N-term approximations in $L^2(\Omega)$ and $H^1(\Omega)$ computed using the wavelet coefficients in the final iteration n as the wavelet coefficients of the exact solution. The numerical values of the relative errors δ_{L^2} and δ_{H^1} are listed in Table 4.5. In this table we report also the number of iterations of the CG iterative solver (CG iters.) and the number of active basis functions ($\#\Lambda_n$). We remark that the number of CG iterations is not growing with the number of basis functions and its levels. This is due to the good conditioning of the matrix $\mathbb{A}$ in all the outer iterations. In Figures 4.8 and 4.9 and in Tables 4.5 and 4.6, we report the behavior of the relative errors using linear and quadratic basis functions.

The rates of convergence α of the least squares approximations cN^{α} of the relative errors with respect to $N = \#\Lambda_n$ are given in Table 4.7 for both linear and quadratic basis functions using all iterations reported in Tables 4.5 and 4.6. The

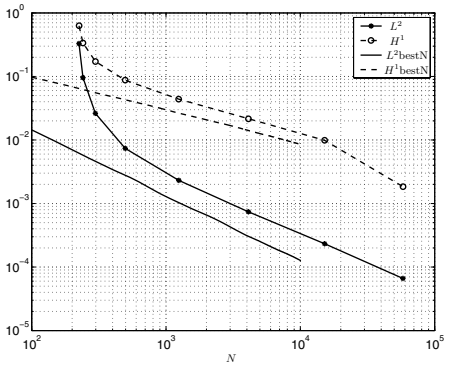


FIG. 4.8. *Example* 1. *Relative errors* $L = 2$, $\tilde{L} = 2$.

TABLE 4.5

Example 1. *Case* $L = 2$, $\tilde{L} = 2$. *Behavior of the number of inner iterations, the number of active basis functions, and the relative errors in the outer iteration.*

Outer iter. n	CG iters.	$\#\Lambda_n$	$\delta_{L^2(\Omega)}$	$\delta_{H^1(\Omega)}$
0	25	225	3.292e-1	6.274e-1
1	17	241	9.576e-2	3.376e-1
2	15	298	2.626e-2	1.725e-1
3	13	497	7.374e-3	8.725e-2
4	15	1245	2.314e-3	4.388e-2
5	13	4103	7.411e-4	2.161e-2
6	12	15161	2.324e-4	9.872e-3
7	14	58049	6.603e-5	1.846e-3

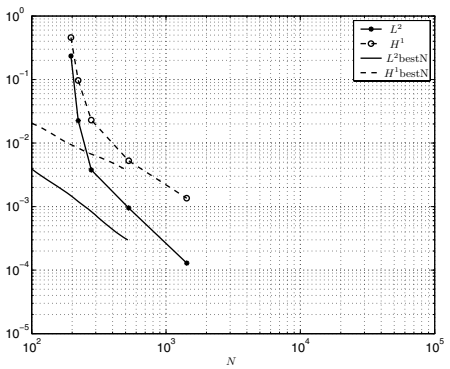


FIG. 4.9. *Example* 1. *Relative errors* $L = 3$, $\tilde{L} = 5$.

theoretical rates of convergence are determined by the polynomial degree of the used basis functions [23].

2. In this case, we split the domain Ω of the previous example into two parts, Ω_1 (lower) and Ω_2 (upper), as shown in Figure 4.10. The mapping consists of two components which map the reference unit square $\hat{\Omega}$ to Ω_1 and Ω_2, respectively. We remark that the splitting intersects one of the Gaussian peaks.

In Figure 4.10, we successively plot the distributions of active basis functions in outer iterations $n = 1, 4, 7$ ($L = 2$, $\tilde{L} = 2$). The behavior of the adaptive algorithm

TABLE 4.6
Example 1. *Case* $L = 3$, $\tilde{L} = 5$. *Behavior of the number of inner iterations, the number of active basis functions, and the relative errors in the outer iteration.*

Outer iter. n	CG iters.	$\#\Lambda_n$	$\delta_{L^2(\Omega)}$	$\delta_{H^1(\Omega)}$
0	33	196	2.343e-1	4.574e-1
1	15	222	2.251e-2	9.697e-2
2	18	278	3.773e-3	2.310e-2
3	21	527	9.514e-4	5.268e-3
4	31	1427	1.288e-4	1.353e-3

TABLE 4.7
Example 1. *Comparison of the rates of convergence.*

	Theoretical uniform (L^2)	Computed adaptive (L^2)	Theoretical uniform (H^1)	Computed adaptive (H^1)
$L = 2$, $\tilde{L} = 2$	-1	-1.346	$-1/2$	-0.902
$L = 3$, $\tilde{L} = 5$	$-3/2$	-2.059	-1	-1.702

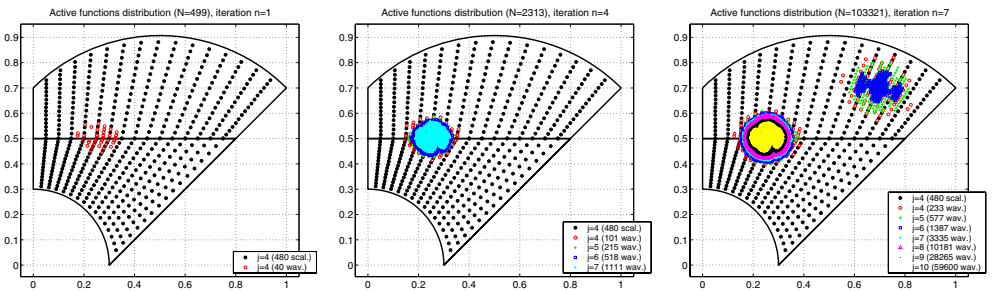


FIG. 4.10. *Example* 2. *Distribution of active functions,* $n = 1, 4, 7$, $N = \#\Lambda_n$.

is similar to the previous case, but we have approximately twice as many candidates for activation. This results from the splitting of Ω into two subdomains Ω_1 and Ω_2 and the use of the same starting scaling level $j_0 = 4$ as in the previous examples in both subdomains. Therefore the algorithm activates more basis functions in the equivalent outer iterations for the same fixed activating and thresholding tolerances than in Example 1 (compare Tables 4.5 and 4.8). The terms involved in the stopping criterion of the adaptive algorithm are depicted in Figure 4.11. Finally, we plot the relative errors $\delta_{L^2(\Omega)}$ and $\delta_{H^1(\Omega)}$ of the approximate solution u_{Λ_n} in Figures 4.12 and 4.13 for $L = 2$, $\tilde{L} = 2$ and $L = 3$, $\tilde{L} = 5$, respectively, together with the relative errors of the best N-term approximations in $L^2(\Omega)$ and $H^1(\Omega)$. The numerical values are listed for $L = 2$, $\tilde{L} = 2$ in Table 4.8. The rates of convergence of the least squares approximations of the errors with respect to N are given in Table 4.9 for both linear and quadratic basis functions.

We remark that the matching induces a faster increase of the number of basis functions caused by the constraint of a minimum construction level and to the larger supports of matched basis functions. These facts do not affect too much the rates of convergence of the ADWEM algorithm because it causes just a shift of the curves (Figures 4.8, 4.9 and 4.12, 4.13).

3 (see [2]). Let $\Omega \subset \mathbb{R}^2$ be the L-shaped domain depicted in Figure 4.16 and let us consider again the problem (4.1) in Ω. Introducing polar coordinates (r, φ),

TABLE 4.8
Example 2. *Case* $L = 2$, $\tilde{L} = 2$. *Behavior of the number of inner iterations, the number of active basis functions, and the relative errors in the outer iteration.*

Outer iter. n	CG iters.	$\#\Lambda_n$	$\delta_{L^2(\Omega)}$	$\delta_{H^1(\Omega)}$
0	53	465	1.834e-1	4.262e-1
1	37	499	9.958e-2	3.146e-1
2	30	607	3.565e-2	1.828e-1
3	37	968	8.829e-3	8.848e-2
4	38	2313	2.341e-3	4.334e-2
5	28	7385	7.523e-4	2.090e-2
6	39	27060	2.229e-4	9.354e-3
7	28	103321	6.594e-5	1.790e-3

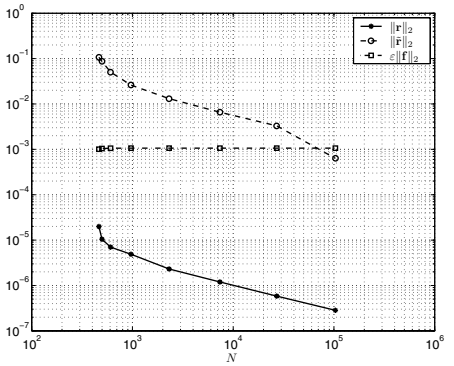


FIG. 4.11. *Example* 2. *Terms in the stopping criterion.*

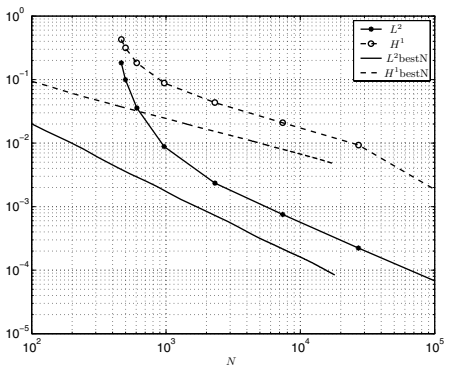


FIG. 4.12. *Example* 2. *Relative errors* $L = 2$, $\tilde{L} = 2$.

the right-hand side f is defined in such a way that the corresponding exact solution is [2] $u_{ex}(r,\varphi) = h(r)r^{2/3}\sin\left(\frac{2}{3}(\varphi + \frac{\pi}{2})\right)$, where $h \in C^\infty(\Omega)$ is the following truncation function:

$$h(r) = \frac{w(3/4 - r)}{w(r - 1/2) + w(3/4 - r)}, \quad w(r) = \begin{cases} e^{-1/r^2} & \text{if } r > 0, \\ 0 & \text{else.} \end{cases}$$

The plots of the right-hand side f and the exact solution u_{ex} are depicted in Figures 4.14 and 4.15, respectively.

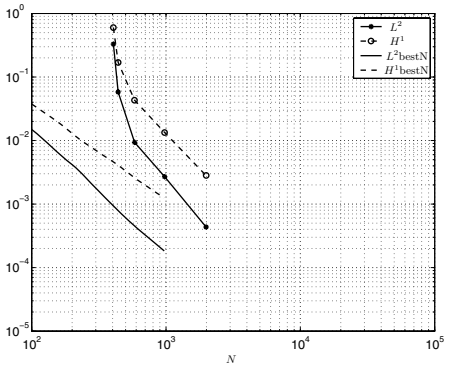


FIG. 4.13. *Example* 2. *Relative errors* $L = 3$, $\tilde{L} = 5$.

TABLE 4.9
Example 2. *Comparison of the rates of convergence.*

	Theoretical uniform (L^2)	Computed adaptive (L^2)	Theoretical uniform (H^1)	Computed adaptive (H^1)
$L = 2$, $\tilde{L} = 2$	-1	-1.374	$-1/2$	-0.911
$L = 3$, $\tilde{L} = 5$	$-3/2$	-2.506	-1	-2.226

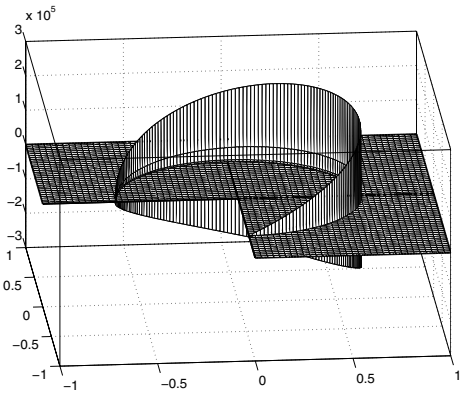


FIG. 4.14. *Example* 3. f *(linear scale).*

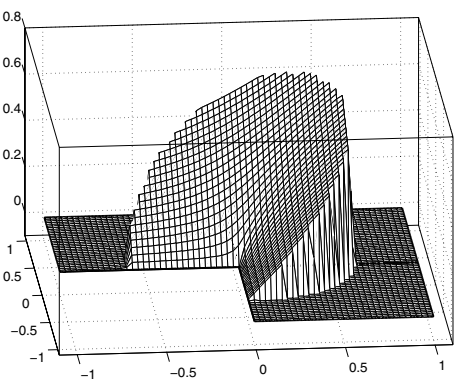


FIG. 4.15. *Example* 3. u_{ex} *(linear scale).*

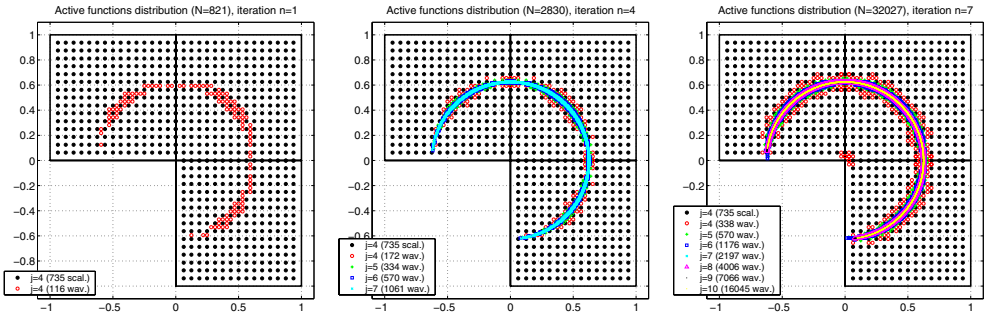


FIG. 4.16. *Example* 3. *Distribution of active functions,* $n = 1, 4, 7$, $N = \#\Lambda_n$.

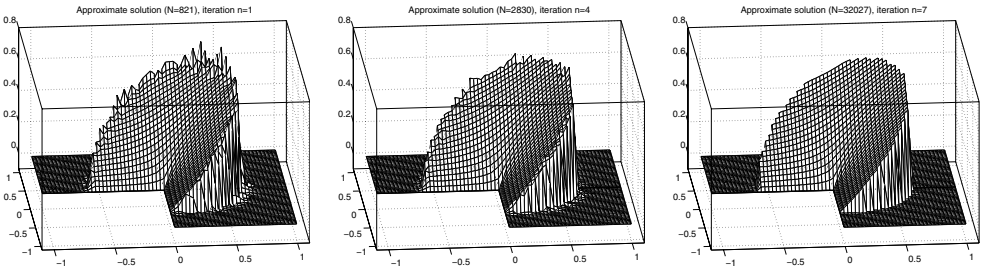


FIG. 4.17. *Example* 3. *Computed solution* u_{Λ_n}, $n = 1, 4, 7$, $N = \#\Lambda_n$.

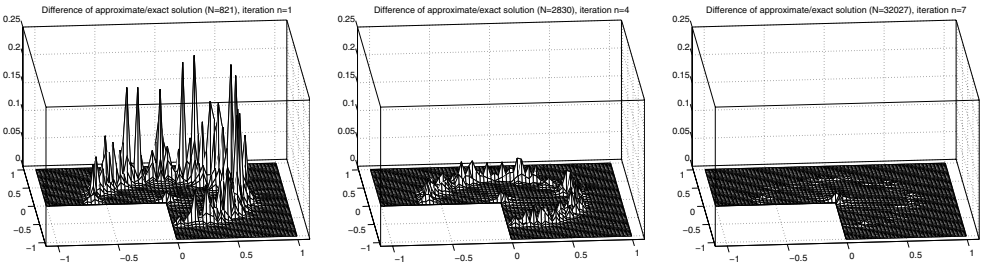


FIG. 4.18. *Example* 3. *Error of the computed solution* $|u_{\Lambda_n} - u_{ex}|$, $n = 1, 4, 7$, $N = \#\Lambda_n$.

In Figure 4.16, we plot successively the distributions of active basis functions in outer iterations $n = 1, 4, 7$ ($L = 2$, $\tilde{L} = 2$). At first the algorithm recognizes the deficiency corresponding to the right-hand side f and activates new basis functions in the region where f displays strong gradients (see Figure 4.14). In iteration 6, the adaptive process reaches a good approximation of the behavior of f, and it reveals another deficiency around the reentrant corner of the L-shaped domain and starts to add new basis functions also there. The approximate solution u_{Λ_n} and the difference $|u_{\Lambda_n} - u_{ex}|$ are depicted in the linear scale in Figures 4.17 and 4.18, respectively, for $n = 1, 4, 7$. In Figures 4.19, 4.20, and 4.21 we plot the behavior of the relative errors corresponding to linear wavelets with $j_0 = 3$, $j_0 = 4$ and quadratic wavelets with $j_0 = 4$, together with the relative errors of the best N-term approximations in $L^2(\Omega)$ and $H^1(\Omega)$. The numerical values of these relative errors for linear basis functions are listed in Table 4.10 ($j_0 = 4$).

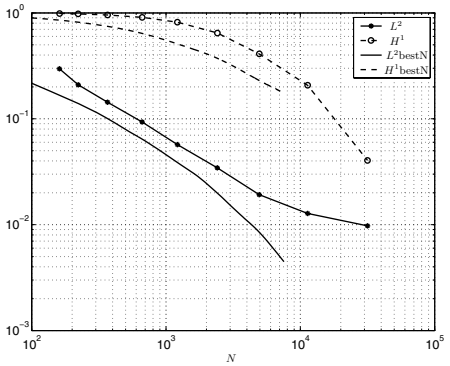


FIG. 4.19. *Example* 3. *Relative errors* $L = 2$, $\tilde{L} = 2$, $j_0 = 3$.

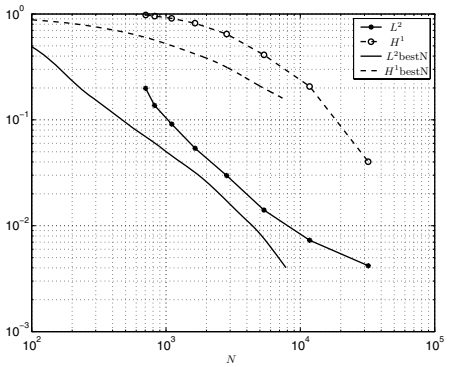


FIG. 4.20. *Example* 3. *Relative errors* $L = 2$, $\tilde{L} = 2$, $j_0 = 4$.

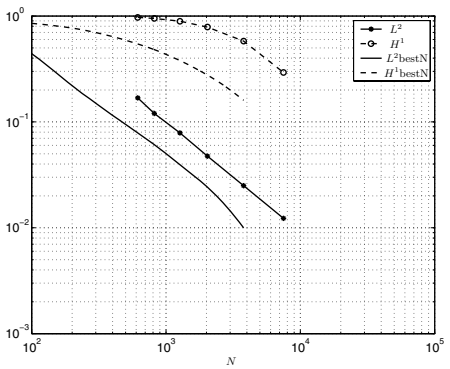


FIG. 4.21. *Example* 3. *Relative errors* $L = 3$, $\tilde{L} = 5$.

The least squares approximations cN^{α} of the errors in the L^2- and H^1-norms with respect to N are given in Table 4.11. In this case, $f \in C^{\infty}(\Omega)$ and $u_{ex} \in H^s(\Omega)$ only for $s < 5/3$ due to its singularity around the reentrant corner [25]. Moreover, u_{ex} belongs to the Besov spaces $B^s_{\tau,\tau}(\Omega)$ for all $s > 0$ and $1/\tau = s/2 + 1/2$ [15]. The upper bound in the Sobolev regularity implies an upper bound in the theoretical rate of convergence for uniform refinement, whereas for the unbounded Besov regularity

TABLE 4.10
Example 3. *Case* $L = 2$, $\tilde{L} = 2$. *Behavior of the number of inner iterations, the number of active basis functions, and the relative errors in the outer iteration.*

Outer iter. n	CG iters.	$\#\Lambda_n$	$\delta_{L^2(\Omega)}$	$\delta_{H^1(\Omega)}$
0	32	705	1.987e-1	9.777e-1
1	25	821	1.369e-1	9.554e-1
2	26	1103	9.119e-2	9.081e-1
3	21	1647	5.390e-2	8.176e-1
4	19	2830	2.976e-2	6.467e-1
5	25	5358	1.405e-2	4.099e-1
6	20	11731	7.284e-3	2.071e-1
7	21	32027	4.191e-3	4.030e-2

TABLE 4.11
Example 3. $L = 2$, $\tilde{L} = 2$. *Comparison of asymptotic constants in the error decay.*

	Theoretical uniform	Computed uniform	Theoretical adaptive	Computed adaptive
L^2	$c_0^u N^{-5/6+\epsilon}$	$3.650N^{-0.408}$	$c_0^a N^{-1}$	$116.734N^{-1.018}$
H^1	$c_1^u N^{-1/3+\epsilon}$	$10.706N^{-0.277}$	$c_1^a N^{-1/2}$	$215.285N^{-0.773}$

TABLE 4.12
Example 4. $L = 2$, $\tilde{L} = 2$. *Comparison of asymptotic constants in the error decay.*

	Theoretical uniform	Computed uniform	Theoretical adaptive	Computed adaptive (1)	Computed adaptive (2)
L^2	$c_0^u N^{-5/6+\epsilon}$	$2.227N^{-0.783}$	$c_0^a N^{-1}$	$17.564N^{-1.131}$	$6.339N^{-0.981}$
H^1	$c_1^u N^{-1/3+\epsilon}$	$2.580N^{-0.456}$	$c_1^a N^{-1/2}$	$6.763N^{-0.636}$	$5.023N^{-0.588}$

we expect a higher theoretical asymptotic rate of convergence for the adaptive case bounded by the order of the polynomial approximation of the basis functions. The theoretically derived rates of convergence for a uniform refinement are $-5/6+\epsilon$ in L^2 and $-1/3+\epsilon$ in H^1 with $\epsilon > 0$.

A similar problem with a L-shape domain with curved boundaries was considered in [6], and results similar to the ones presented above were obtained, meaning that for this kind of problem a nonaffine mapping in the subdomains does not affect the behavior of the method.

EXAMPLE 4. Let us consider now the same problem as in Example 3, but with the truncation function h defined by

$$h(r) = \frac{w(3/4 - r)}{w(r - 1/4) + w(3/4 - r)}, \quad w(r) = \begin{cases} r^2 & \text{if } r > 0, \\ 0 & \text{else.} \end{cases}$$

In contrast to the solution of Example 3, this solution does not present any sharp structure. Thus both adaptive and uniform refinement methods can reach the asymptotic rates of convergence with relatively few degrees of freedom. For this problem we consider two adaptive solutions with the following refinement parameters: $\text{tol}_{on} = 0.9$, $\text{tol}_{off} = 0.01$ (case 1) and $\text{tol}_{on} = 0.99$, $\text{tol}_{off} = 0.001$ (case 2), respectively, and a uniform refinement with linear basis functions.

In Table 4.12, we report the least squares approximations cN^{α} of the errors in L^2- and H^1-norms. The numerical computations give results very close to the expected

theoretical rates of convergence, and in this case the constants of the adaptive case, proportional to the Besov norm of the exact solution, are larger than the constant of the uniform case, proportional to the Sobolev norm of the exact solution.

5. Conclusion. The origin of wavelets (signal processing, harmonic analysis) is far from the numerical solution of boundary value problems in general bounded domains. Yet, the adaptive wavelets algorithm that we have investigated shows that wavelet expansions can be employed in this task as well, with encouraging results. The proposed algorithm shows good behavior with respect to efficiency and reliability in all considered examples, and also when the discretization of the problem requires mapping and splitting into subdomains. To our knowledge this is the first time that these aspects of an adaptive wavelet Galerkin method have been thoroughly investigated. In our tests, we find that the asymptotic behavior of the error is parallel to that of the best N-term approximation in H^1, although the number of basis functions involved in the solution process depends also on the mapping and the splitting. The comparison of results obtained in Examples 1 and 2 clearly suggests keeping the number of subdomains as small as possible.

The efficiency of a wavelet discretization method with respect to other methods (finite elements, spectral methods) is strongly dependent on the specific data structures and algorithms used to represent the discrete operator equations (matrix against matrix-free, approximation of integrals, etc.). By comparing the number of unknowns needed by some well-assessed adaptive FEMs (mentioned in the appendix) and by our ADWEM method to reach the same H^1-relative error on the proposed examples, we find that our method requires significantly fewer basis functions, denoting a good capability of adaptive wavelets methods to catch and resolve very sharp structures in the solution and singularities. However, by comparing the CPU-time, the FEMs are faster, the assembling of the linear system in our method being still quite slow. The gap in CPU-time between the two methods can be strongly reduced with some suitable choices of the data structures, and many improvements are still possible. We remark that wavelet discretization methods offer other advantages; for example, for systems of equations they allow easy choices of different basis functions for different unknowns without mesh constraint. These advantages can be relevant in some applications.

Appendix. Comparison with adaptive FEMs. In this appendix, we compare some results achieved by our algorithm ($L = \tilde{L} = 2$, corresponding to linear wavelets) with the results produced using linear finite elements by the well-assessed adaptive finite element code ALBERTA (Adaptive multiLevel finite element toolbox using Bisectioning refinement and Error control by Residual Techniques for scientific Applications) [30, 1]. For adaptive FEMs, as for the adaptive wavelets methods, important results on the convergence and optimality are known [24, 29, 9, 31].

We point out that the comparisons reported here are aimed at drawing indications about viability of the proposed ADWEM approach and not to establish a competition between the software or the implementations of the methods.

A.1. Residual based adaptive FEM algorithms. In order to discretize problem (2.2), we introduce a regular family of conforming partitions $\{\mathcal{T}_h\}_h$ of the domain $\overline{\Omega}$ into triangles; let us denote by h_T the diameter of the element $T \in \mathcal{T}_h$. The space H in (2.2) is approximated by $V_h = \left\{v_h \in H \cap \mathcal{C}^0(\overline{\Omega}) : \ v_{h|_T} \in \mathbb{P}_1(T) \text{ for all } T \in \mathcal{T}_h\right\}$, where $\mathbb{P}_1(T)$ is the space of polynomials of degree at most 1, on the element $T \in \mathcal{T}_h$. For simplicity we do not consider any approximation of the data. For problem (4.1)

an equivalence relation between the true error $\|u-u_h\|_{H^1(\Omega)}$ and a residual-based a posteriori error estimator can be proved [32].

For all triangles $T\in\mathcal{T}_h$ we denote by $\mathcal{E}(T)$ the set of its edges and by $\mathcal{E}_{h,\Omega}$ the set of all edges of the triangulation $\mathcal{T}_h$ inside the domain Ω. For each edge $E\in\mathcal{E}_{h,\Omega}$, $\hat{n}_E$ represents a unit vector which is orthogonal to E and $\omega_E=\bigcup_{\{T':\ E\in\mathcal{E}(T')\}}T'$. Given $E\in\mathcal{E}_{h,\Omega}$ and $\varphi\in \mathrm{L}^2(\,\omega_E\,)$ with $\varphi_{|_{T'}}\in\mathcal{C}^0(\,T'\,)$ for all $T'\in\omega_E$, we denote by $[\![\varphi]\!]_E$ the jump of φ across E in the direction $\hat{n}_E$. Finally, let us define the [illegible] $R_{T,h}$ and the [illegible] $J_{E,h}$ for this problem,

$$R_{T,h}=-\triangle u_h-f|_T\,,\qquad J_{E,h}=\left[\!\!\left[\frac{\partial u_h}{\partial\,\hat{n}_E}\right]\!\!\right]_E,$$

and the local and global error estimators,

$$\eta_{R,T}^2=C_0^2\,\|\,R_{T,h}\,\|_{L^2(T)}^2+C_1^2\sum_{E\in\mathcal{E}(T)\,\cap\,\mathcal{E}_{h,\Omega}}\|\,J_{E,h}\,\|_{L^2(E)}^2,\qquad \eta_\Omega^2=\sum_{T\in\mathcal{T}_h}\eta_{R,T}^2\,.$$

The following equivalence relation between the global error estimator and the true error holds: $c\,\eta_\Omega^2\le\|\,u_h-u\,\|_{H^1(\Omega)}^2\le C\,\eta_\Omega^2$.

We refer to the online manual of the code [1] or to [30] for a detailed description of the adaptive methods implemented in ALBERTA. We are presenting results obtained applying the following strategies: uniform refinement, maximum strategy (MS), equidistribution strategy (ES), and guaranteed error reduction strategy (GERS). We briefly recall these adaptive marking strategies for refinement:

MS: mark T with $\eta_{R,T}>\gamma\max_{T'\in\mathcal{T}_h}\eta_{R,T'}$.

ES: mark T with $\eta_{R,T}>\theta\mathrm{tol}/\sqrt{N_T}$, where N_T is the number of elements in $\mathcal{T}_h$.

GERS: mark $T\in\mathcal{A}$ such that $\sum_{T\in\mathcal{A}}R_{T,h}^2>(1-\theta_*)^2\,\eta_\Omega^2$; see [24, 29] for more details on the choice of the set $\mathcal{A}$.

In all the numerical experiments, coarsening is allowed and we use the following parameters, which correspond, except for polynomial_degree and max_iteration, to the default values [1].

Member	Value	Member	Value
polynomial_degree	1	MS_gamma	0.5
estimator C0	1.0	MS_gamma_c	0.1
estimator C1	1.0	ES_theta	0.9
max_iteration	200	ES_theta_c	0.2
refine_bisection	2	GERS_theta_star	0.6
coarsen_allowed	1	GERS_nu	0.1
coarse_bisections	2	GERS_theta_c	0.1

A.2. Numerical examples. Let us consider Examples 3 and 4 in section 4. We will solve them by the adaptive FEMs implemented in ALBERTA and by uniform refinement. These results should be compared with the corresponding ones from the previous section.

Example 3. The solution is obtained using ALBERTA with tolerance tol = 4.0 (chosen to have a final H^1-relative error close to the one in Table 4.10) starting from a coarse grid with $N=2113$ nodes. In Table A.1 we report the quantities δ_{L^2} and δ_{H^1}; in Table A.2 we report the constant c and the slope α of the least squares approximations cN^α of the relative errors with respect to the number of nodes. From these data we can conclude that the behavior of the relative error on these meshes

TABLE A.1

Example 3. Behavior of the adaptive FEMs, number of nodes, and relative errors in some adaptive iterations.

MS			
Iter.	N	$\delta_{L^2(\Omega)}$	$\delta_{H^1(\Omega)}$
0	2113	8.078e+1	1.302e+1
1	2225	7.277e+1	1.304e+1
2	2337	3.669e+1	6.010e+0
3	2556	2.495e+1	5.710e+0
4	2740	5.934e+1	6.950e+0
5	2698	6.171e+1	7.276e+0
6	3112	2.783e+1	4.193e+0
7	2960	2.289e+1	3.893e+0
8	3259	8.540e+0	3.157e+0
9	3243	9.822e+0	3.299e+0
10	4612	9.427e+0	2.224e+0
11	4381	1.251e+1	2.387e+0
12	4825	3.928e+0	1.649e+0
13	6492	5.164e+0	1.247e+0
14	6894	3.990e+0	1.083e+0
15	8328	2.995e+0	8.471e−1
16	8174	2.841e+0	8.480e−1
17	13092	7.083e−1	4.777e−1
18	19644	6.209e−1	3.338e−1
19	23163	4.434e−1	2.843e−1
20	37020	1.218e−1	1.972e−1
21	53319	3.960e−2	1.527e−1
22	88401	2.313e−2	1.091e−1
23	124269	3.768e−2	9.011e−2
24	234983	1.169e−2	6.331e−2
25	358596	5.962e−3	5.065e−2
26	684219	4.500e−3	3.621e−2

ES			
Iter.	N	$\delta_{L^2(\Omega)}$	$\delta_{H^1(\Omega)}$
0	2113	8.072e+1	1.301e+1
1	8275	2.249e+1	5.370e+0
2	26441	1.684e+1	4.482e+0
3	74534	9.302e+0	1.734e+0
4	98914	1.656e+0	7.136e−1
5	87681	2.430e−1	2.967e−1
6	96674	5.811e−3	1.507e−1
7	224574	2.216e−3	7.641e−2
8	605245	2.851e−3	3.968e−2

GERS			
Iter.	N	$\delta_{L^2(\Omega)}$	$\delta_{H^1(\Omega)}$
0	2113	8.112e+1	1.305e+1
1	2142	5.293e+1	9.553e+0
2	2155	4.849e+1	9.077e+0
3	2195	9.724e+1	1.405e+1
4	2220	7.840e+1	1.329e+1
5	2264	4.305e+1	9.828e+0
6	2368	3.599e+1	6.064e+0
7	2464	6.333e+1	7.928e+0
8	2571	3.220e+1	6.013e+0
9	2699	4.957e+1	7.004e+0
10	3041	3.443e+1	5.664e+0
11	3150	2.478e+1	4.443e+0
12	3560	1.408e+1	3.039e+0
13	4056	2.697e+1	3.304e+0
14	16093	4.259e+0	1.138e+0
15	16744	1.689e+0	9.141e−1
16	17723	1.496e+0	7.300e−1
17	18864	1.874e+0	6.714e−1
18	20824	1.175e+0	5.223e−1
19	22958	1.178e+0	4.425e−1
20	26229	6.394e−1	3.589e−1
21	31278	2.613e−1	2.814e−1
22	36930	2.037e−1	2.306e−1
23	44012	7.848e−2	1.961e−1
24	51949	9.507e−2	1.703e−1
25	69388	6.032e−2	1.350e−1
26	84972	4.188e−2	1.161e−1
27	105211	2.095e−2	1.015e−1
28	130982	2.128e−2	8.935e−2
29	179451	1.380e−2	7.456e−2
30	257101	1.177e−2	6.060e−2
31	337177	7.635e−3	5.243e−2
32	474162	8.456e−3	4.390e−2

is still far from the asymptotic one. Comparing results reported in Table A.1 with the results of Table 4.10 we can clearly see the very effective capability of a wavelets method to catch and resolve sharp structures and singularities. The comparison of the number of basis functions (DOF) between the two methods is not necessarily related to the computational cost. In fact, the task of assembling the linear system is usually more expensive for wavelet codes than for FEMs. Moreover, the state of the art for the finite element codes is widely more assessed than for the few and recent wavelets codes, and many improvements may be possible.

Example 4. The solution is obtained using tolerance tol = 0.03, starting from a coarse grid with $N = 2113$ nodes. The final relative errors are $\delta_{L^2(\Omega)} \approx 3\text{e}{-4}$ and $\delta_{H^1(\Omega)} \approx 5\text{e}{-3}$ for the three adaptive methods considered. In Table A.3 (compare with Table 4.12), we report the data of the least squares approximations cN^{α} of

TABLE A.2
Example 3. *Comparison of constants in the error decay.*

	Theoretical uniform	Computed uniform
L^2	$c_0^u N^{-5/6+\epsilon}$	$4.482e{+}3N^{-0.628}$
H^1	$c_1^u N^{-1/3+\epsilon}$	$6.291e{+}2N^{-0.503}$

		Theoretical adaptive	Computed adaptive
MS	L^2	$c_0^a N^{-1}$	$3.067e{+}7N^{-1.777}$
	H^1	$c_1^a N^{-1/2}$	$1.221e{+}4N^{-1.008}$
ES	L^2	$c_0^a N^{-1}$	$2.223e{+}9N^{-2.031}$
	H^1	$c_1^a N^{-1/2}$	$1.120e{+}5N^{-1.097}$
GERS	L^2	$c_0^a N^{-1}$	$9.485e{+}7N^{-1.858}$
	H^1	$c_1^a N^{-1/2}$	$3.816e{+}4N^{-1.101}$

TABLE A.3
Example 4. *Comparison of constants in the error decay.*

	Theoretical uniform	Computed uniform
L^2	$c_0^u N^{-5/6+\epsilon}$	$4.831N^{-0.752}$
H^1	$c_1^u N^{-1/3+\epsilon}$	$2.972N^{-0.438}$

		Theoretical adaptive	Computed adaptive
MS	L^2	$c_0^a N^{-1}$	$2.864N^{-0.717}$
	H^1	$c_1^a N^{-1/2}$	$6.198N^{-0.551}$
ES	L^2	$c_0^a N^{-1}$	$5.930N^{-0.801}$
	H^1	$c_1^a N^{-1/2}$	$4.201N^{-0.491}$
GERS	L^2	$c_0^a N^{-1}$	$4.267N^{-0.760}$
	H^1	$c_1^a N^{-1/2}$	$6.178N^{-0.552}$

the relative errors in L^2- and H^1-norms obtained by the adaptive algorithms and a uniform refinement. In this case, as for the ADWEM algorithm, we note a very good agreement between the theoretical values and the obtained ones for the H^1-norm [9], while for the L^2-norm we have a small mismatch and the slope is closer to the expected one for a linear approximation technique (uniform refinement) than to the expected one for a nonlinear approximation technique (adaptivity).

Acknowledgments. The authors would like to thank the editor and two anonymous referees for several helpful comments and suggestions which significantly improved the paper. This work was written when the second author was at the Dipartimento di Matematica of Politecnico di Torino.

REFERENCES

[1] *ALBERTA: An Adaptive Hierarchical Finite Element Toolbox*, http://www.alberta-fem.de/.
[2] A. BARINKA, T. BARSCH, P. CHARTON, A. COHEN, S. DAHLKE, W. DAHMEN, AND K. URBAN, *Adaptive wavelet schemes for elliptic problems—implementation and numerical experiments*, SIAM J. Sci. Comput., 23 (2001), pp. 910–939.
[3] A. BARINKA, S. DAHLKE, AND W. DAHMEN, *Adaptive application of operators in standard representation*, Adv. Comput. Math., 24 (2006), pp. 5–34.
[4] S. BERRONE AND L. EMMEL, *A realization of a wavelet Galerkin method on nontrivial domains*, Math. Models Methods Appl. Sci., 12 (2002), pp. 1525–1554.
[5] S. BERRONE AND L. EMMEL, *A Realization of a Wavelet Galerkin Method on Non-trivial Domains*, Preprint 33/2001, Dipartimento di Matematica, Politecnico di Torino, Italy, 2001.
[6] S. BERRONE AND T. KOZUBEK, *An Adaptive WEM Algorithm for Solving Elliptic Boundary Value Problems in Fairly General Domains*, Report 38, Dipartimento di Matematica, Politecnico di Torino, Italy, 2004.

[7] S. Berrone and K. Urban, *Adaptive wavelet Galerkin methods on distorted domains: Setup of the algebraic system*, in Curve and Surface Fitting: Saint-Malo 1999, A. Cohen, C. Rabut, and L. L. Schumaker, eds., Vanderbilt University Press, Nashville, TN, pp. 65–74.
[8] S. Bertoluzza, C. Canuto, and K. Urban, *On the adaptive computation of integrals of wavelets*, Appl. Numer. Math. 34 (2000), pp. 13–38.
[9] P. Binev, W. Dahmen, and R. DeVore, *Adaptive finite element methods with convergence rates*, Numer. Math., 97 (2004), pp. 219–268.
[10] C. Canuto, A. Tabacco, and K. Urban, *The wavelet element method.* I. *Construction and analysis*, Appl. Comput. Harmon. Anal., 6 (1999), pp. 1–52.
[11] C. Canuto, A. Tabacco, and K. Urban, *The wavelet element method.* II. *Realization and additional features in* $2D$ *and* $3D$, Appl. Comput. Harmon. Anal., 8 (1999), pp. 123–165.
[12] A. Cohen, W. Dahmen, and R. DeVore, *Adaptive wavelet methods for elliptic operator equations: Convergence rates*, Math. Comp., 70 (2001), pp. 27–75.
[13] A. Cohen, W. Dahmen, and R. DeVore, *Adaptive wavelet methods.* II. *Beyond the elliptic case*, Found. Comput. Math., 2 (2002), pp. 203–245.
[14] A. Cohen and R. Masson, *Wavelet adaptive method for second order elliptic problems: Boundary conditions and domain decomposition*, Numer. Math., 86 (2000), pp. 193–238.
[15] S. Dahlke, *Besov regularity for elliptic boundary value problems in polygonal domains*, Appl. Math. Lett, 12 (1999), pp. 31–36.
[16] W. Dahmen, *Wavelet and multiscale methods for operator equations*, Acta Numer., 6 (1997), pp. 55–228.
[17] W. Dahmen, *Wavelet methods for PDEs: Some recent developments* J. Comput. Appl. Math., 128 (2001), pp. 133–185.
[18] W. Dahmen, *Multiscale and wavelet methods for operator equations*, in Multiscale Problems and Methods in Numerical Simulations, C. Canuto, ed., Springer-Verlag, Berlin, 2003, pp. 31–96.
[19] W. Dahmen and A. Kunoth, *Multilevel preconditioning*, Numer. Math., 63 (1992), pp. 315–344.
[20] W. Dahmen, A. Kunoth, and K. Urban, *Biorthogonal spline wavelets on the interval-stability and moment conditions*, Appl. Comput. Harmon. Anal., 6 (1999), pp. 132–196.
[21] W. Dahmen, S. Prössdorf, and R. Schneider, *Multiscale methods for pseudo-differential equations on smooth manifolds*, in Proceedings of the International Conference on Wavelets: Theory, Algorithms and Applications, C. K. Chui, L. Montefusco, and L. Puccio, eds., Academic Press, New York, 1994, pp. 385–424.
[22] W. Dahmen and R. Schneider, *Composite wavelet bases for operator equations*, Math. Comp., 68 (1999), pp. 1533–1567.
[23] R. DeVore, *Nonlinear approximation*, Acta Numer., 7 (1998), pp. 51–150.
[24] W. Dörfler, *A convergent adaptive algorithm for Poisson's equation*, SIAM J. Numer. Anal., 33 (1996), pp. 1106–1124.
[25] P. Grisvard, *Singularities in Boundary Value Problems*, Rech. Math. Appl. 22, Springer-Verlag, Berlin, 1992.
[26] S. Grivet-Talocia and A. Tabacco, *Wavelets on the interval with optimal localization*, Math. Models Methods Appl. Sci., 10 (2000), pp. 441–462.
[27] S. Jaffard, *Wavelets methods for fast resolution of elliptic problems*, SIAM J. Numer. Anal., 29 (1992), pp. 965–986.
[28] A. Kunoth, A.S. Piquemal, and J. Vorloeper, *Multilevel Preconditioners for Discretizations of Elliptic Boundary Value Problems, Experiences with Different Software Packages*, IGPM report 194, RWTH Aachen, Aachen, Germany, 2000.
[29] P. Morin, R. H. Nochetto, and K. G. Siebert, *Data oscillation and convergence of adaptive FEM*, SIAM J. Numer. Anal., 38 (2000), pp. 466–488.
[30] A. Schmidt and K. G. Siebert, *Design of Adaptive Finite Element Software the Finite Element Toolbox ALBERTA*, Lect. Notes Comput. Sci. Eng. 42, Springer-Verlag, Berlin, 2005.
[31] R. Stevenson, *An optimal adaptive finite element method*, SIAM J. Numer. Anal., 42 (2005), pp. 2188–2217.
[32] R. Verfürth, *A Review of A Posteriori Error Estimation and Adaptive Mesh-Refinement Techniques*, John Wiley & Sons, Chichester, UK, New York, 1996.

SIAM J. SCI. COMPUT.
Vol. 28, No. 6, pp. 2139–2161
© 2006 Society for Industrial and Applied Mathematics

A PREDICTOR-CORRECTOR ALGORITHM FOR REACTION-DIFFUSION EQUATIONS ASSOCIATED WITH NEURAL ACTIVITY ON BRANCHED STRUCTURES*

M. J. REMPE† AND D. L. CHOPP†

Abstract. In this paper, we present a numerical method for solving reaction-diffusion equations on one-dimensional branched structures. The method effectively decouples the many branches at the nodal junctions so that the equations can be solved as a system of smaller problems that are purely tridiagonal. This strategy enables a number of improvements. In particular, difficulties associated with implicit methods for closed loops in the network are eliminated, and spatial adaptivity can be easily implemented. We apply this method to the electrical activity of a neuron. A neuron can be effectively represented as a one-dimensional branched structure, and the model equations, based on the Hodgkin–Huxley cable equations, are a set of reaction equations coupled to a single diffusion process. Spatially adaptive methods are particularly effective for neuronal simulations as the region of activity is often very localized in space. Since the algorithm scales with activity instead of network size, it will make a substantial reduction in computational cost over existing methods.

Key words. reaction-diffusion equation, neural biology, finite difference methods

AMS subject classifications. 35K57, 92C20, 65M06

DOI. 10.1137/050643210

1. Background. In this paper, we propose a numerical method for solving reaction-diffusion equations on a branched system of one-dimensional structures. More precisely, let $B_i = [0, L_i]$ for each $i = 1, \ldots, N_b$ be individual branches. The branches are joined by nodal points p_j, $j = 1, \ldots, N_n$, which couple two or more branches, $B_{i_j^\pm}$, through their end points. The "$\pm$" indicates whether the $x = 0$ or $x = L_{i_j}$ end of B_{i_j} is joined at p_j. For example, Figure 1.1 illustrates a typical relationship between three branches and a node. The reaction-diffusion system can then be expressed as follows, coupled with appropriate boundary conditions on branch ends that are not joined at a node:

$$V_t^i = AV_{xx}^i + R(V^i), \quad x \in B_i,\ i = 1, \ldots, N_b, \tag{1.1}$$

$$W_t^j = \sum_{i_j} \pm AW_x^{i_j} + R(W^j) \quad \text{at } p_j,\ j = 1, \ldots, N_n. \tag{1.2}$$

Here $V^i(x)$ is the value of the solution along branch B_i, W^j is the solution at the single node p_j, and $R(V)$ is a possibly nonlinear reaction term. Thus, we have a collection of separate reaction-diffusion equations, (1.1), coupled at nodes where conservation-of-flux equation (1.2) is applied.

One example of where this type of system arises is in modeling excitation of neurons. In this case V represents the voltage across the cell membrane, and $R(V)$ represents the currents due to the movement of ions through channels in the membrane. Written out with all the reaction terms, and their corresponding ordinary differential equations, the system is referred to as the Hodgkin–Huxley equations [10].

*Received by the editors October 20, 2005; accepted for publication (in revised form) June 12, 2006; published electronically December 5, 2006. This work is supported by NIH under contract number R01 NS046064.

http://www.siam.org/journals/sisc/28-6/64321.html

†Department of Engineering Sciences and Applied Mathematics, Northwestern University, 2145 Sheridan Rd., Evanston, IL 60208 (mjrempe@northwestern.edu, chopp@northwestern.edu).

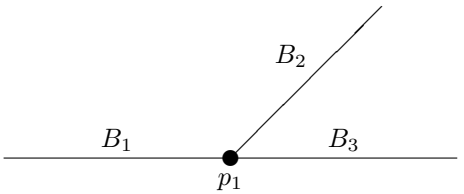


FIG. 1.1. *Sample branched structure with three branches and one node. The node p_1 joins B_{1+}, B_{2-}, and B_{3-}.*

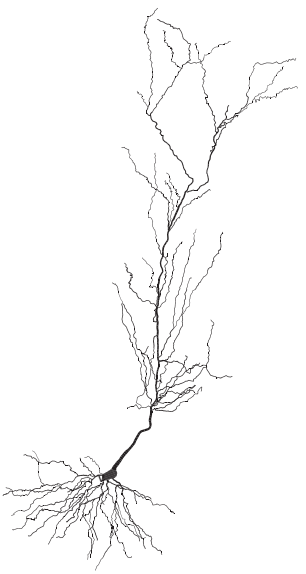


FIG. 1.2. *Example cell shape. This is the shape of a subicular pyramidal cell from rat hippocampus. Branches extend outward from the cell body, or soma.*

In terms of structure, one neuron may have hundreds of branches, called dendrites. A representative example of a neuron's shape is shown in Figure 1.2, which is a pyramidal cell neuron from the subiculum region of rat hippocampus. This shape was traced out using a real cell that was stained and reconstructed on a computer. Note the large number of long, slender branches extending out in a tree-like manner. It is important to note that for each branch, since the length is so much greater than the diameter, a one-dimensional mathematical model for the branches is reasonable. A number of software packages, including NEURON [8] and GENESIS [1], take this approach to solve the Hodgkin–Huxley reaction-diffusion system on realistic models of branched neurons. These neuron simulation packages trace their roots to the key work of Cooley and Dodge, who presented a numerical algorithm to solve the Hodgkin–Huxley equations using Crank–Nicolson for the diffusion term and an iterative procedure for the nonlinear reaction term [3]. This was the first time a second-order numerical method was demonstrated for the Hodgkin–Huxley equations.

Hines [6] followed up the work of Cooley and Dodge by presenting a number of numerical techniques that increase the efficiency of the numerical method, especially on branched structures. One of the improvements was a method for numbering the branches of the structure and the compartments within each branch such that the

resulting linear system for the voltage on the entire structure can be made upper triangular and solved as efficiently as the system for an unbranched cable. Cai, Walsh, and McGee [2] also presented a similar branch-numbering procedure to update the voltage of the tree in $O(N)$ operations. Note that both Hines's and Cai's approaches use one linear system for the entire tree structure and hence are not easily made spatially adaptive.

Mascagni [12] took a different approach to the same problem by using an exact domain decomposition algorithm on the branched structure. For each branch the principle of superposition is employed to construct the solution as the sum of two or three solutions to linear systems for that branch. This approach separates the problem into small linear systems, one for each branch. These linear systems are tridiagonal, so the computational cost for updating one branch is $O(N)$ operations, where N is the number of compartments in that branch. The benefits of this approach are that it is easily made parallelizable and that it can handle closed loops, which the earlier methods could not. However, the superposition approach requires an additional solution of a dense linear system corresponding to the voltage at the branch points, which couples all the branches together, and consequently the evolution on the whole cell is again required.

One observation about neuronal activity is that often large regions of the network are essentially at equilibrium. For this reason, spatial adaptivity is expected to yield improved performance for computing large neuron simulations. None of the aforementioned methods can be made spatially adaptive. In this paper, we present a numerical method that combines an implicit method on the branches with a predictor-corrector scheme for the branch points of the tree structure, allowing the structure to be split up into individual branches. In this way our numerical method provides a framework for parallelization: each branch of the tree structure can be computed independently by a separate processor. In contrast to the method developed by Mascagni, our method avoids the dense linear system for the nodes and hence is more easily parallelized. In addition, our method is able to do local spatial adaptivity because each branch is completely uncoupled from the other branches. Furthermore, because the branches are uncoupled, incorporating spatial adaptivity is natural and straightforward: branches are updated only if they are active and turned off as they return to the equilibrium resting voltage. Spatial adaptivity within the tree structure is a novel development in neural simulations.

To explain our algorithm for splitting up a branched structure we will focus on the Hodgkin–Huxley equations on a branched neuron; the details are presented in section 2. In section 3, we describe the application of Crank–Nicolson and backward Euler methods to the Hodgkin–Huxley cable equations. In section 4, we present both first- and second-order accurate versions of the new predictor-corrector scheme. We will discuss the convergence and stability properties of the new methods in sections 5 and 6, respectively. In section 7 we present an example for both methods and compare the accuracy of the results to those of standard methods. In section 8 we present a number of examples of simulations involving the spatial adaptivity algorithm including simulations on complex branched structures. Finally, in section 9 we give an example on a closed loop.

2. Hodgkin–Huxley mathematical model. The Hodgkin–Huxley equations are written out as follows:

$$C_m \frac{\partial V}{\partial t} = \frac{a}{2R_a} \frac{\partial^2 V}{\partial x^2} - R(V), \tag{2.1a}$$

where

(2.1b)
$$R(V) = \overline{g_{Na}} m^3 h (V - E_{Na}) + \overline{g_K} n^4 (V - E_K) + g_{leak}(V - E_{leak}) - \frac{I_e}{2\pi a}\delta(x - x_0),$$

(2.1c)
$$\begin{aligned} \frac{dm}{dt} &= \alpha_m(V)(1-m) - \beta_m(V)m, \\ \frac{dn}{dt} &= \alpha_n(V)(1-n) - \beta_n(V)n, \\ \frac{dh}{dt} &= \alpha_h(V)(1-h) - \beta_h(V)h, \end{aligned}$$

(2.1d)
$$\begin{aligned} \alpha_m(V) &= \frac{0.1(25-V)}{e^{(25-V)/10}-1}, \qquad \beta_m(V) = 4e^{-V/18}, \\ \alpha_n(V) &= \frac{0.01(10-V)}{e^{(10-V)/10}-1}, \qquad \beta_n(V) = 0.125e^{-V/80}, \\ \alpha_h(V) &= 0.07e^{-V/20}, \qquad \beta_h(V) = \frac{1}{e^{30-V/10}+1}, \end{aligned}$$

where I_e is the stimulating current from an electrode, C_m is the membrane capacitance, R_a is the resistance, a is the radius, and $\overline{g_{Na}}$, $\overline{g_K}$, g_{leak}, E_{Na}, E_K, and E_{leak} are constants. The final term in (2.1c) represents the current due to a stimulating electrode at a location x_0. The variables m, n, and h are dimensionless. Equations (2.1c) describe the kinetics of the variables m, n, and h used in (2.1c). Equation (2.1d) gives the form of $\alpha(V)$ and $\beta(V)$ used in (2.1c). In this case, $R(V)$ is a current, and the terms in (2.1c) model the flow of current, through the membrane due to the movement of ions through channels. The Hodgkin–Huxley model in particular includes the currents due to the movement of sodium ions and potassium ions, as well as a general nonspecific ion current referred to as "leak" current. These currents are each reasonably modeled using Ohm's law: $I = gV$. However, before being used in this model, Ohm's law has been modified from its simple form in two ways. First, instead of the absolute membrane voltage, V, the voltage relative to each ion's reversal potential (E_{Na}, E_K, or E_{leak}) is used. Second, since the conductance depends on channels in the membrane that open and close, the conductance, g, is not constant but instead is the product of a constant and one or more dimensionless rate variables (m^3h in the case of sodium, n^4 in the case of potassium). Also referred to as "gating variables," these rate variables range from 0 to 1, where a value of 0 means all of the channels for that ion are closed and 1 means all of the channels for that ion are open. For instance, a value of 1 for n would mean that the potassium conductance is at its maximum, $\overline{g_K}$. The equations (2.1d) were determined empirically by Hodgkin–Huxley to fit experimental data.

The Hodgkin–Huxley model serves as the framework for a multitude of other mathematical models that include the influence of many different kinds of ion channels, not just sodium and potassium. Each of these can be modeled analogously to the sodium and potassium channels used in the Hodgkin–Huxley model. The constant conductances, the form of the gating variables, and the form of the rate functions are often not known beforehand but instead are determined by fitting the model to

experimental data. A common practice with mathematical models of this sort is to vary the form of the rate functions α and β along with the constant conductances to make the model output match up with experimental results. Once the model is fit to data, it can be used to make predictions that can be verified by experiments.

3. Numerical methods. The most common way to solve this reaction-diffusion system for the voltage involves using an implicit method for the diffusion term, either Crank–Nicolson [7], [3] or backward Euler [11]. The nonlinear reaction terms are then updated separately using an iterative procedure [3], [11] or with the trapezoidal rule using a staggered time step [6]. In this section we describe both the backward Euler and the Crank–Nicolson methods for solving the system (2.1) on a single branch.

3.1. Backward Euler. Assuming constant grid spacing, Δx, and using backward Euler to discretize the system (2.1) yields the equation

$$\lambda V_{i-1}^{t+\Delta t} - 2\lambda V_i^{t+\Delta t} + \lambda V_{i+1}^{t+\Delta t} = \mu\left(V_i^{t+\Delta t} - V_i^t\right) + \Gamma^{t+\Delta t} V_i^{t+\Delta t} - \gamma^{t+\Delta t}, \tag{3.1}$$

where we have made the following substitutions to simplify the notation:

$$\begin{aligned}
\lambda &= \frac{a}{2R_a \Delta x^2},\\
\Gamma^t &= \overline{g_{Na}} m^3(t) h(t) + \overline{g_K} n^4(t) + g_{leak},\\
\gamma^t &= \overline{g_{Na}} m^3(t) h(t) E_{Na} + \overline{g_K} n^4(t) E_K + g_{leak} E_{leak},\\
\mu &= \frac{C_m}{\Delta t}.
\end{aligned}$$

In this case since the reaction term, R, is linear in $V_i^{t+\Delta t}$ we can group the $V_i^{t+\Delta t}$ terms to the left-hand side, giving a matrix equation for the voltage:

$$\lambda V_{i-1}^{t+\Delta t} - \left(2\lambda + \mu + \Gamma^{t+\Delta t}\right) V_i^{t+\Delta t} + \lambda V_{i+1}^{t+\Delta t} = -\mu V_i^t - \gamma^{t+\Delta t}. \tag{3.2}$$

To update the gating variables, m, n, and h, we discretize (2.1c) using backward Euler as well:

$$\frac{m^{t+\Delta t} - m^t}{\Delta t} = \alpha_m(V^{t+\Delta t})(1 - m^{t+\Delta t}) - \beta_m(V^{t+\Delta t}) m^{t+\Delta t}. \tag{3.3}$$

When solved for $m^{t+\Delta t}$ this yields

$$m^{t+\Delta t} = \frac{m^t + \Delta t \alpha_m(V^{t+\Delta t})}{1 + \Delta t \beta_m(V^{t+\Delta t}) + \Delta t \alpha_m(V^{t+\Delta t})}. \tag{3.4}$$

Since (3.2) and (3.4) are coupled through $V^{t+\Delta t}$, they must be solved simultaneously. One approach to solving this nonlinear system is Picard iteration as outlined in [11] and [13]. The solution procedure is as follows:

1. Solve (3.2) for the voltage at time $t + \Delta t$ using values for m, n, and h at time t.
2. Update m, n, and h using (3.4) and the voltage computed in step 1.
3. Correct the voltage using (3.2) and m, n, and h computed in step 2.
4. Correct the gating variables using the new voltage computed in step 3.
5. Return to step 2 until the voltage and gating variables have converged.

Since backward Euler is first order in time anyway, a simpler approach is to update the voltage and the gating variables only once instead of using Picard iteration [9].

3.2. Crank–Nicolson. A similar numerical method used to solve (2.1) is the Crank–Nicolson method. Hines's algorithm [6] performs Crank–Nicolson in two separate steps. The first step is a backward Euler step of size $\Delta t/2$ for (2.1a), which is exactly the same as in (3.2) except that a step size of $\Delta t/2$ is used instead of Δt:

$$\lambda V_{i-1}^{t+\Delta t/2} - \left(2\lambda + 2\mu + \Gamma^{t+\Delta t/2}\right) V_i^{t+\Delta t/2} + \lambda V_{i+1}^{t+\Delta t/2} = -2\mu V_i^t - \gamma^{t+\Delta t/2}. \tag{3.5}$$

Having solved (3.5) for $V_i^{t+\Delta t/2}$, an explicit step is used to update V from time t to time $t + \Delta t$:

$$V_i^{t+\Delta t} = 2V_i^{t+\Delta t/2} - V_i^t. \tag{3.6}$$

The combination of (3.5) and (3.6) to advance V over a full time interval is equivalent to the Crank–Nicolson method [4]. Note that (3.5) is linear in V if the dimensionless gating variables (m, n, and h) are known at time $t + \Delta t/2$, a property referred to as "conditional linearity" by Mascagni and Sherman [13]. Because the system is conditionally linear, full second-order accuracy in time can be maintained if the values of m, n, and h are known at offset time steps. That is, if the voltage is computed at times t, $t + \Delta t$, $t + 2\Delta t$, etc. and the gating variables are computed at times $t + \Delta t/2$, $t + 3\Delta t/2$, $t + 5\Delta t/2$, then they can be used directly in (3.5) to update the voltage at time $\Delta t/2$. The final ingredient needed to make sure the method is second order in time is a second-order accurate method updating the gating variables. This is achieved by applying the trapezoidal rule to (2.1c):

$$\frac{m^{t+\Delta t/2} - m^{t-\Delta t/2}}{\Delta t} = \alpha_m(V^t) - (\alpha_m(V^t) + \beta_m(V^t))\frac{m^{t+\Delta t/2} + m^{t-\Delta t/2}}{2}. \tag{3.7}$$

Solving this for $m^{t+\Delta t/2}$ yields

$$m^{t+\Delta t/2} = \left(\frac{\alpha_m}{\frac{1}{\Delta t} + \frac{1}{2}(\alpha_m + \beta_m)}\right) + \left(\frac{\frac{1}{\Delta t} - \frac{1}{2}(\alpha_m + \beta_m)}{\frac{1}{\Delta t} + \frac{1}{2}(\alpha_m + \beta_m)}\right) m^{t-\Delta t/2}. \tag{3.8}$$

Once m, n, and h are updated in this manner, they can be placed in the coefficient matrix which multiplies the voltage vector in (3.5). Since this linear system is tridiagonal, its solution can be found in $O(N)$ operations, and consequently, the overall computational cost to update one branch is $O(N)$.

3.3. Branching. At the points where branches connect, an additional conservation equation needs to be solved:

$$\frac{1}{A} \sum_{\substack{branches \\ into \\ node}} \pm \left(\frac{\pi a^2}{R_a}\frac{\partial V}{\partial x}\right) = C_m \left.\frac{\partial V}{\partial t}\right|_{node} + \Gamma V_{node} - \gamma, \tag{3.9}$$

where Γ, γ are defined as before and A is the surface area of the node.

This equation represents a conservation of charge at the node: the sum of the axial currents coming into the node due to the branches (left-hand side) is equal to the current flowing out of the node through the membrane (the right-hand side). When evaluating the axial currents, care must be taken with regard to the direction of flow, hence the need for the $\pm$ sign.

Since Hines's method updates the entire branching structure using one large linear system, his method handles branch points by simply changing the corresponding lines

in the linear system to solve (3.9) instead of (3.5). The result is a linear system for the tree that can be solved in $O(N)$ operations, where N is the number of compartments in the tree structure. One of the main purposes of the algorithm presented here is to avoid the use of this large linear system and instead use small linear systems, one for each branch.

4. Predictor-corrector numerical schemes. To split up the structure into branches and nodes, we propose two predictor-corrector schemes for the nodes, one to be used with backward Euler for the branches and one to be used with the two-step version of Crank–Nicolson for the branches. For the case of backward Euler, the numerical scheme for the nodes gives a numerical method that is second order in space and first order in time, while for the Crank–Nicolson case the scheme is second order in time and space. Each method costs $O(N)$ operations. To illustrate the algorithm, consider the simple branching structure of two branches connected by a node as shown in Figure 4.1. The point labeled V_0 represents the node. Both the predictor-corrector algorithms for updating the node will be described using this structure.

4.1. Backward Euler predictor-corrector. To update both branches and the node from time t to time $t + \Delta t$, the backward Euler predictor-corrector algorithm proceeds as follows. First, (3.9) is discretized using an explicit method:

$$C_m \frac{V_0^* - V_0^t}{\Delta t} = \frac{1}{A}\left[\frac{\pi a^2}{R_a}\left\{\frac{V_1^t - V_0^t}{\Delta x} - \frac{V_0^t - V_{-1}^t}{\Delta x}\right\}\right] - \Gamma^{t+\Delta t} V_0^* + \gamma^{t+\Delta t}, \tag{4.1}$$

where V_0^* is a tentative approximation for $V_0^{t+\Delta t}$. Solving for V_0^* gives the update

$$V_0^* = \left(\frac{1}{1 + \frac{1}{\mu}\Gamma^{t+\Delta t}}\right)\left[V_0^t + \frac{\Delta t \pi}{C_m A}\frac{a^2}{R_a}\left\{\frac{V_1^t - V_0^t}{\Delta x} - \frac{V_0^t - V_{-1}^t}{\Delta x}\right\} + \frac{1}{\mu}\gamma^{t+\Delta t}\right]. \tag{4.2}$$

Second, update the voltage, $V_i^{t+\Delta t}$ for $i \neq 0$, on the branches using backward Euler for a full time step using (3.2):

$$\lambda V_{i-1}^{t+\Delta t} - \left(2\lambda + \mu + \Gamma^{t+\Delta t}\right) V_i^{t+\Delta t} + \lambda V_{i+1}^{t+\Delta t} = -\mu V_i^t - \gamma^{t+\Delta t} \quad \text{for } |i| > 1, \tag{4.3}$$

$$\lambda V_{-2}^{t+\Delta t} - \left(2\lambda + \mu + \Gamma^{t+\Delta t}\right) V_{-1}^{t+\Delta t} = -\lambda V_0^* - \mu V_{-1}^t - \gamma^{t+\Delta t}, \tag{4.4}$$

$$-\left(2\lambda + \mu + \Gamma^{t+\Delta t}\right) V_1^{t+\Delta t} + \lambda V_2^{t+\Delta t} = -\lambda V_0^* - \mu V_1^t - \gamma^{t+\Delta t}. \tag{4.5}$$

Note that in (4.3)–(4.5), the unknowns are on the left side of the equations, and the known values are on the right. Note also that (4.4) and (4.5) are the same as equation (4.3) with $V_0^{t+\Delta t}$ replaced with V_0^*.

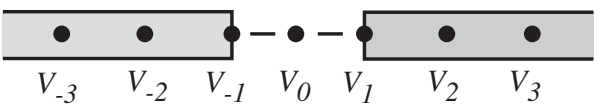


FIG. 4.1. *Sample branching structure. This simple two-branch, one-node structure is used to illustrate the predictor-corrector updating scheme.*

Third, the values for the nodes at time $t+\Delta t$ are corrected using an implicit-like corrector step. This step comes from discretizing (3.9) using backward Euler and is meant to mimic what would have happened at the node if it was updated using backward Euler like the branches:

$$C_m \frac{V_0^{t+\Delta t} - V_0^t}{\Delta t} = \frac{1}{A}\left[\frac{\pi a^2}{R_a}\left\{\frac{V_1^{t+\Delta t} - V_0^{t+\Delta t}}{\Delta x} - \frac{V_0^{t+\Delta t} - V_{-1}^{t+\Delta t}}{\Delta x}\right\}\right] - \Gamma^{t+\Delta t} V_0^{t+\Delta t} + \gamma^{t+\Delta t}. \tag{4.6}$$

Since $V_{-1}^{t+\Delta t}$ and $V_1^{t+\Delta t}$ were computed in the second step, this can be solved explicitly for $V_0^{t+\Delta t}$, the corrected node voltage at time $t+\Delta t$. The update looks like

$$V_0^{t+\Delta t} = \frac{1}{\mu + \frac{2\pi a^2}{A R_a \Delta x} + \Gamma^{t+\Delta t}}\left[\mu V_0^t + \frac{1}{A}\left\{\frac{\pi a^2}{R_a \Delta x}\left(V_1^{t+\Delta t} + V_{-1}^{t+\Delta t}\right)\right\} + \gamma^{t+\Delta t}\right]. \tag{4.7}$$

This completes the update for the nodes and branches using backward Euler. The steps of this algorithm are illustrated on a two-branch structure in Figure 4.2a.

4.2. Crank–Nicolson predictor-corrector. The Crank–Nicolson version of the predictor-corrector scheme is similar to the backward Euler version, but requires offsetting the update of the voltage and the gating variables, as described in section 3.2. The first two steps are exactly like the backward Euler predictor-corrector except that a step of size $\Delta t/2$ is taken instead of Δt:

$$V_0^* = \left(\frac{1}{1 + \frac{1}{2\mu}\Gamma^{t+\Delta t/2}}\right)\left[V_0^t + \frac{\Delta t \pi}{C_m A}\frac{a^2}{R_a}\left\{\frac{V_1^t - V_0^t}{\Delta x} - \frac{V_0^t - V_{-1}^t}{\Delta x}\right\} + \frac{1}{2\mu}\gamma^{t+\Delta t/2}\right]. \tag{4.8}$$

As before, V_0^* is a tentative approximation for $V_0^{t+\Delta t/2}$.

Second, the voltage of the branches is updated for half of a time step using the backward Euler method, (3.2):

$$\lambda V_{i-1}^{t+\Delta t/2} - \left(2\lambda + 2\mu + \Gamma^{t+\Delta t/2}\right) V_i^{t+\Delta t/2} + \lambda V_{i+1}^{t+\Delta t/2} = -2\mu V_i^t - \gamma^{t+\Delta t/2} \quad \text{for } |i| > 1, \tag{4.9}$$

$$\lambda V_{-2}^{t+\Delta t/2} - \left[2\lambda + 2\mu + \Gamma^{t+\Delta t/2}\right] V_{-1}^{t+\Delta t/2} = -\lambda V_0^* - 2\mu V_{-1}^t - \gamma^{t+\Delta t/2}, \tag{4.10}$$

$$-\left[2\lambda + 2\mu + \Gamma^{t+\Delta t/2}\right] V_1^{t+\Delta t/2} + \lambda V_2^{t+\Delta t/2} = -\lambda V_0^* - 2\mu V_1^t - \gamma^{t+\Delta t/2}. \tag{4.11}$$

Third, the same implicit-like corrector step as for backward Euler is used for the nodes, except for a time step of $\Delta t/2$:

$$C_m \frac{V_0^{t+\Delta t/2} - V_0^t}{\Delta t/2} = \frac{1}{A}\left[\frac{\pi a^2}{R_a}\left\{\frac{V_1^{t+\Delta t/2} - V_0^{t+\Delta t/2}}{\Delta x} - \frac{V_0^{t+\Delta t/2} - V_{-1}^{t+\Delta t/2}}{\Delta x}\right\}\right] - \Gamma^{t+\Delta t/2} V_0^{t+\Delta t/2} + \gamma^{t+\Delta t/2}. \tag{4.12}$$

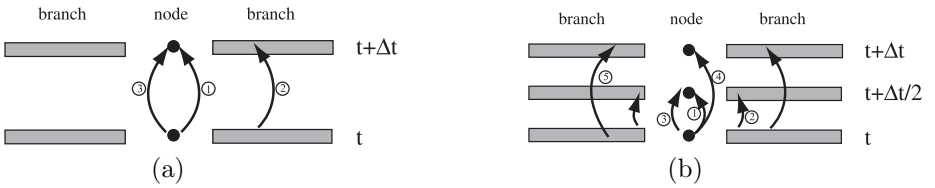


FIG. 4.2. *Each node update scheme is illustrated on a simple configuration of two branches connected by a node. Shown in* (a) *is the backward Euler predictor-corrector method. The steps are as follows:* (1) *An explicit step of size* Δt *is taken for the nodes.* (2) *The branches use this temporary value as a boundary condition as they are updated a full time step using backward Euler.* (3) *Finally the node value is corrected using an implicit-like corrector step as described in the text. Shown in* (b) *are the steps of the Crank–Nicolson predictor-corrector algorithm:* (1) *an explicit step of size* $\Delta t/2$ *for the nodes,* (2) *a backward Euler step of size* $\Delta t/2$ *for the branches,* (3) *an implicit-like corrector step of size* $\Delta t/2$ *for the nodes,* (4) *an explicit step for the branches to update them a full time step, and finally,* (5) *an explicit step for the nodes to update them the full time step.*

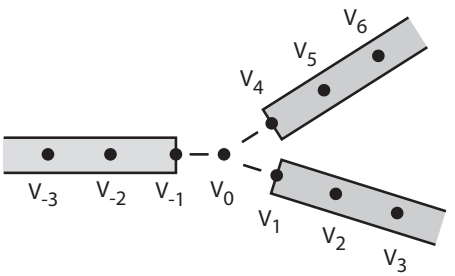


FIG. 4.3. *Three-branch structure. The predictor-corrector updating scheme is described on this three-branch structure.*

As for backward Euler, $V_{-1}^{t+\Delta t/2}$ and $V_1^{t+\Delta t/2}$ come from the second step of the algorithm, so this equation can be solved explicitly for $V_0^{t+\Delta t/2}$. The update looks like

$$V_0^{t+\Delta t/2} = \frac{1}{2\mu + \frac{2\pi a^2}{R_a \Delta x} + \Gamma^{t+\Delta t/2}} \times \left[2\mu V_0^t + \frac{1}{A} \left\{ \frac{\pi a^2}{R_a \Delta x} (V_1^{t+\Delta t/2} + V_{-1}^{t+\Delta t/2}) \right\} + \gamma^{t+\Delta t/2} \right]. \tag{4.13}$$

Finally, the explicit step, (3.6), is carried out for both the branches and the nodes to update the entire tree a full time step. See the schematic in Figure 4.2(b) for a step-by-step description of the Crank–Nicolson update of this simple two-branch, one-node structure.

4.3. Backward Euler predictor-corrector on a three-branch structure. Both schemes presented here for two branches and one node are easily extended to nodes connecting many branches. Consider the three-branch structure shown in Figure 4.3.

To update this node, the first step is an explicit step analogous to (4.1):

$$C_m \frac{V_0^* - V_0^t}{\Delta t} = \frac{1}{A} \left[\frac{\pi a^2}{R_a} \left\{ \frac{V_1^t - V_0^t}{\Delta x} + \frac{V_4^t - V_0^t}{\Delta x} - \frac{V_0^t - V_{-1}^t}{\Delta x} \right\} \right] - \Gamma^{t+\Delta t} V_0^* + \gamma^{t+\Delta t}. \tag{4.14}$$

Solving for V_0^* gives the update for this case:

(4.15)
$$V_0^* = \left(\frac{1}{1+\frac{1}{\mu}\Gamma^{t+\Delta t}}\right)\left[V_0^t + \frac{\Delta t\pi}{C_m A}\frac{a^2}{R_a}\left\{\frac{V_1^t - V_0^t}{\Delta x} + \frac{V_4^t - V_0^t}{\Delta x} - \frac{V_0^t - V_{-1}^t}{\Delta x}\right\} + \frac{1}{\mu}\gamma^{t+\Delta t}\right].$$

The update for the branches is identical to (4.3)–(4.5). The third step is also the same as in (4.6) and (4.7) with the addition of a term corresponding to the third branch:

(4.16)
$$C_m\frac{V_0^{t+\Delta t} - V_0^t}{\Delta t} = \frac{1}{A}\left[\frac{\pi a^2}{R_a}\left\{\frac{V_1^{t+\Delta t} - V_0^{t+\Delta t}}{\Delta x} + \frac{V_4^{t+\Delta t} - V_0^{t+\Delta t}}{\Delta x} - \frac{V_0^{t+\Delta t} - V_{-1}^{t+\Delta t}}{\Delta x}\right\}\right] - \Gamma^{t+\Delta t}V_0^{t+\Delta t} + \gamma^{t+\Delta t}.$$

With this additional term the update is

(4.17)
$$V_0^{t+\Delta t} = \frac{1}{\mu + \frac{3\pi a^2}{AR_a\Delta x} + \Gamma^{t+\Delta t}}\left[\mu V_0^t + \frac{1}{A}\left\{\frac{\pi a^2}{R_a\Delta x}\left(V_1^{t+\Delta t} + V_{-1}^{t+\Delta t}\right)\right\} + \gamma^{t+\Delta t}\right].$$

This step completes the update for the node and the three branches connected to it. The Crank–Nicolson method on this three-branch structure is analogous. It should be clear how a node with n branches connected to it can be updated with this technique. Each additional branch connecting to the node adds another term to the summation in (3.9), which means that (4.14), (4.15), and (4.16) each have an additional term. The added term is of the form $\frac{V_j - V_0}{\Delta x}$. The corrector step, (4.17), then includes the voltage from all n branches, and the coefficient in the denominator of (4.17) reflects the number of branches connected to this node.

Notice how both predictor-corrector methods reduce the problem of updating the entire tree to solving a relatively small linear system for each branch instead of one large system for the entire cell. While the Crank–Nicolson predictor-corrector scheme has more intermediate steps to update the voltage one time step as compared to Hines's method, the overall order of the method is still $O(N)$ and does not introduce any additional linear systems to be solved. Considering that the number of nodes in a neuron is relatively much smaller than the number of compartments in the model, the extra computing done in the predictor-corrector scheme for the nodes makes a negligible contribution to overall computing time.

5. Convergence. We expect the overall accuracy of the new predictor-corrector method to be the same as the underlying numerical methods (backward Euler and Crank–Nicolson). The first test we performed to investigate the numerical convergence of our schemes was to use them on the simple configuration mentioned in section 3: two branches connected by a node.

Figure 5.1 shows convergence plots for the backward Euler predictor-corrector scheme for both grid size and time step indicating second-order convergence in space and first-order in time. To test the space convergence, we fixed the time step while successively reducing the grid spacing. The analogous test was carried out to test the time convergence. Since there is not an exact solution to compare to, a solution was found with a very small time step and grid spacing that did not change when the grid

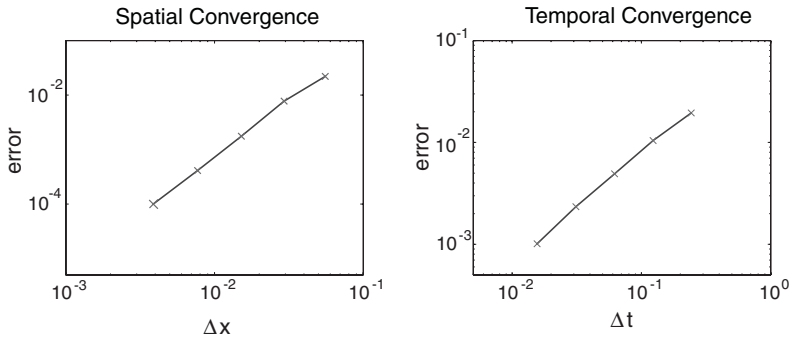


FIG. 5.1. *Convergence of the backward Euler method. Spatial convergence of the backward Euler method holding Δt fixed (left). Temporal convergence of backward Euler method holding Δx fixed (right). As can be seen, an $O(10^{-1})$ change in grid size reduces the error by $O(10^{-2})$, whereas an $O(10^{-1})$ change in time step size reduces the error by $O(10^{-1})$, indicative of an $O(\Delta x^2, \Delta t)$ method. In both cases the error was computed using a 2-norm as compared to the solution found using a very small grid spacing and a small time step*

size or time step were further reduced, indicating that it had converged. The error was computed by comparing to this "smallest grid" solution using an L^2-norm. As can be seen in the plot, the introduction of our predictor-corrector scheme for the nodes has not affected the overall accuracy of the backward Euler method.

We performed the same convergence study for the Crank–Nicolson version of the predictor-corrector scheme, with the results shown in Figure 5.2. In this case we did

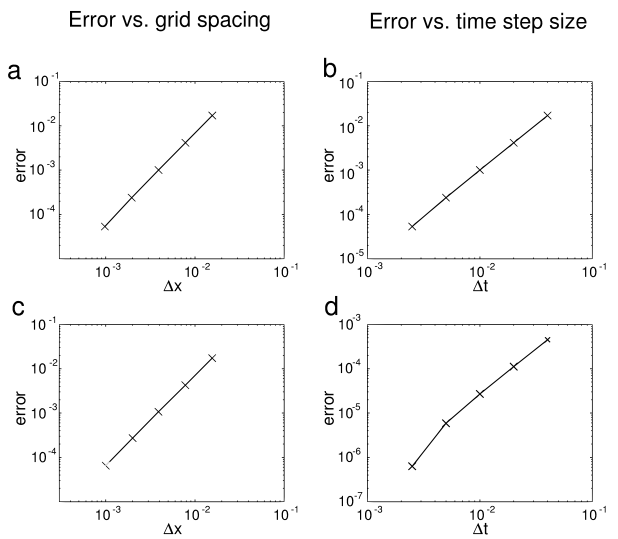


FIG. 5.2. *Second order convergence. Spatial convergence of the numerical scheme presented with* (a) *the ratio $\Delta x/\Delta t$ fixed and* (c) *holding Δt fixed. Temporal convergence of the numerical scheme presented with* (b) *the ratio $\Delta x/\Delta t$ fixed and* (d) *holding Δx fixed. In all cases, an $O(10^{-1})$ change in grid size or time step produces an $O(10^{-2})$ change in error, indicative of an $O(\Delta x^2, \Delta t^2)$ method. In all cases the error was computed using a 2-norm as compared to the solution found using a very small grid spacing and a small time step.*

two separate tests: first we held the ratio of $\Delta x/\Delta t$ fixed and successively reduced the grid spacing. Next, we fixed the time step while reducing the grid spacing. In both cases, when the grid size was cut in half, the error was reduced by a factor of 4, indicating second-order convergence. The two analogous tests were done for the convergence in time, and similar results were found.

To further verify the convergence, the model was run with only passive properties so an exact solution could be computed and used in calculating error. If g_{Na}, g_K, and E_{leak} are 0, then the resulting equation is the passive cable equation

$$C_m \frac{\partial V}{\partial t} = \frac{a}{2R_a} \frac{\partial^2 V}{\partial x^2} - g_{leak} V. \tag{5.1}$$

Rewriting this equation to make the time constant and diffusion coefficient more explicit yields the following:

$$\frac{\partial V}{\partial t} = k \frac{\partial^2 V}{\partial x^2} - \frac{V}{\tau}, \quad \text{where} \quad k = \frac{a}{2C_m R_a} \quad \text{and} \quad \tau = \frac{C_m}{g_{leak}}. \tag{5.2}$$

Given appropriate boundary conditions on a cable of length L and an initial distribution of the voltage, $f(x)$, this equation can be solved explicitly using separation of variables. Assuming that both ends are electrically insulated, the general form of the solution is

$$V(x,t) = A_0 e^{-t/\tau} + \sum_{n=1}^{\infty} A_n cos\left(\frac{n\pi x}{L}\right) e^{-\left(\left(\frac{n\pi}{L}\right)^2 + \frac{1}{k\tau}\right)kt}, \tag{5.3}$$

where

$$A_0 = \frac{1}{L} \int_0^L f(x) dx$$

and

$$A_n = \frac{2}{L} \int_0^L f(x) cos\left(\frac{n\pi x}{L}\right) dx.$$

To simulate the response to a brief current injection, a delta function was used as an initial condition. To construct a delta function located at $x = x_0$, we used the following cosine series:

$$\delta(x) \approx \frac{1}{L} + \frac{2}{L} \sum_{n=1}^{\infty} \cos\left(\frac{n\pi x_0}{L}\right) \cos\left(\frac{n\pi x}{L}\right). \tag{5.4}$$

When this expression is used as the initial condition in (5.3), the result is

$$V(x,t) = \frac{1}{L} e^{-t/\tau} + \frac{2}{L} \sum_{n=1}^{\infty} \cos\left(\frac{n\pi x_0}{L}\right) \cos\left(\frac{n\pi x}{L}\right) e^{-\left(\left(\frac{n\pi}{L}\right)^2 + \frac{1}{k\tau}\right)kt}. \tag{5.5}$$

Computing this sum gives an "exact" solution that can be used to compute the error introduced by our numerical method.

To determine how many terms to use in the Fourier series expansion of the exact solution, the error between finite sums was computed using a 2-norm. The number

of terms used in the series was doubled until it was found that the difference between using 100 terms in the series and 200 was $O(10^{-15})$. At this point it was determined that 100 terms was sufficient. To be consistent with the previous examples in this section, the 100-term summation was used as an "exact" solution for two cables connected by a node. It should be noted, however, that these results are equivalent to simulating a branching structure which follows some geometric constraints. Rall [15], [14] showed that a branching tree structure whose branch points follow the "three-halves" rule for their diameters, i.e., $d_0^{3/2} = d_1^{3/2} + d_2^{3/2}$, can be collapsed into a single equivalent cylinder whose exact solution can be found from (5.5). So the results presented here can be extended to complex branching structures that follow the three-halves rule.

Figure 5.3 shows the convergence results for both numerical methods. In both cases errors are computed using an L^2-norm to measure the relative difference between the computed solution and the exact solution found using the cosine series. The same protocol was used as in Figure 5.1, where the grid size was fixed and the time step was successively reduced and vice versa. In each case, the spatial and temporal convergence rates are the same as for standard backward Euler and Crank–Nicolson.

It should be noted here that we chose parameter values that gave a smaller amount of diffusion so that convergence could be calculated before the δ-function completely diffused away. These values are not realistic for most neural simulations, but the convergence properties demonstrated here are not limited to this choice of parameters.

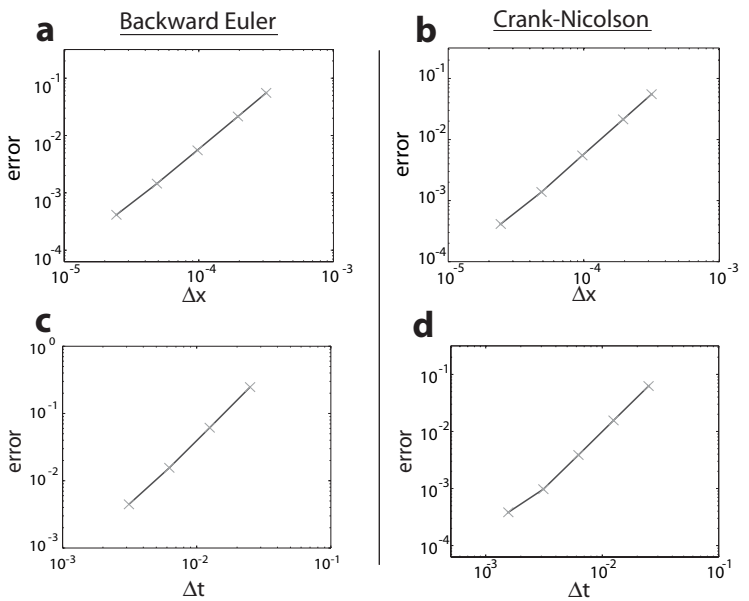


FIG. 5.3. *Convergence of methods as found by comparing to an exact solution. Spatial (*(a) *and* (b)*), and temporal convergence (*(c) *and* (d)*) of both predictor-corrector methods used to simulate a passive cable on a similar two-branch structure as used in Figures* 5.1 *and* 5.2*. In this case, the relative errors were computed using a* 2*-norm as compared to the exact solution,* (5.5)*. Both methods show their usual convergence properties: backward Euler shows* $O(\Delta x^2, \Delta t)$*, while Crank–Nicolson exhibits* $O(\Delta x^2, \Delta t^2)$ *convergence. The parameter values used were* $a = 2\ \mu m$*,* $C_m = 0.1\ mF/cm^2$*,* $R_a = 400\ \Omega cm$*, and* $g_{leak} = 3 \cdot 10^{-4}\ S/cm^2$*.*

6. Stability. Since the new predictor-corrector methods are not truly implicit, we must check that they are stable for the same range of time step size and grid spacing as would be used by the regular backward Euler and Crank–Nicolson implementations. To test the numerical stability of the schemes, we implemented both methods on the same structure used in section 5. Each branch was 80 μm long with a radius of 0.2 μm. A grid spacing of 8 μm was used as this is a conservative choice. Larger choices of grid spacing would only be more stable. An initial time step of 0.025 ms was chosen since that is the default time step size used in NEURON. Figure 6.1 shows the computed error versus the time step size for both predictor-corrector methods. As the time step size was increased, while holding the grid spacing constant, both numerical methods were numerically stable for reasonable choices of time step size, but produced significant errors when the time step became too large. In the case of backward Euler, a time step size of 0.5 ms, double the default used in NEURON, produced stable results, while further increases produced large errors.

As expected, the Crank–Nicolson scheme is stable using a larger time step than the backward Euler method. The case corresponding to the largest time step (0.167 ms) is deemed unacceptable even though the error is still relatively small, since it introduces nonphysical oscillations into the solution. The time step restriction introduced by the method is a very modest one since the time step size that was considered unacceptable is more than six times larger than the default time step used in NEURON (indicated with an arrow in both plots). With a reasonable choice of grid spacing and time step size, the time step restriction introduced by the explicit step is not actually very restrictive for typical neural simulations.

One reason why the time step restriction is not actually very constraining is because the predictor-corrector method is not used in every point of the tree but only at the branch points. In general the number of branch points in a tree is far fewer than the number of points being resolved within the branches using Crank–Nicolson. For instance, for the model cell used in section 7, there were 242 total points being resolved within the branches, while only 11 nodes. In a representative example of an entire neuron, not just the upper apical tuft, the ratio is much more striking: 574 compartments in the entire cell and just 38 nodes. This results in a time step

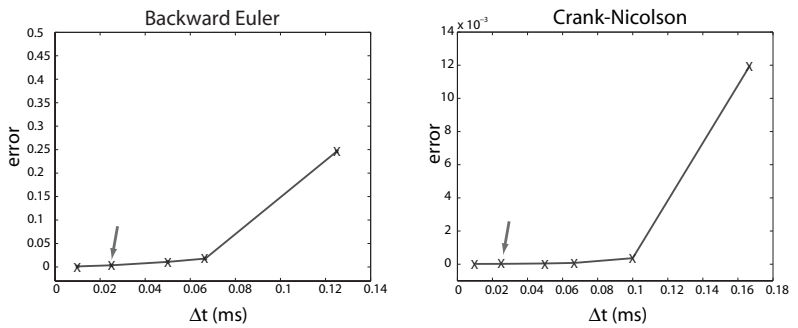


FIG. 6.1. *Both methods are numerically stable for reasonable values of* Δt*. Error versus time step size for the backward Euler version (left) and the Crank–Nicolson version (right) of the predictor-corrector scheme. The values of* Δt *used in both cases were* 0.01 *ms,* 0.025 *ms,* 0.05 *ms,* 0.067 *ms, and* 0.125 *ms. With the largest time step, the backward Euler version of the method produces unacceptable error. The Crank–Nicolson is stable for larger values of* Δt*, but produces unphysical oscillations when* $\Delta t = 0.167$*. The default time step used in NEURON is indicated with an arrow.*

restriction in practice that is not nearly as severe as would result if each of the points in the tree were updated with the predictor-corrector scheme.

7. Accuracy. To test the accuracy of both the backward Euler predictor-corrector and the Crank–Nicolson predictor-corrector methods, we programmed both methods and compared them to their corresponding counterparts which use one large matrix system for the entire tree. The branched structure used was the dendritic tree shown in Figure 7.1(a).

This is the distal portion of the dendritic arbor of a CA1 pyramidal cell from rat hippocampus. In a previous study this reconstruction was used to construct a mathematical model which was compared to experimental results [5]. For the current study Hodgkin–Huxley type channels were added to all of the branches in the tree, making the entire tree excitable. The tree structure was stimulated at its base with a 0.5 nA current for 0.5 ms. This caused an action potential to spread upward throughout the tree. Figure 7.1(b) shows the time course of the membrane voltage as measured part way up the tree (indicated with an arrow in Figure 7.1(a)) using both predictor-corrector numerical methods presented here. Also plotted are the corresponding simulations computed using one large matrix for the entire tree. As can be seen, in both cases the traces made using the predictor-corrector scheme closely match the simulations carried out using one large matrix, indicating that accuracy has not been lost by employing the predictor-corrector schemes.

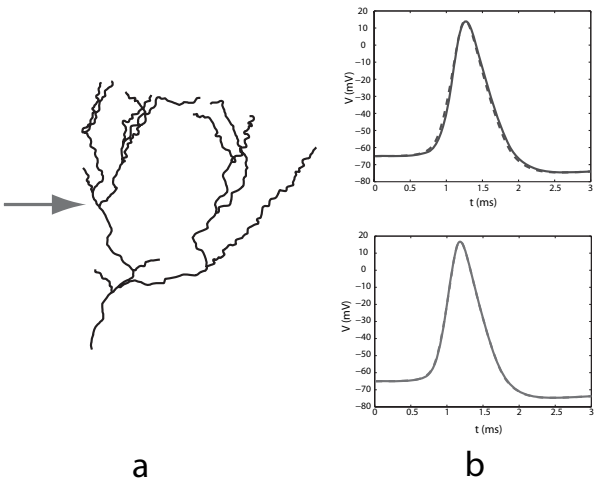


FIG. 7.1. *Accuracy comparison of both predictor-corrector methods. The dendritic tree shown in* (a) *was stimulated at the base of its trunk with a* 0.5 *nA stimulus lasting for* 0.5 *ms. Shown in* (b) *are the time courses of the voltage at the location marked with the arrow in* (a). *The top graph shows the solution computed using the backward Euler version of our predictor-corrector method (solid trace) and using Hines's implementation of backward Euler (dashed trace). The bottom graph is a comparison of the Crank–Nicolson version of our predictor-corrector method (solid trace) and Hines's implementation of Crank–Nicolson (dashed trace). Each trace made using the predictor-corrector method matches the corresponding method using Hines's formulation. The parameters used were* $T = 20°\,C$, $R_i = 150\ \Omega cm$, $C_m = .001\ \mu F/cm$, $\Delta x = 8\ \mu m$, *and* $\Delta t = .015$ *ms.*

8. Spatial adaptivity. One of the main benefits of the numerical methods presented here is that they can be used as the foundation for spatially adaptive numerical methods for neural simulations. Since the computation on each branch and node is done separately, computational effort can be focused where it is needed. We present

here a description of the adaptivity algorithm. To implement the idea of spatial adaptivity, the simulation is started with only a minimum number of branches "active" or computing. All of the other branches have their voltage set to the resting voltage. Those nodes connected to active branches are "listening" for activity to reach them, while the other nodes are set to the resting voltage like the inactive branches. The listening nodes calculate the charge flux coming into them from the active branches using the following expression:

$$\left(\frac{\pi a^2}{R_a}\frac{\partial V}{\partial x}\right). \tag{8.1}$$

This is simply the term in the summation on the left-hand side of (3.9) and is approximated the same way as in (4.1):

$$\frac{\pi a^2}{R_a}\sum_{i=0}^{N}\pm\frac{V_i^t - V_0^t}{\Delta x} \quad \text{for } N-1 \text{ branches connected to this node.} \tag{8.2}$$

If a listening node computes a charge flux that is above a predetermined tolerance, this indicates that there is activity in the corresponding branch. The node then "turns on" all of the other branches connected to it, meaning that the algorithm begins computing the differential equations on those branches. When branches are turned on, the nodes connected to them are sent into the "listening" state and the process repeats itself.

Meanwhile, activity within the branches is monitored to determine when a branch becomes inactive. The branch is considered inactive if the following two criteria are met. The first criterion is that the charge fluxes coming in either end of the branch must be below a predetermined tolerance. These charge fluxes are computed exactly as they were for the node, using (8.2), except now they are thought of as flux into or out of the branch end, instead of the node. The second criteria for turning off a branch is that the voltage everywhere in the branch is within tolerance of the resting voltage. If it is determined that the voltage everywhere in the branch is within tolerance of rest and the charge fluxes coming into or out of the branch ends are small, the branch can then be turned off. When a branch is turned off its voltage is set to the resting potential and is not updated on subsequent time steps. Even though the voltage is within tolerance of the resting voltage when it is turned off and the flux out the ends is small, sometimes prescribing the voltage in this manner can be problematic. In some cases this causes an unphysical charge flux to be introduced into the node connected to the branch that has just been turned off.

To counteract this effect, we have instituted the following rule: if any of the branches connected to a node are turned off, the total flux through that node is set to 0, its value when all of the branches are at the same voltage. In this way, the unnatural flux introduced by setting the branch to the resting voltage is ignored and the charge flux is balanced between the remaining branches. A common situation is when one branch is within tolerance and should be turned off, and the other branches connected to that node are near tolerance, but are not turned off yet because of some remaining activity at the other end. In this case, simply turning off the inactive branch and computing as usual can result in the other branches settling to a voltage that is not the resting voltage. This unnatural effect is negated by setting the total flux through the node to zero. Finally, to complete the spatial adaptivity algorithm, a node is turned off if all the branches connected to it are turned off.

8.1. Example of spatial adaptivity on one cell. To demonstrate the algorithm with spatial adaptivity, we used the cell shown in Figure 1.2. For the purpose of this simulation Hodgkin–Huxley channels were placed throughout the tree making the entire tree excitable. The tree was stimulated at its base with a current injection of 1 nA for 0.5 ms causing an action potential to move upward throughout the tree. Figure 8.1 shows snapshots of the solution at different times. Only those branches that are active are plotted.

Figure 8.2 is an accuracy comparison between the spatially adaptive simulation to the nonadaptive simulation. Shown in the insets are the voltage traces at the corresponding locations pointed to in the tree. As can be seen, in both cases the adaptive trace and the nonadaptive trace overlay, indicating that the spatial adaptivity has not affected the accuracy of the simulation.

To quantify the savings brought about by the spatial adaptivity, we performed many simulations on this cell with and without adaptivity. Due to the variability in timing from one simulation to the next, average computing times were calculated for the algorithm with adaptivity and without it. The results, shown in Figure 8.3, indicate a 30% decrease in computing time for this example. We expect the relative savings to increase as we continue to develop the technique and as the scale of the simulations becomes larger. It is important to note that for this algorithm the computational effort scales with activity, not with the number of cells.

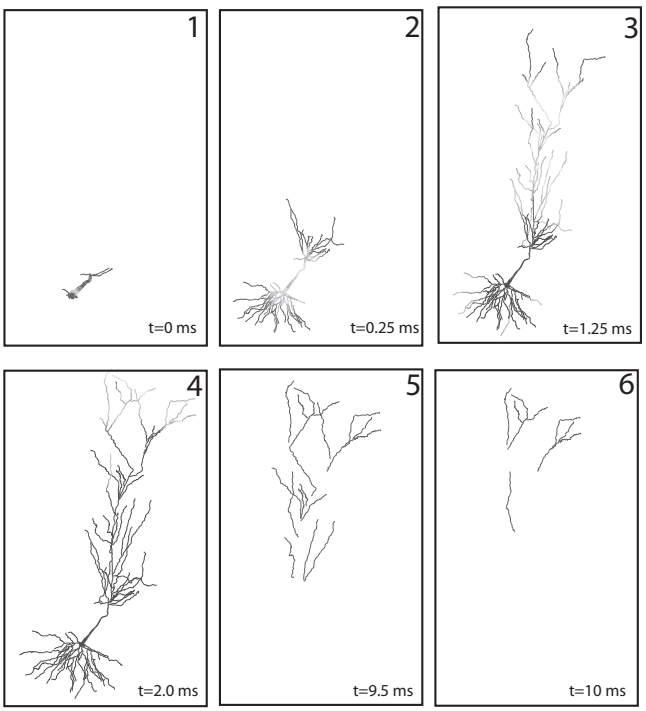


FIG. 8.1. *Snapshots of response to a somatic current injection of* 1 *nA for* 0.5 *ms. Only the branches that are computing are plotted. Shading indicates voltage, with the depolarized regions shaded in light colors and hyperpolarized regions in dark colors. For this simulation an action potential travels outward from the cell body until it leaves the cell entirely. As the cell returns to its resting voltage, branches are turned off.*

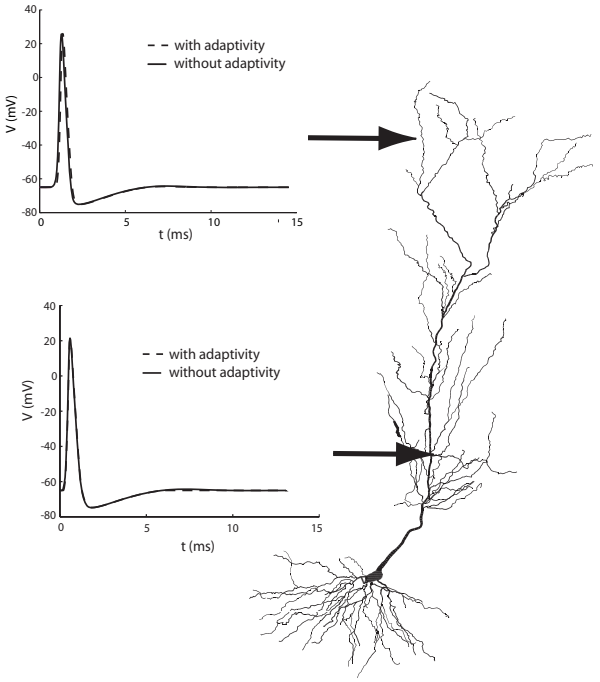


FIG. 8.2. *Comparison of voltage at a specific location on tree. On the right is the shape of the neuron used to compare the methods with and without adaptivity. Shown on the left is the computed voltage versus time at the location marked with the arrow. Since the two traces overlay, the adaptive method is able to accurately resolve the voltage without introducing any additional errors.*

8.2. Example of spatial adaptivity on two cells connected together. As another example, we applied the spatially adaptive algorithm to a simulation of two cells connected by a synapse. The first cell is a pyramidal cell from the CA1 region, while the second cell is the same subicular cell used previously in this section. The CA1 cell was stimulated with a current injection into its cell body, causing an action potential to spread throughout its branches and down its axon, the long branch connecting it to the other cell. Once the action potential reaches the end of the axon, a timer is started. This timer is designed to mimic the delay associated with the

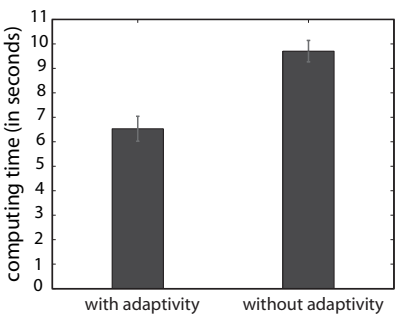


FIG. 8.3. *Comparison of computing times between the methods with and without spatial adaptivity. For this example the spatially adaptive method results in a* 30% *decrease in computing time.*

physiological processes that happen at a synapse before the postsynaptic cell fires. Once the requisite time has passed, a current is injected into the soma of the second cell. This injected current is designed to mimic the time course and size of a typical synaptic current. For the purpose of this example the synapse was made strong enough that it is able to make the second cell fire an action potential that starts in the soma and spreads throughout the cell. Typically, many synapses would be required to make the second cell fire, but the effect is analogous. Snapshots of this simulation are shown in Figure 8.4. Again, only the active branches are plotted. As can be seen, the branches are turned on as needed and turned off again once they inactivate.

To measure the effectiveness of spatial adaptivity for this example, the same test of timing was done as for the previous example, with the results shown in Figure 8.5. In this case, spatial adaptivity brings about a savings of about 50%. Looking closely at the snapshots, this is reasonable since at each snapshot, roughly half of the total branches are active at any one time.

It should be noted that certain simulations are particularly well suited to the spatially adaptive algorithm, in particular, those simulations involving networks of complex cell models where activity is localized within the network. In this case, many of the cells are inactive, allowing our algorithm to only compute on the branches of the few cells that are active, instead of updating each branch of every cell in the network, regardless of activity. Also, sometimes researchers perform many simulations on the same cell model while varying the values of some physical parameters. This

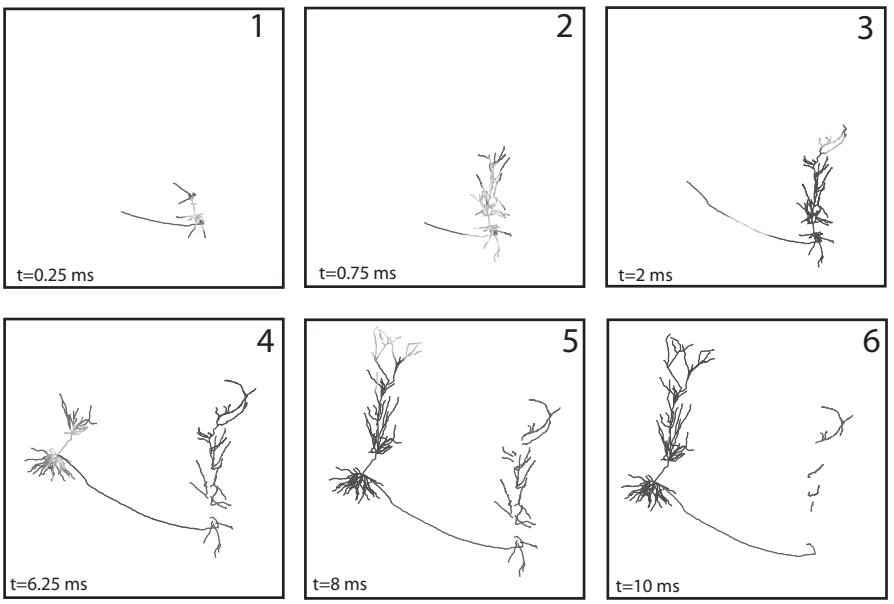


FIG. 8.4. *Snapshots of the response to a current injection of* 1 *nA for* 0.5 *ms into the cell body of a CA*1 *pyramidal cell connected synaptically to a subicular pyramidal cell. The stimulation causes an action potential to form and spread throughout the cell and down its axon. When the activity reaches the end of the axon, a simulated excitatory postsynaptic current was injected into the soma of the subiculum cell after a* 2 *ms delay. This causes the second cell to fire its own action potential. Only the branches that are computing are plotted. Shading indicates voltage, with the depolarized regions shaded in light colors and hyperpolarized regions in dark colors.*

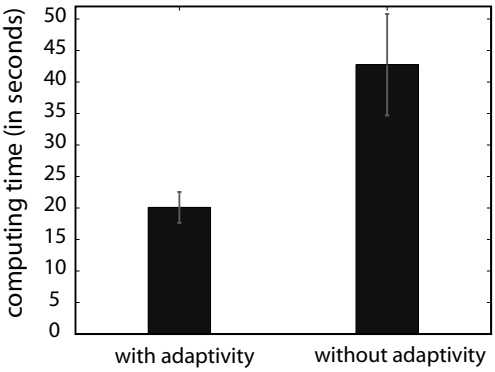


FIG. 8.5. *Comparison of computing times between the methods with and without spatial adaptivity. Using the spatially adaptive method results in a* 50% *decrease in computing time.*

allows them to determine the combination of parameters that yields results matching experimental data. Some combinations of these parameters may not result in an action potential forming and spreading throughout the cell. Conventional methods would require the same amount of effort and time for those simulations with an action potential and those without. Our method, however, would compute on a minimal number of branches and therefore save time and effort on those combinations of parameters that do not result in an action potential.

Furthermore, simulations run out to large times often exhibit large regions of inactivity as the active regions return to rest. Our algorithm would stop computing on those inactive branches, meaning that a simulation like this run out to long times would require only minimal extra effort as compared to a simulation run out to a short time. Conventional methods, on the other hand, scale linearly with simulation time, regardless of activity.

8.3. Example of spatial adaptivity on branches connected together in a straight line. The numerical methods we have presented here allow for spatial adaptivity, but at the cost of introducing extra steps into the update algorithm. In order to quantify the additional computational cost of the predictor-corrector numerical method for the nodes and the relative savings brought about by adaptivity, we compared all 3 methods: the method used by Hines [6], our predictor-corrector method for the nodes without adaptivity, and our predictor-corrector method with adaptivity. For each method we simulated 4 different "super branches" of length 500 μm, 1000 μm, 2000 μm, and 4000 μm, respectively. Each "super branch" was made up of individual branches of length 32 μm connected by nodes. Although these structures are longer than what is commonly seen in real neurons, the total number of branches and nodes is well within reason for typical cells. Figure 8.6 shows the timings of each of the three methods plotted against cable length. In each case the predictor-corrector method without adaptivity is slightly more costly than the corresponding computation done with one large matrix. This is expected due to the additional calculations performed in the predictor-corrector method. Both of these methods scale linearly with the cable length. In contrast, since the adaptive method only computes on the few branches that contain the action potential, its computational cost does not significantly increase as the cable length increases.

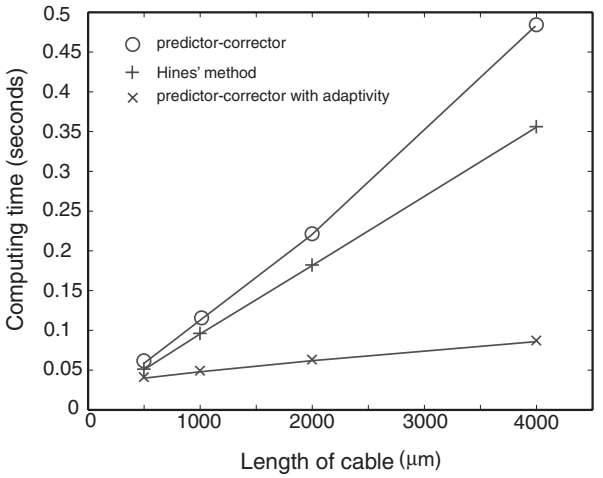


FIG. 8.6. *Timing comparison of* 3 *methods on various length cables. Shown is the computing time in seconds to simulate an action potential on a single cable of various lengths. Each cable is made up of several branches that are each* 32 μm *long and are connected by nodes. The three methods being compared are Hines's method, our predictor-corrector method using Crank–Nicolson for the nodes, and our predictor-corrector method using Crank–Nicolson and adaptivity. Both Hines' method and our method without adaptivity show a linear increase in computing time as the length of cable increases. The predictor-corrector is slightly slower than the method of Hines due to the additional overhead of computing the predictor-corrector steps at each node. Note, however, that since the adaptive method only computes on a few branches that are affected by the action potential, the computational cost is nearly independent of the length of the cable.*

9. Closed loop. Another advantage of the numerical methods presented here is the fact that they make no limitations on the branching nature of the structures being simulated. In particular, there is no restriction on trees connecting back onto themselves. This is in contrast to the method used in [6]. Consider the structure shown in Figure 9.1. This hypothetical structure is one large branch connected back onto itself via a one-way connection representing a synapse (indicated with a circle

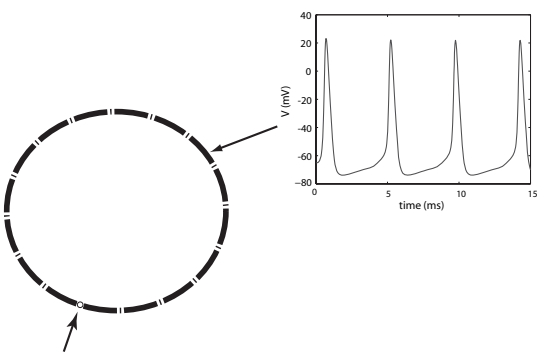


FIG. 9.1. *Loop of branches connected together by nodes. The arrow on the bottom indicates the location of the synapse. The synapse completes the loop and provides only one-way communication in the counterclockwise direction. The branch to the right of the synapse is stimulated, causing an action potential to form and propagate counterclockwise. This action potential continues to move around the circle indefinitely. Shown in the inset is the voltage versus time of one of the branches, indicating the persistent activity.*

in the figure). The large branch is made up of several smaller branches connected by ordinary nodes.

This object produces an output whose time course is similar to that of a real synapse, but whose amplitude has been increased such that it is able to make the connecting branch fire on its own. This synaptic connection is indicated with an arrow in the lower portion of the loop. The output of the synapse causes the branch just to the right to fire an action potential that travels in the counterclockwise direction. The action potential spreads around the circle of branches and triggers the synapse which activates the first branch and starts the action potential once again. The result is sustained propagation of an action potential that travels around the loop of branches indefinitely. Shown in the inset is the voltage versus time in one of the branches, indicating recurrent activity.

Having shown that closed loops of branches can be simulated without difficulty, it is easy to see that the method could naturally be extended to simulate large networks of cells connected by chemical or electrical synapses by simply dividing the network into its respective pieces: branches, nodes, and synapses. Synapses could be modeled in a manner similar to nodes, but instead of firing an action potential like a node, a synapse would have an output that matches that of a real synapse and acts in only one direction. Combining synapses with the spatial adaptivity algorithm would require that synapses perform the same listening task as the nodes but are never allowed to turn off. In this manner, when the synapse begins to have activity, it is always ready to turn on the branches connected to it.

10. Discussion. Two of the most common numerical methods used to solve the equations that describe electrical activity in neurons are Crank–Nicolson and backward Euler. We present a predictor-corrector numerical method to be used with each of these methods that splits a tree into its individual branches. The approaches presented here are advantageous on a number of fronts: (1) They require $O(N)$ operations. (2) They exhibit the same accuracy, respectively, as the methods used within the branches. (3) The voltage update for the tree involves solving a linear system for the voltage on each branch instead of one large linear system for the entire tree. The linear system for each branch is relatively much smaller and is truly tridiagonal, allowing for very efficient solution. (4) These methods do not introduce any dense linear systems to be solved for the nodes. (5) This approach provides a framework for spatially adaptive methods and parallel numerical methods. (6) These methods can handle closed loops.

REFERENCES

[1] J. Bower and D. Beeman, *The Book of Genesis: Exploring Realistic Neural Models with the GEneral NEural SImulation System*, Telos, Santa Clara, CA, 1998.

[2] Y. Cai, E. Walsh, and J. McGee, *A simple program for simulating the responses of neurons with arbitrarily structured and active dendritic trees*, J. Neurosci. Methods, 74 (1997), pp. 27–35.

[3] J. Cooley and F. Dodge, *Digital computer solutions for excitation and propagation of the nerve impulse*, Biophys. J., 6 (1966), pp. 583–599.

[4] J. Crank and P. Nichoson, *A practical method for numerical evaluation of solutions of partial differential equations of the heat-conduction type*, Proc. Cambridge Philos. Soc., 43 (1947), pp. 50–67.

[5] N. Golding, W.L. Kath, and N. Spruston, *Dichotomy of action-potential backpropagation in CA*1 *pyramidal neuron dendrites*, J. Neurophysiol., 86 (2001), pp. 2998–3010.

[6] M. Hines, *Efficient computation of branched nerve equations*, Int. J. Bio-Med. Comput., 15 (1984), pp. 69–76.

[7] M. Hines and N. Carnevale, *The NEURON simulation environment*, Neural Computation, 9 (1997), pp. 1179–1209.
[8] M. Hines and N. Carnevale, *Neuron: A tool for neuroscientists*, The Neuroscientist, 7 (2001), pp. 123–135.
[9] M. Hines and N. Carnevale, *The NEURON simulation environment*, in The Handbook of Brain Theory and Neural Networks, 2nd ed., MIT Press, Cambridge, MA, 2002, pp. 769–773.
[10] A. Hodgkin and A.F. Huxley, *A quantitative description of membrane current and its application to conduction and excitation in nerve*, J. Physiol. London, 117 (1952), pp. 500–544.
[11] M. Mascagni, *The backward Euler method for numerical solution of the Hodgkin–Huxley equations of nerve conduction*, SIAM J. Numer. Anal., 27 (1990), pp. 941–962.
[12] M. Mascagni, *A parallelizing algorithm for computing solutions to arbitrarily branched cable neuron models*, J. Neurosci. Methods, 36 (1991), pp. 105–114.
[13] M. Mascagni and A. Sherman, *Numerical methods for neuronal modeling*, in Methods of Neuronal Modeling: From Ions to Networks, 2nd ed., MIT Press, Cambridge, MA, 1996, pp. 569–606.
[14] W. Rall, *Theory of physiological properties of dendrites*, Ann. N.Y. Acad. Sci., 96 (1962), pp. 1071–1092.
[15] W. Rall, *Theoretical significance of dendritic trees for input-output relations*, in Neural Theory and Modeling, Stanford University Press, Stanford, CA, 1964, pp. 73–97.

SIAM J. SCI. COMPUT.
Vol. 28, No. 6, pp. 2162–2188
© 2006 Society for Industrial and Applied Mathematics

CONSTRUCTING EMBEDDED LATTICE RULES FOR MULTIVARIATE INTEGRATION*

RONALD COOLS†, FRANCES Y. KUO‡, AND DIRK NUYENS†

Abstract. Lattice rules are a family of equal-weight cubature formulae for approximating high-dimensional integrals. By now it is well established that good generating vectors for lattice rules having n points can be constructed component-by-component for integrands belonging to certain weighted function spaces, and that they can achieve the optimal rate of convergence. Although the lattice rules constructed this way are extensible in dimension, they are not extensible in n; thus when n is changed the generating vector needs to be constructed anew. In this paper we introduce a new algorithm for constructing good generating vectors for embedded lattice rules which can be used for a range of n while still being extensible in dimension. By using an adaptation of the fast component-by-component construction algorithm (which makes use of fast Fourier transforms), we are able to obtain good generating vectors for thousands of dimensions and millions of points, under both product weight and order-dependent weight settings, at the cost of $\mathcal{O}(dn(\log(n))^2)$ operations. With a sufficiently large number of points and good overall quality, these embedded lattice rules can be used for practical purposes in the same way as a low-discrepancy sequence. We show for a range of weight settings in an unanchored Sobolev space that our embedded lattice rules achieve the same (optimal) rate of convergence $\mathcal{O}(n^{-1+\delta})$, $\delta > 0$, as those constructed for a fixed number of points, and that the implied constant gains only a factor of 1.30 to 1.55.

Key words. numerical integration, quasi-Monte Carlo methods, rank-1 lattice rules, fast component-by-component construction, embedded lattice rules, extensible lattice sequences

AMS subject classifications. 65D30, 65D32, 68W40

DOI. 10.1137/06065074X

1. Introduction.

Integrals with hundreds or even thousands of dimensions occur in many practical applications in areas such as statistics and mathematical finance. After some appropriate transformations, these integrals can be formulated over the d-dimensional unit cube $[0,1]^d$:

$$I_d(f) = \int_0^1 \cdots \int_0^1 f(x_1, \ldots, x_d)\, \mathrm{d}x_1 \cdots \mathrm{d}x_d = \int_{[0,1]^d} f(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x}\,,$$

where f is a function of d variables and $\boldsymbol{x} = (x_1, \ldots, x_d)$. In this paper we develop an algorithm for constructing "embedded lattice rules" for evaluating such integrals, in the mathematical setting of "weighted reproducing kernel Hilbert spaces." The details about our methods and our function space settings will be discussed in the next section. Here in the introduction we shall give a general overview of the historical background relevant to the new construction.

When d is large, it is too costly to approximate these integrals using product rules based on low-dimensional integration formulae. The [illegible] is an

*Received by the editors January 24, 2006; accepted for publication (in revised form) July 6, 2006; published electronically December 5, 2006. Financial support from the Onderzoeksfonds K.U.Leuven / Research Fund K.U.Leuven and the University of New South Wales Vice-chancellor's Postdoctoral Research Fellowship, as well as the support of the Australian Research Council under its Centres of Excellence program are gratefully acknowledged.

http://www.siam.org/journals/sisc/28-6/65074.html

†Department of Computer Science, K.U.Leuven, Heverlee, Belgium (ronald.cools@cs.kuleuven.be, dirk.nuyens@cs.kuleuven.be).

‡School of Mathematics and Statistics, University of New South Wales, Sydney NSW 2052, Australia (f.kuo@unsw.edu.au).

equal-weight cubature rule of the form

$$Q_{n,d}(f) \;=\; \frac{1}{n}\sum_{k=0}^{n-1} f(\boldsymbol{t}_k)\,,$$

where the n cubature points $\boldsymbol{t}_0,\ldots,\boldsymbol{t}_{n-1}$ are independent and identically distributed uniform random samples from $[0,1]^d$. The expected error for the Monte Carlo method is $\sigma(f)/\sqrt{n}$, where $\sigma^2(f)$ is the variance of f. This leads to a rate of convergence of $\mathcal{O}(n^{-1/2})$ and the convergence is independent of the dimension d provided that $f \in L_2([0,1]^d)$. [illegible] have the same form as the Monte Carlo method, but the cubature points $\boldsymbol{t}_0,\ldots,\boldsymbol{t}_{n-1}$, instead of being generated randomly, are chosen deterministically, in the hope of obtaining a better rate of convergence than the Monte Carlo order $\mathcal{O}(n^{-1/2})$. QMC error bounds are typically in the form of a Koksma–Hlawka-type inequality

$$|I_d(f) - Q_{n,d}(f)| \;\le\; D(\boldsymbol{t}_0,\ldots,\boldsymbol{t}_{n-1})\,V(f)\,, \tag{1.1}$$

where $D(\boldsymbol{t}_0,\ldots,\boldsymbol{t}_{n-1})$ is the [illegible] which measures the nonuniformity of the point set, and $V(f)$ measures the variability of f. Error bounds of this form separate the dependence on the cubature points from the dependence on the integrand. In general we do not have control over the integrand and so we focus on choosing cubature points so that the discrepancy is as small as possible. QMC point sets with discrepancy of order $\mathcal{O}(n^{-1}(\log n)^d)$ or better are collectively known as [illegible] (see, e.g., [12]). This includes, for example, the well-known Sobol′ sequences and Niederreiter sequences. Note that the word "sequences" used in this context implies a special feature: the construction of the point set is independent of the number of points n and the dimension d. Thus these low-discrepancy sequences are "extensible" in the sense that if we need more cubature points or if we need to increase the dimension, we need only to compute the additional points or the additional components of the points.

Lattice rules are a family of QMC rules which were traditionally used to approximate integrals of periodic functions. The oldest and simplest form of lattice rules, now known as [illegible], have the point set

$$\left\{\left\{\frac{k\boldsymbol{z}}{n}\right\} : k = 0,1,\ldots,n-1\right\}.$$

Here $\boldsymbol{z} = (z_1,\ldots,z_d)$, known as the [illegible], is a d-dimensional integer vector having no factor in common with n, and the braces around a vector indicate that each component of the vector is to be replaced by its fractional part. Lattice point sets are low-discrepancy point sets although their constructions are typically not extensible. For a long time the most popular way to construct lattice rules in higher dimensions was to take the generating vector of the [illegible]:

$$\boldsymbol{z} \;=\; (1,a,a^2,\ldots,a^{d-1}) \bmod n\,, \qquad 1 \le a \le n-1,\ \gcd(a,n) = 1\,. \tag{1.2}$$

For a given value of n and a given value of d, an optimal value of a can be chosen by minimizing some discrepancy measure. However, this value of a is only good for the specific pair of n and d, and so if either n or d is changed, we need to obtain a new value of a. (One could use a chosen value of a for higher dimensions simply by taking higher powers of a modulo n, in which case the projections of the points over

consecutive dimensions will remain stationary. Unfortunately, this does not guarantee that the discrepancy measure which we are interested in will remain small.)

In 2002 Sloan and his collaborators (see, e.g., [16]) developed the "component-by-component (CBC)" construction: the components of $\boldsymbol{z}$ are constructed one at a time to minimize the [illegible] in some weighted function space. The details about worst-case analysis and weighted function spaces will be given in the next section. For now we can think of the worst-case error as some form of discrepancy which measures the quality of our lattice rule. Let $e_{n,d}(\boldsymbol{z}) = e_{n,d}(z_1, \ldots, z_d)$ denote the worst-case error for a rank-1 lattice rule with generating vector $\boldsymbol{z}$. The CBC algorithm is as follows.

ALGORITHM 1 (CBC algorithm). [illegible] n

1. [illegible] $z_1 = 1$
2. [illegible] $d = 2, 3, \ldots$
   [illegible] $z_1, \ldots, z_{d-1}$ [illegible]
   [illegible] z_d [illegible] $\{1 \le z \le n-1 : \gcd(z,n) = 1\}$
   [illegible] $e_{n,d}(z_1, \ldots, z_{d-1}, z_d)$

This construction is extensible in d in the sense that if further dimensions are needed, the algorithm can be restarted in an obvious way. It is, however, not extensible in n and so a different value of n means that a completely new $\boldsymbol{z}$ must be constructed.

It has been proved by Hickernell and Niederreiter in 2003 (see [8]) that good generating vectors for extensible lattice sequences do exist; that is, there exists some vector $\boldsymbol{z}$ which is good for all n and all d. However their proof is nonconstructive and it leaves no clues as to how to find such a vector $\boldsymbol{z}$. Since n can be arbitrarily large, the components of the generating vector $\boldsymbol{z}$ can also be arbitrarily large, making it rather difficult to find one generating vector that is good simultaneously for all values of n. Back in 2000, Hickernell and his collaborators (see [7]) had already experimented with the construction of extensible lattice sequences in an empirical setting. By assuming that

$$n = b^m, \quad b \ge 2, \quad m_1 \le m \le m_2, \quad \text{and} \quad 1 \le d \le d_{\max},$$

they designed an algorithm which finds a generating vector of the Korobov form (1.2), with the parameter a chosen by minimizing a loss function

$$G_{m_1,m_2,d_{\max}}(a) := \max_{\substack{m_1 \le m \le m_2 \\ 1 \le d \le d_{\max}}} \frac{D_{b^m,d}(a)}{D^{(\mathrm{asy})}_{b^m,d}}.$$

Here $D_{b^m,d}(a)$ is some discrepancy measure for a d-dimensional b^m-point lattice rule with Korobov parameter a, and $D^{(\mathrm{asy})}_{b^m,d}$ is some asymptotically derived normalization chosen to ensure that for various values of m and d the discrepancy $D_{b^m,d}(a)$ is scaled appropriately for comparison.

Lattice rules constructed in [7] are not so much "extensible" as "embedded" in their nature, the distinction being that extensible rules would leave n unbounded while embedded rules have n bounded. Therefore, with each d-dimensional generating vector $\boldsymbol{z}$, we have a sequence of point sets

$$\{L_m\}_{m_1 \le m \le m_2},$$

where each L_m is the point set for a lattice rule with b^m points, and all the points in L_m are also points of L_{m+1}. For example, when $b = 2$, the number of points doubles

as we move along the sequence, and each rule in the sequence contains all the points from the previous rule together with an equal number of new points. This embedding property is very attractive in practice because not only can we easily increase the number of points without wasting any function evaluations, but also we can estimate the error based on the results from successive rules in the sequence.

Finally we come to the point where we can discuss our new construction. Our algorithm is similar to that of [7], but rather than making use of the Korobov form, our algorithm is based on CBC constructions and so there is no need to assume that d is bounded. For each m from m_1 to m_2, we use Algorithm 1 to find a generating vector $\boldsymbol{z}^{(m)}$ for a b^m-point rule and we store the resulting worst-case error $e_{b^m,d}(\boldsymbol{z}^{(m)})$ for each $d = 1, 2, \ldots$. We define a new error measure for our embedded lattice rule by

$$X_{m_1,m_2,d}(\boldsymbol{z}) := \max_{m_1 \le m \le m_2} \frac{e_{b^m,d}(\boldsymbol{z})}{e_{b^m,d}(\boldsymbol{z}^{(m)})}, \tag{1.3}$$

where we use $e_{b^m,d}(\boldsymbol{z}^{(m)})$ as our reference error to provide a good normalization. Our new algorithm is as follows.

ALGORITHM 2 (CBC algorithm for embedded lattice rules). [illegible] b m_1 m_2 [illegible] $e_{b^m,d}(\boldsymbol{z}^{(m)})$ [illegible] $m_1 \le m \le m_2$ [illegible] $d = 1, 2, \ldots$

1. [illegible] $z_1 = 1$
2. [illegible] $d = 2, 3, \ldots$
   [illegible] $z_1, \ldots, z_{d-1}$ [illegible]
   [illegible] z_d [illegible] $\{1 \le z \le b^{m_2} - 1 : \gcd(z, b) = 1\}$
   [illegible] $X_{m_1,m_2,d}(z_1, \ldots, z_{d-1}, z_d)$

Just as is the case for the algorithm in [7], our new algorithm does not come with an a priori error bound, nor does it give us a fully extensible lattice sequence that can be used for arbitrarily large n. Having said that, in addition to the bonus that d need not be bounded, our algorithm does have two advantages over the algorithm of [7]. First, it is proved in [10] and in [4] that the CBC algorithm can achieve a rate of convergence of $\mathcal{O}(n^{-1+\delta})$, $\delta > 0$, if we "randomly shift" the lattice points (see below). Thus the worst-case error $e_{b^m,d}(\boldsymbol{z}^{(m)})$ can be said to be almost optimal, which means that we are using a fairly reliable normalization. The quantity $X_{m_1,m_2,d}(\boldsymbol{z})$ represents how much worse our chosen vector $\boldsymbol{z}$ is compared with the vectors $\boldsymbol{z}^{(m)}$ which are the near-optimal choices for different $n = b^m$. With all the benefits of having embedded rules, we would be happy with a value of $X_{m_1,m_2,d}(\boldsymbol{z})$ that is no more than 2, and this indeed turns out to be the case in all our computations. Thus in practice our construction comes with an a posteriori worst-case error bound that is of optimal order, and within a factor of 2 of that achieved by the original CBC algorithm.

The second advantage of our algorithm is that, surprisingly, it has a relatively low computational cost. Nuyens and Cools [13, 14] developed a fast implementation of the CBC algorithm by making use of fast Fourier transforms which allows the construction with n in the millions and d in the thousands within a feasible time frame. Thus the precomputation of $\boldsymbol{z}^{(m)}$ for each m can be done in a fast way. In fact if we take b to be prime, our new algorithm based on minimizing $X_{m_1,m_2,d}(\boldsymbol{z})$ can also be done in a fast way, and the cost depends only on m_2 rather than the sum of all m. The total computational cost of Algorithm 2, including the precomputations of Algorithm 1, is $\mathcal{O}(dn(\log(n))^2)$ operations.

The description above was a bit simplified: it is actually "randomly shifted" rank-1 lattice rules that we construct generating vectors for. [illegible] 1 [illegible]

are QMC rules with the point set

$$\left\{ \left\{ \frac{k\boldsymbol{z}}{n} + \boldsymbol{\Delta} \right\} : k = 0, 1, \ldots, n-1 \right\},$$

where $\boldsymbol{z}$ is the generating vector as in a rank-1 lattice rule, and $\boldsymbol{\Delta} \in [0,1]^d$ is the [illegible]. In particular, we are interested in using [illegible] shifts. There are several advantages in doing this: we need only to determine the generating vector; the error expression is simpler (see section 2); and most importantly, we can obtain a probabilistic error estimate (see section 5).

The paper is organized as follows. In section 2 we discuss weighted reproducing kernel Hilbert spaces. We consider weighted Korobov spaces and weighted Sobolev spaces, as it is already known that the CBC algorithm leads to lattice rules which achieve the optimal rate of convergence in these spaces, under suitable conditions on the weights. In section 3 we outline some important properties of embedded lattice rules. In section 4 we discuss the implementation of our new algorithm under two different weight settings: the product weights and the order-dependent weights. We consider lattice rules with $n = b^m$ points, and for simplicity we consider only prime base b. In section 5 we provide some notes on how to use these embedded lattice rules as a sequence and how to obtain practical error estimation by using random shifts. In section 6 we carry out some numerical experiments to see how well our new lattice rules perform for some finance problems. Finally a brief summary is given in section 7. In the appendix we list one generating vector up to 360 dimensions (obtained with $b = 2$, $m_1 = 10$, and $m_2 = 20$) which is good for n ranging from 1024 to 1048576.

2. Function space setting. In this section we briefly review the mechanics behind worst-case analysis in a reproducing kernel Hilbert space and the concept of weighted function spaces. For more details, see, e.g., [6, 17, 5].

2.1. Worst-case analysis in reproducing kernel Hilbert spaces. Suppose that our integrands belong to some reproducing kernel Hilbert space H. A reproducing kernel $K(\boldsymbol{x}, \boldsymbol{y})$ is a symmetric and positive definite function satisfying $K(\boldsymbol{x}, \cdot) \in H$ for all $\boldsymbol{x} \in [0,1]^d$, and $\langle K(\boldsymbol{x}, \cdot), f \rangle_H = f(\boldsymbol{x})$ for all $f \in H$ and all $\boldsymbol{x} \in [0,1]^d$. The existence of a reproducing kernel follows from the Riesz representation theorem, under the assumption that point evaluations are bounded linear functionals in H. We assume that the function $h(\boldsymbol{x}) := \int_{[0,1]^d} K(\boldsymbol{x}, \boldsymbol{y}) \, \mathrm{d}\boldsymbol{y}$ belongs to the space H. Then multivariate integration I_d is a well-defined continuous linear functional, with h as its representer in H, and the [illegible] for multivariate integration is

$$\|I_d\| = \|h\|_H = \left(\int_{[0,1]^{2d}} K(\boldsymbol{x}, \boldsymbol{y}) \, \mathrm{d}\boldsymbol{x} \, \mathrm{d}\boldsymbol{y} \right)^{1/2}.$$

To ensure that H is embedded in $L_2([0,1]^d)$, we assume that $\int_{[0,1]^d} K(\boldsymbol{x}, \boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} < \infty$ which also guarantees that the initial error is finite.

The [illegible] for a QMC rule $Q_{n,d}$ with point set $P_n = \{\boldsymbol{t}_0, \ldots, \boldsymbol{t}_{n-1}\}$ in H is defined by

$$e_{n,d}(P_n, K) := \sup_{f \in H, \, \|f\|_H \le 1} |I_d(f) - Q_{n,d}(f)|.$$

Clearly $|I_d(f) - Q_{n,d}(f)| \le e_{n,d}(P_n, K) \|f\|_H$. Thus the worst-case error can be

thought of as a type of discrepancy measure (see (1.1)). Using the reproducing property of K one can derive an expression for the squared worst-case error

(2.1)
$$e^2_{n,d}(P_n, K) = \int_{[0,1]^{2d}} K(\boldsymbol{x},\boldsymbol{y})\,\mathrm{d}\boldsymbol{x}\,\mathrm{d}\boldsymbol{y} - \frac{2}{n}\sum_{k=0}^{n-1}\int_{[0,1]^d} K(\boldsymbol{t}_k,\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} + \frac{1}{n^2}\sum_{k=0}^{n-1}\sum_{\ell=0}^{n-1} K(\boldsymbol{t}_k,\boldsymbol{t}_\ell).$$

This explicit expression is especially useful when the reproducing kernel has a simple closed form.

The error expression is simpler for [illegible] kernels. A shift-invariant kernel is one which satisfies

$$K(\{\boldsymbol{x}+\boldsymbol{\Delta}\},\{\boldsymbol{y}+\boldsymbol{\Delta}\}) = K(\boldsymbol{x},\boldsymbol{y}) = K(\{\boldsymbol{x}-\boldsymbol{y}\},\boldsymbol{0}) \qquad \forall \boldsymbol{x},\boldsymbol{y},\boldsymbol{\Delta}\in[0,1]^d\,.$$

For any arbitrary kernel K, there is an associated shift-invariant kernel K^{sh} defined by

$$K^{\mathrm{sh}}(\boldsymbol{x},\boldsymbol{y}) := \int_{[0,1]^d} K(\{\boldsymbol{x}+\boldsymbol{\Delta}\},\{\boldsymbol{y}+\boldsymbol{\Delta}\})\,\mathrm{d}\boldsymbol{\Delta}\,.$$

Let $P_n+\boldsymbol{\Delta} := \{\{\boldsymbol{t}+\boldsymbol{\Delta}\} : \boldsymbol{t}\in P_n\}$ denote the point set obtained by adding a shift $\boldsymbol{\Delta}$ to every point in P_n and then taking the fractional part of every component. One can easily derive the important property that

$$\begin{aligned} e^2_{n,d}(P_n, K^{\mathrm{sh}}) &= \int_{[0,1]^d} e^2_{n,d}(P_n+\boldsymbol{\Delta}, K)\,\mathrm{d}\boldsymbol{\Delta} \\ &= -\int_{[0,1]^d} K^{\mathrm{sh}}(\boldsymbol{x},\boldsymbol{0})\,\mathrm{d}\boldsymbol{x} + \frac{1}{n^2}\sum_{k=0}^{n-1}\sum_{\ell=0}^{n-1} K^{\mathrm{sh}}\left(\{\boldsymbol{t}_k-\boldsymbol{t}_\ell\},\boldsymbol{0}\right). \end{aligned} \tag{2.2}$$

We want to use randomly shifted rank-1 lattice rules in the function space with kernel K. It makes sense to study this "shift-averaged" error given by (2.2). In addition, the double sum reduces to a single sum for lattice rules; explicitly we have

$$e^2_{n,d}(\boldsymbol{z}) = -\int_{[0,1]^d} K^{\mathrm{sh}}(\boldsymbol{x},\boldsymbol{0})\,\mathrm{d}\boldsymbol{x} + \frac{1}{n}\sum_{k=0}^{n-1} K^{\mathrm{sh}}\left(\left\{\tfrac{k\boldsymbol{z}}{n}\right\},\boldsymbol{0}\right). \tag{2.3}$$

We shall refer to this quantity as [illegible] 1 [illegible], and it is this worst-case error which we shall consider throughout the rest of the paper. Note that $K^{\mathrm{sh}} = K$ if K is already shift-invariant.

2.2. Weighted function spaces. In this paper we consider reproducing kernel Hilbert spaces H with kernels of the form

$$K(\boldsymbol{x},\boldsymbol{y}) = \sum_{\mathfrak{u}\subseteq\{1,\ldots,d\}} \gamma_{\mathfrak{u}} \prod_{j\in\mathfrak{u}} \eta(x_j,y_j)\,,$$

with $\eta(x,y)=\eta(y,x)$. This general form of kernels has been considered in [17, 5]. The space H is a direct sum of 2^d reproducing kernel Hilbert spaces $H_{\mathfrak{u}}$, where each $H_{\mathfrak{u}}$ is a $|\mathfrak{u}|$-dimensional tensor-product space with kernel $K_{\mathfrak{u}}(\boldsymbol{x}_{\mathfrak{u}},\boldsymbol{y}_{\mathfrak{u}}) = \gamma_{\mathfrak{u}}\prod_{j\in\mathfrak{u}}\eta(x_j,y_j)$.

Functions in $H_{\mathfrak{u}}$ depend only on the set of variables $\boldsymbol{x}_{\mathfrak{u}} := \{x_j : j \in \mathfrak{u}\}$. Functions $f \in H$ have the natural decomposition

$$f(\boldsymbol{x}) = \sum_{\mathfrak{u} \subseteq \{1,\ldots,d\}} f_{\mathfrak{u}}(\boldsymbol{x}_{\mathfrak{u}}), \quad \text{with} \quad f_{\mathfrak{u}}(\boldsymbol{x}_{\mathfrak{u}}) = \langle K_{\mathfrak{u}}(\boldsymbol{x}_u, \cdot), f \rangle_H \quad \text{and} \quad f_{\mathfrak{u}} \in H_{\mathfrak{u}}.$$

We associate a [illegible] parameter $\gamma_{\mathfrak{u}}$ with each set of variables $\boldsymbol{x}_{\mathfrak{u}}$, which describes the level of [illegible] between the variables in $\boldsymbol{x}_{\mathfrak{u}}$. The 2^d weights $\gamma_{\mathfrak{u}}$ together model the relative importance between various sets of variables. Roughly speaking, a small $\gamma_{\mathfrak{u}}$ means that the contribution of the term $f_{\mathfrak{u}}$ is small compared with the other terms. To fix the scaling we set $\gamma_\emptyset = 1$. The limiting case of $\gamma_{\mathfrak{u}} = 0$ implies that there is no $f_{\mathfrak{u}}$ term in the decomposition, or in other words, there is no interaction between the variables in $\boldsymbol{x}_{\mathfrak{u}}$.

The worst-case error for randomly shifted rank-1 lattice rules (see (2.3)) is expressed in terms of the associated shift-invariant kernel

$$K^{\mathrm{sh}}(\boldsymbol{x}, \boldsymbol{y}) = \sum_{\mathfrak{u} \subseteq \{1,\ldots,d\}} \gamma_{\mathfrak{u}} \prod_{j \in \mathfrak{u}} \omega(\{x_j - y_j\}), \quad \text{with} \quad \omega(x) = \int_0^1 \eta(\{x + \Delta\}, \Delta) \, \mathrm{d}\Delta.$$

Note that ω is symmetric, i.e., $\omega(x) = \omega(1-x)$. For simplicity we impose the additional condition $\int_0^1 \omega(x) \, \mathrm{d}x = 0$. The initial error is then 1 and the worst-case error (2.3) then satisfies

$$e^2_{n,d}(\boldsymbol{z}) = -1 + \frac{1}{n} \sum_{k=0}^{n-1} \sum_{\mathfrak{u} \subseteq \{1,\ldots,d\}} \gamma_{\mathfrak{u}} \prod_{j \in \mathfrak{u}} \omega\left(\left\{\tfrac{k z_j}{n}\right\}\right). \tag{2.4}$$

(If the initial error were not 1, we could consider instead the relative error; that is, we divide $e^2_{n,d}(\boldsymbol{z})$ by the squared initial error. The expression for the relative error would be of the same form as (2.4), but with rescaled weights $\gamma_{\mathfrak{u}}$.)

To have a function space that best describes our integrand f, we must choose the weights $\gamma_{\mathfrak{u}}$ according to the dimension structure of f. (See [3] for two notions of [illegible], and see [19] for numerical schemes to estimate them.) When the nominal dimension is d, there are 2^d weights $\gamma_{\mathfrak{u}}$ to decide upon. Here are a few options to help limit the choices.

(i) [illegible]. Here $\gamma_{\mathfrak{u}} := \prod_{j \in \mathfrak{u}} \gamma_{\{j\}}$; that is, the weight associated with the set of variables $\boldsymbol{x}_{\mathfrak{u}}$ is automatically assigned the product of the weights for each individual variable x_j in this set. The product-weight setting leads to a tensor-product space H, and the worst-case error for randomly shifted rank-1 lattice rules satisfies

$$e^2_{n,d}(\boldsymbol{z}) = -1 + \frac{1}{n} \sum_{k=0}^{n-1} \prod_{j=1}^{d} \left(1 + \gamma_j \, \omega\left(\left\{\tfrac{k z_j}{n}\right\}\right)\right).$$

(ii) [illegible]. Here $\gamma_{\mathfrak{u}} := \Gamma_{|\mathfrak{u}|}$; that is, the weight associated with the set of variables $\boldsymbol{x}_{\mathfrak{u}}$ depends only on the cardinality of $\mathfrak{u}$. In this case the worst-case error for randomly shifted rank-1 lattice rules can be written as

$$e^2_{n,d}(\boldsymbol{z}) = \frac{1}{n} \sum_{k=0}^{n-1} \sum_{\ell=1}^{d} \Gamma_\ell \sum_{\substack{\mathfrak{u} \subseteq \{1,\ldots,d\} \\ |\mathfrak{u}| = \ell}} \prod_{j \in \mathfrak{u}} \omega\left(\left\{\tfrac{k z_j}{n}\right\}\right).$$

(iii) [illegible]. Here $\gamma_{\mathfrak{u}} = 0$ for all $|\mathfrak{u}| > q$, with some fixed number q. The finite-order-weight setting can be very useful because many high-dimensional integrals in practice do appear to have a finite order of just 2 or 3. We often combine the finite-order-weight setting with the order-dependent-weight setting.

We end this section with two important examples of such spaces.

[illegible] 1 (weighted Korobov spaces). The weighted Korobov space consists of one-periodic functions with absolutely convergent Fourier series; see, e.g., [5]. The reproducing kernel is already shift-invariant, with

$$\eta(x,y) \,=\, \omega(\{x-y\}) \quad \text{and} \quad \omega(x) \,=\, \sum_{h\in\mathbb{Z}\setminus\{0\}} \frac{e^{2\pi \mathrm{i} h x}}{|h|^{\alpha}}\,,$$

where $\alpha > 1$ is a smoothness parameter which corresponds to the rate of decay of the Fourier coefficients. It is known that good (unshifted or randomly shifted) rank-1 lattice rules can be constructed using the CBC algorithm to achieve the optimal rate of convergence $\mathcal{O}(n^{-\alpha/2+\delta})$, $\delta > 0$ (see [10, 4]). Note that when α is an even integer, ω can be written in terms of the Bernoulli polynomial of degree α:

$$\omega(x) \,=\, \frac{(2\pi)^{\alpha}}{(-1)^{\frac{\alpha}{2}-1}\alpha!} B_{\alpha}(x)\,.$$

[illegible] 2 (weighted Sobolev spaces). The weighted Sobolev space (with smoothness one) consists of functions with square-integrable mixed first derivatives; see, e.g., [17]. These functions are in general nonperiodic. There are a few variants of these spaces depending on how the inner product is defined. One popular variant, now commonly referred to as the [illegible] Sobolev space, corresponds to

$$\eta(x,y) \,=\, \tfrac{1}{2} B_2(\{x-y\}) + \left(x-\tfrac{1}{2}\right)\left(y-\tfrac{1}{2}\right) \quad \text{and} \quad \omega(x) \,=\, B_2(x) \,=\, x^2 - x + \tfrac{1}{6}\,,$$

which satisfies the condition $\int_0^1 \omega(x)\,\mathrm{d}x = 0$. This space has a direct association with the ANOVA decomposition of functions (see, e.g., [3, 19]). It is known that good randomly shifted rank-1 lattice rules can be constructed using the CBC algorithm to achieve the optimal rate of convergence $\mathcal{O}(n^{-1+\delta})$, $\delta > 0$, in a probabilistic sense (see [10, 4]).

3. Properties of embedded lattice rules. Before we get into the ideas behind the fast implementations of CBC algorithms, we need to review some fundamental properties of embedded lattice rules.

Given an integer base $b \geq 2$, a fixed dimension d, and any integer vector $\boldsymbol{z} \in \mathbb{Z}^d$ whose components are all coprime with b, define

$$L_m \,=\, L_m(\boldsymbol{z}) \,:=\, \left\{ \left\{ \frac{k\boldsymbol{z}}{b^m} \right\} : k = 0, 1, \ldots, b^m - 1 \right\}, \quad m = 0, 1, \ldots\,. \tag{3.1}$$

The following properties hold for all $m \geq 0$.

[illegible] 1. Clearly L_m is a point set for a rank-1 lattice rule with b^m distinct points.

[illegible] 2. The generating vector for the lattice point set L_m can be reduced modulo b^m to give the same point set; i.e., $L_m(\boldsymbol{z}) = L_m(\boldsymbol{z} \bmod b^m)$.

[illegible] 3. The lattice point set L_m is embedded in L_{m+1}; i.e., $L_m \subset L_{m+1}$.

[illegible] 4. The points in L_m are exactly those points in L_{m+1} whose indices k in (3.1) are multiples of b.

For a given m, we can construct a generating vector $\boldsymbol{z}$ using Algorithm 1 with $n = b^m$ so that L_m is a "good" lattice point set with b^m points. Due to Property 2, we can restrict our search for the components of $\boldsymbol{z}$ to a finite set as indicated in Algorithm 1, with the largest possible choice being $b^m - 1$. Note that the goodness of the point set L_m in no way implies that all smaller lattice point sets embedded in L_m are good. On the contrary, it is quite possible that the same generating vector $\boldsymbol{z}$ leads to a very poor point set L_ℓ for $\ell < m$ (as will be demonstrated in section 6.1).

This brings us to Algorithm 2, where a generating vector $\boldsymbol{z}$ is constructed with the hope that the lattice point sets L_m, for all values of m in a finite range $m_1 \le m \le m_2$, would be good simultaneously. The assumption that m is bounded from above by m_2 means that the maximum number of points is b^{m_2}; hence we can restrict the components of $\boldsymbol{z}$ to be at most $b^{m_2} - 1$, ensuring a finite search space. The quality of this generating vector $\boldsymbol{z}$ is assessed by the error measure $X_{m_1,m_2,d}(\boldsymbol{z})$ defined in (1.3), which compares the worst-case error of $\boldsymbol{z}$ with the worst-case error of the near-optimal generating vectors constructed by Algorithm 1 for each m in our range. Put differently, $X_{m_1,m_2,d}(\boldsymbol{z})$ measures how much worse than optimal our "one for all" vector $\boldsymbol{z}$ is, and Algorithm 2 selects a vector $\boldsymbol{z}$ so that $X_{m_1,m_2,d}(\boldsymbol{z})$ is as small as possible for m in the prescribed range. This can be described as the "mini-max" optimization of the worst-case error ratios. The quality of $\boldsymbol{z}$ as a "one for all" vector is ensured as long as $X_{m_1,m_2,d}(\boldsymbol{z})$ is uniformly bounded by a small constant, say, 2.

4. Fast construction of embedded lattice rules. Before we can use Algorithm 2, we need to use Algorithm 1 to find the near-optimal generating vectors for each m from m_1 to m_2 and store the worst-case errors in each dimension. The apparent cost of Algorithm 1 for n points up to dimension d is $\mathcal{O}(d^2n^2)$, and so the cost of the precomputations for Algorithm 2 up to $n = b^{m_2}$ points can be $\mathcal{O}(d^2n^2\log(n))$. Once this is done, the cost of Algorithm 2 appears to be another $\mathcal{O}(d^2n^2\log(n))$. Thus unless we can speed up the computations in some way, Algorithm 2 would be a mere theoretical exercise.

Three ideas are used to obtain a fast alternative for Algorithm 2 (the first two were already known for Algorithm 1).

(i) The nature of the CBC algorithm is iterative with respect to the dimension. Therefore we will store previous calculations, sacrificing memory for speed. This reduces the construction cost of Algorithm 1 to $\mathcal{O}(dn^2)$.

(ii) The calculation of the worst-case errors resembles a structured matrix-vector product which can be computed in a fast way. Thus the construction cost of Algorithm 1 can be dramatically reduced, to $\mathcal{O}(dn\log(n))$, implying that the cost of the precomputations for Algorithm 2 is $\mathcal{O}(dn(\log(n))^2)$.

(iii) The calculation of the worst-case error $e_{n,d}(\boldsymbol{z})$ for the maximum number of points $n = b^{m_2}$ contains within it all worst-case errors $e_{b^m,d}(\boldsymbol{z})$ for $m \le m_2$. This means that, with a slight adaptation of the algorithm, the actual cost of Algorithm 2 is only $\mathcal{O}(dn\log(n))$ while the cost for those precomputations based on Algorithm 1 is $\mathcal{O}(dn(\log(n))^2)$. In other words, we get the embedded rule almost for free.

In the next three subsections we will explain these ideas as independently of each other as possible.

4.1. Matrix-vector formulation. Suppose we are at the point in the CBC algorithm where we want to choose the dth component z_d. It makes sense to consider the squared worst-case error $e^2_{n,d}(\boldsymbol{z}) = e^2_{n,d}(z_1,\dots,z_d)$ as a function of z_d, and we denote it by $E^2_d(z_d)$. First we will rewrite the expression for $E^2_d(z_d)$ in a form that is

more suitable for recursion with respect to d. For computational efficiency we consider only two types of weights.

(i) [illegible]:

$$E_d^2(z_d) \;=\; -1 + \frac{1}{n}\sum_{k=0}^{n-1}\left(1+\gamma_d\,\omega\left(\left\{\frac{kz_d}{n}\right\}\right)\right)\underbrace{\prod_{j=1}^{d-1}\left(1+\gamma_j\,\omega\left(\left\{\frac{kz_j}{n}\right\}\right)\right)}_{p_{d-1}(k)}$$

$$= e_{n,d-1}^2(z_1,\ldots,z_{d-1}) + \frac{\gamma_d}{n}\sum_{k=0}^{n-1}\omega\left(\left\{\frac{kz_d}{n}\right\}\right)p_{d-1}(k)\,, \tag{4.1}$$

where $e_{n,0}^2 := 0$ and the products $p_d(k)$ are defined recursively by

$$\begin{aligned} p_0(k) &:= 1\,,\\ p_d(k) &:= \left(1+\gamma_d\,\omega\left(\left\{\tfrac{kz_d}{n}\right\}\right)\right)p_{d-1}(k)\,. \end{aligned} \tag{4.2}$$

(ii) [illegible] (q):

$$E_d^2(z_d) \;=\; \frac{1}{n}\sum_{k=0}^{n-1}\sum_{\ell=1}^{\min(q,d)}\Gamma_\ell\left(\underbrace{\sum_{\substack{\mathfrak{u}\subseteq\{1,\ldots,d-1\}\\|\mathfrak{u}|=\ell}}\prod_{j\in\mathfrak{u}}\omega\left(\left\{\frac{kz_j}{n}\right\}\right)}_{p_{d-1,\ell}(k)} + \omega\left(\left\{\frac{kz_d}{n}\right\}\right)\underbrace{\sum_{\substack{\mathfrak{u}\subseteq\{1,\ldots,d-1\}\\|\mathfrak{u}|=\ell-1}}\prod_{j\in\mathfrak{u}}\omega\left(\left\{\frac{kz_j}{n}\right\}\right)}_{p_{d-1,\ell-1}(k)}\right)$$

$$= e_{n,d-1}^2(z_1,\ldots,z_{d-1}) + \frac{1}{n}\sum_{k=0}^{n-1}\omega\left(\left\{\frac{kz_d}{n}\right\}\right)\left(\sum_{\ell=1}^{\min(q,d)}\Gamma_\ell\,p_{d-1,\ell-1}(k)\right), \tag{4.3}$$

where $e_{n,0}^2 := 0$ and the products $p_{d,\ell}(k)$ are defined recursively by

$$\begin{aligned} p_{d,0}(k) &:= 1\,,\\ p_{d,\ell}(k) &:= p_{d-1,\ell}(k) + \omega\left(\left\{\tfrac{kz_d}{n}\right\}\right)p_{d-1,\ell-1}(k)\,. \end{aligned} \tag{4.4}$$

Let $\mathbb{Z}_n := \{0,1,\ldots,n-1\}$ denote the set of the integers modulo n, and U_n the multiplicative group of integers modulo n, i.e.,

$$U_n \;:=\; \{u\in\mathbb{Z}_n : \gcd(u,n)=1\}\,,$$

with the cardinality of U_n given by the Euler totient function $\phi(n)=|U_n|$. We need to evaluate $E_d^2(z_d)$ for every choice of $z_d\in U_n$, which suggests the definition of the vectors

$$\boldsymbol{E}_d^2 := \left[E_d^2(z)\right]_{z\in U_n}\,,\quad \boldsymbol{p}_d := \left[p_d(k)\right]_{k\in\mathbb{Z}_n}\,,\quad \boldsymbol{p}_{d,\ell} := \left[p_{d,\ell}(k)\right]_{k\in\mathbb{Z}_n} \tag{4.5}$$

and the matrix

$$\boldsymbol{\Omega}_n \;:=\; \left[\omega\left(\left\{\frac{kz}{n}\right\}\right)\right]_{\substack{z\in U_n\\ k\in\mathbb{Z}_n}} \;=\; \left[\omega\left(\frac{kz \bmod n}{n}\right)\right]_{\substack{z\in U_n\\ k\in\mathbb{Z}_n}}. \tag{4.6}$$

We now observe from (4.1) and (4.3) that the squared worst-case error vector $\boldsymbol{E}_d^2$ can be expressed in terms of a matrix-vector product with the matrix $\boldsymbol{\Omega}_n$ as

$$\boldsymbol{E}_d^2 \;:=\; \mathbf{1}_{\phi(n)}\, e_{n,d-1}^2(z_1,\ldots,z_{d-1}) \;+\; \frac{\gamma_d}{n}\,\boldsymbol{\Omega}_n\,\boldsymbol{p}_{d-1} \qquad \text{(product)},$$

$$\boldsymbol{E}_d^2 \;:=\; \mathbf{1}_{\phi(n)}\, e_{n,d-1}^2(z_1,\ldots,z_{d-1}) \;+\; \frac{1}{n}\,\boldsymbol{\Omega}_n \left(\sum_{\ell=1}^{\min(q,d)} \Gamma_\ell\,\boldsymbol{p}_{d-1,\ell-1}\right) \qquad \text{(order-dependent)},$$

where $\mathbf{1}_t$ denotes a vector of ones of length t. Algorithm 1 picks the value of $z_d \in U_n$ which corresponds to the smallest entry in $\boldsymbol{E}_d^2$. Then it is clear from (4.2) and (4.4) that the vectors $\boldsymbol{p}_d$ or $\boldsymbol{p}_{d,\ell}$ for the next iteration can be obtained recursively via

$$\boldsymbol{p}_d \;:=\; (\mathbf{1}_n + \gamma_d\,\boldsymbol{\Omega}_n(z_d)) \mathbin{.*} \boldsymbol{p}_{d-1} \qquad \text{(product)}, \tag{4.7}$$

$$\boldsymbol{p}_{d,\ell} \;:=\; \boldsymbol{p}_{d-1,\ell} + \boldsymbol{\Omega}_n(z_d) \mathbin{.*} \boldsymbol{p}_{d-1,\ell-1} \qquad \text{(order-dependent)}, \tag{4.8}$$

where $\boldsymbol{\Omega}_n(z_d)$ denotes the row of $\boldsymbol{\Omega}_n$ corresponding to the chosen z_d, and the operator $.*$ denotes the element-wise vector multiplication. Since the vectors $\boldsymbol{p}_{d-1}$ or $\boldsymbol{p}_{d-1,\ell}$ are no longer needed in the next iteration, we can simply overwrite $\boldsymbol{p}_{d-1}$ with $\boldsymbol{p}_d$ or overwrite $\boldsymbol{p}_{d-1,\ell}$ with $\boldsymbol{p}_{d,\ell}$. Hence, starting with the vectors $\boldsymbol{p}_0 := \mathbf{1}_n$ or $\boldsymbol{p}_{0,\ell} := \mathbf{1}_n$, we require only $\mathcal{O}(n)$ storage for product weights and $\mathcal{O}(qn)$ storage for order-dependent weights of finite order q.

The trick now is to order the indices $z \in U_n$ and $k \in \mathbb{Z}_n$ in (4.5) and (4.6) in a clever way to allow fast matrix-vector multiplications.

4.2. Fast matrix-vector product. To get an idea of the general concept, we first consider the matrix $\boldsymbol{\Omega}_n$ for n prime. Using the Rader factorization [15, 18], this matrix can be reordered such that it has a circulant submatrix of size $(n-1)\times(n-1)$. This is achieved by taking the indices as $z = g^i$ and $k = g^{-i'}$ for $0 \le i, i' \le n-2$, where g is a primitive root of n (i.e., g is a generator for the cyclic group U_n). We denote this particular ordering of $\boldsymbol{\Omega}_n$ by $\boldsymbol{\Omega}_n^{\langle g\rangle}$.

The effect of such reordering is explained most easily by an example. Take $n = 11$; then the possible components for the generating vector which we want to consider are

$$U_{11} \;=\; \{1,2,3,4,5,6,7,8,9,10\}.$$

Since n is prime, we can always find a generator for this group. For our example we choose $g = 2$. The ordering for the z indices we want to use is then

$$[g^i : 0 \le i \le 9] \;=\; [1,2,4,8,5,10,9,7,3,6].$$

For the indices $k \neq 0$ we take the negative powers of g,

$$[g^{-i'} : 0 \le i' \le 9] \;=\; [1,6,3,7,9,10,5,8,4,2],$$

which can easily be seen as reversing the order of the z indices, but keeping the first index fixed (the zeroth power of g). This reordering, together with a column of zeros

for $k = 0$, gives a matrix with the following structure:

$$\boldsymbol{\Omega}_{11}^{\langle 2\rangle} = \omega \left(\frac{1}{11} \left[\begin{array}{cccccccccc|c} 1 & 6 & 3 & 7 & 9 & 10 & 5 & 8 & 4 & 2 & 0 \\ 2 & 1 & 6 & 3 & 7 & 9 & 10 & 5 & 8 & 4 & 0 \\ 4 & 2 & 1 & 6 & 3 & 7 & 9 & 10 & 5 & 8 & 0 \\ 8 & 4 & 2 & 1 & 6 & 3 & 7 & 9 & 10 & 5 & 0 \\ 5 & 8 & 4 & 2 & 1 & 6 & 3 & 7 & 9 & 10 & 0 \\ 10 & 5 & 8 & 4 & 2 & 1 & 6 & 3 & 7 & 9 & 0 \\ 9 & 10 & 5 & 8 & 4 & 2 & 1 & 6 & 3 & 7 & 0 \\ 7 & 9 & 10 & 5 & 8 & 4 & 2 & 1 & 6 & 3 & 0 \\ 3 & 7 & 9 & 10 & 5 & 8 & 4 & 2 & 1 & 6 & 0 \\ \underbrace{6 \quad 3 \quad 7 \quad 9 \quad 10 \quad 5 \quad 8 \quad 4 \quad 2 \quad 1}_{\boldsymbol{C}_{11}} & 0 \end{array} \right] \right).$$

As was proved in [13], we can halve the number of rows and the number of columns of the submatrix $\boldsymbol{C}_{11}$ due to the symmetry $\omega(x) = \omega(1 - x)$. More specifically, we can write $\boldsymbol{C}_{11}$ in a reduced form (denoted by a $\sim$ instead of $=$)

$$\boldsymbol{\Omega}_{11}^{\langle 2\rangle} \sim \omega \left(\frac{1}{11} \left[\begin{array}{ccccc\|c} 1 & 6 & 3 & 7 & 9 & 0 \\ 2 & 1 & 6 & 3 & 7 & 0 \\ 4 & 2 & 1 & 6 & 3 & 0 \\ 8 & 4 & 2 & 1 & 6 & 0 \\ \underbrace{5 \quad 8 \quad 4 \quad 2 \quad 1}_{\widetilde{\boldsymbol{C}}_{11}} & 0 \end{array} \right] \right),$$

where we remember that the full matrix $\boldsymbol{\Omega}_{11}^{\langle 2\rangle}$ can be obtained by doubling each partition vertically and also horizontally up to the double bar.

A matrix with a structure like the submatrix $\boldsymbol{C}_{11}$ (as well as $\widetilde{\boldsymbol{C}}_{11}$) is called a [illegible] and is completely defined by its first column $\boldsymbol{c}$. Moreover, matrix-vector multiplication with a circulant matrix of order n can be done in $\mathcal{O}(n \log(n))$ using fast Fourier transforms.

LEMMA 4.1 (fast matrix-vector product with a circulant matrix). [illegible] $\boldsymbol{C}_n$ [illegible] n [illegible] $\boldsymbol{c}$ [illegible] $\mathcal{O}(n \log(n))$ [illegible] $\mathcal{O}(n^2)$ [illegible] $\mathcal{O}(n)$ [illegible] $\boldsymbol{F}_n$ [illegible] $\boldsymbol{F}_n^{-1}$

$$\begin{aligned} \boldsymbol{C}_n \, \boldsymbol{x} &= \boldsymbol{F}_n^{-1} \, \operatorname{diag}(\boldsymbol{F}_n \, \boldsymbol{c}) \, \boldsymbol{F}_n \, \boldsymbol{x} \\ &= \operatorname{IFFT}(\operatorname{FFT}(\boldsymbol{c}) \mathbin{.*} \operatorname{FFT}(\boldsymbol{x})) \,. \end{aligned}$$

A fast CBC algorithm based on this fast matrix-vector product was first introduced for n prime in [13] and later extended for composite n in [14]. The general idea for composite n is that the complete matrix $\boldsymbol{\Omega}_n$ can be partitioned in blocks which have a circulant or block-circulant structure. Basically, we consider a vertical partition $\boldsymbol{A}_{d_i}^{\langle g\rangle}$ for each divisor d_i of n

$$\boldsymbol{\Omega}_n^{\langle g\rangle} := \left[\begin{array}{c|c|c|c} \boldsymbol{A}_{d_1}^{\langle g\rangle} & \boldsymbol{A}_{d_2}^{\langle g\rangle} & \cdots & \boldsymbol{A}_{d_\nu}^{\langle g\rangle} \end{array} \right],$$

where g is a generator (or technically, a set of generators) for the group U_n, and each $\boldsymbol{A}_{d_i}^{\langle g\rangle}$ is constructed out of circulant (or technically, nested block-circulant) matrices $\boldsymbol{B}_{n/d_i}^{\langle g\rangle}$. The full details can be found in [14], which describes how the ordering is done and why a matrix-vector multiplication with $\boldsymbol{\Omega}_n^{\langle g\rangle}$ takes $\mathcal{O}(n\log(n))$ time using $\mathcal{O}(n)$ memory.

For embedded lattice rules, we shall restrict ourselves to n being a power of a prime p, because in this case the matrix $\boldsymbol{\Omega}_{p^m}^{\langle g\rangle}$ is of a particularly simple form. For this case we have $|U_{p^m}| = p^{m-1}(p-1)$. We will state the results for $p \neq 2$ and $p = 2$ in two separate theorems below; both results follow from [14, Theorem 4 and Corollary 4]. These two theorems differ in subtle yet crucial ways: the form of the matrices $\boldsymbol{B}_{p^m}^{\langle g\rangle}$ differs, and there are two degenerate cases $\boldsymbol{B}_{2^1}^{\langle 5\rangle}$ and $\boldsymbol{B}_{2^0}^{\langle 5\rangle}$ for $p = 2$, while there is only one degenerate case $\boldsymbol{B}_{p^0}^{\langle g\rangle}$ for $p \neq 2$. In the theorems $\mathbf{1}_t \otimes \boldsymbol{B}$ denotes the stacking of t replications of a matrix $\boldsymbol{B}$.

THEOREM 4.2 (block partitioning $\boldsymbol{\Omega}_{p^m}$, p prime, $p \neq 2$). [illegible]

$$\boldsymbol{\Omega}_{p^m} := \left[\omega\left(\frac{kz \bmod p^m}{p^m}\right)\right]_{\substack{z\in U_{p^m}\\ k\in\mathbb{Z}_{p^m}}},$$

[illegible] $\omega(x) = \omega(1-x)$ [illegible] p^m [illegible]

$$\boldsymbol{\Omega}_{p^m}^{\langle g\rangle} := \left[\begin{array}{c|c|c|c} \mathbf{1}_{p^0}\otimes \boldsymbol{B}_{p^m}^{\langle g\rangle} & \mathbf{1}_{p^1}\otimes \boldsymbol{B}_{p^{m-1}}^{\langle g\rangle} & \cdots & \mathbf{1}_{p^{m-1}}\otimes \boldsymbol{B}_{p^1}^{\langle g\rangle} \;\big|\; \mathbf{1}_{p^{m-1}(p-1)}\otimes \boldsymbol{B}_{p^0}^{\langle g\rangle} \end{array}\right],$$

[illegible] g [illegible] U_{p^m} [illegible] $\boldsymbol{B}_{p^0}^{\langle g\rangle} := [\omega(0)]$ [illegible] $\ell \geq 1$ [illegible] $\boldsymbol{B}_{p^\ell}^{\langle g\rangle}$ [illegible] $p^{m-\ell}$ [illegible]

$$\boldsymbol{B}_{p^\ell}^{\langle g\rangle} := \left[\begin{array}{c|c} \boldsymbol{M}_{p^\ell}^{\langle g\rangle} & \boldsymbol{M}_{p^\ell}^{\langle g\rangle} \\ \hline \boldsymbol{M}_{p^\ell}^{\langle g\rangle} & \boldsymbol{M}_{p^\ell}^{\langle g\rangle} \end{array}\right],$$

[illegible]

$$\boldsymbol{M}_{p^\ell}^{\langle g\rangle} := \left[\omega\left(\frac{kz \bmod p^\ell}{p^\ell}\right)\right]_{\substack{z\in \langle\!\langle g\rangle\!\rangle_{p^\ell}\\ k\in\langle\!\langle g^{-1}\rangle\!\rangle_{p^\ell}}}.$$

[illegible] $\langle\!\langle g\rangle\!\rangle_{p^\ell}$ [illegible] $[g^i \bmod p^\ell : 0 \leq i \leq \phi(p^\ell)/2 - 1]$ [illegible] U_{p^ℓ} [illegible]

THEOREM 4.3 (block partitioning $\boldsymbol{\Omega}_{2^m}$). [illegible]

$$\boldsymbol{\Omega}_{2^m} := \left[\omega\left(\frac{kz \bmod 2^m}{2^m}\right)\right]_{\substack{z\in U_{2^m}\\ k\in\mathbb{Z}_{2^m}}},$$

[illegible] $\omega(x) = \omega(1-x)$ [illegible] 2^m [illegible]

$$\boldsymbol{\Omega}_{2^m}^{\langle 5\rangle} := \left[\begin{array}{c|c|c|c} \mathbf{1}_{2^0}\otimes \boldsymbol{B}_{2^m}^{\langle 5\rangle} & \mathbf{1}_{2^1}\otimes \boldsymbol{B}_{2^{m-1}}^{\langle 5\rangle} & \cdots & \mathbf{1}_{2^{m-1}}\otimes \boldsymbol{B}_{2^1}^{\langle 5\rangle} \;\big|\; \mathbf{1}_{2^{m-1}(2-1)}\otimes \boldsymbol{B}_{2^0}^{\langle 5\rangle} \end{array}\right],$$

[illegible] $\boldsymbol{B}_{2^0}^{\langle 5\rangle} := [\omega(0)]$ [illegible] $\boldsymbol{B}_{2^1}^{\langle 5\rangle} := [\omega(1/2)]$ [illegible] $\ell \geq 2$ [illegible] $\boldsymbol{B}_{2^\ell}^{\langle 5\rangle}$ [illegible] $2^{m-\ell}$ [illegible]

$$\boldsymbol{B}_{2^\ell}^{\langle 5\rangle} := \left[\begin{array}{c|c} \boldsymbol{M}_{2^\ell}^{\langle 5\rangle} & \boldsymbol{M}_{2^\ell}^{\langle 5\rangle} \\ \hline \boldsymbol{M}_{2^\ell}^{\langle 5\rangle} & \boldsymbol{M}_{2^\ell}^{\langle 5\rangle} \end{array}\right], \tag{4.9}$$

$$\boldsymbol{M}_{2^\ell}^{\langle 5\rangle} \;:=\; \left[\omega\left(\frac{kz \bmod 2^\ell}{2^\ell}\right)\right]_{\substack{z\in\langle 5\rangle_{2^\ell}\\ k\in\langle 5^{-1}\rangle_{2^\ell}}} .$$

$\langle 5\rangle_{2^\ell}$ $[5^i \bmod 2^\ell : 0 \le i \le 2^{\ell-2}-1]$ U_{2^ℓ}

We will illustrate this theorem with an example shortly. First we discuss the effect of the symmetry of ω. Note that Theorem 4.3 rests upon the partition

$$U_{2^\ell} \;=\; \langle 5\rangle_{2^\ell} \cup (-1)\langle 5\rangle_{2^\ell} \qquad \text{for } \ell \ge 2\,. \tag{4.10}$$

If we do not make use of the symmetry of ω, formula (4.9) would actually look like

$$\widehat{\boldsymbol{B}}_{2^\ell}^{\langle 5\rangle} \;:=\; \left[\begin{array}{c|c} \boldsymbol{M}_{2^\ell}^{\langle 5\rangle} & \boldsymbol{M}_{2^\ell}^{\langle 5\rangle\star} \\ \hline \boldsymbol{M}_{2^\ell}^{\langle 5\rangle\star} & \boldsymbol{M}_{2^\ell}^{\langle 5\rangle} \end{array}\right], \text{ with } \left\{\begin{array}{rcl} \boldsymbol{M}_{2^\ell}^{\langle 5\rangle} &=& \left[\omega\left(\frac{kz \bmod 2^\ell}{2^\ell}\right)\right]_{\substack{z\in\langle 5\rangle_{2^\ell}\\ k\in\langle 5^{-1}\rangle_{2^\ell}}}, \\ \boldsymbol{M}_{2^\ell}^{\langle 5\rangle\star} &=& \left[\omega\left(\frac{-kz \bmod 2^\ell}{2^\ell}\right)\right]_{\substack{z\in\langle 5\rangle_{2^\ell}\\ k\in\langle 5^{-1}\rangle_{2^\ell}}}, \end{array}\right.$$

and the stacking of the matrices $\widehat{\boldsymbol{B}}_{2^\ell}^{\langle 5\rangle}$ would be

$$\widehat{\boldsymbol{\Omega}}_{2^m}^{\langle 5\rangle} \;:=\; \left[\; \boldsymbol{R}_{2^0}\,\widehat{\boldsymbol{B}}_{2^m}^{\langle 5\rangle} \;\middle|\; \cdots \;\middle|\; \boldsymbol{R}_{2^{m-2}}\,\widehat{\boldsymbol{B}}_{2^2}^{\langle 5\rangle} \;\middle|\; \mathbf{1}_{2^{m-1}} \otimes \boldsymbol{B}_{2^1}^{\langle 5\rangle} \;\middle|\; \mathbf{1}_{2^{m-1}} \otimes \boldsymbol{B}_{2^0}^{\langle 5\rangle} \;\right], \tag{4.11}$$

where $\boldsymbol{R}_{2^\ell} := \boldsymbol{I}_2 \otimes \mathbf{1}_{2^\ell} \otimes \boldsymbol{I}_{2^{m-\ell-2}}$, which stacks the top and bottom half of $\widehat{\boldsymbol{B}}_{2^{m-\ell}}^{\langle 5\rangle}$ separately. The symmetry of ω implies $\boldsymbol{M}_{2^\ell}^{\langle 5\rangle} = \boldsymbol{M}_{2^\ell}^{\langle 5\rangle\star}$ from which the theorem follows.

We consider now the example $n = 2^4 = 16$, initially without using the symmetry of ω. Following (4.10), the first half of U_{16} is generated by 5,

$$[5^i \bmod 16 : 0 \le i \le 3] \;=\; [1, 5, 9, 13]\,,$$

where i stays less than $\phi(n)/2$ since this is only half of the group U_{16}. The second half is a "copy" of this subgroup, but multiplied with $(-1) \equiv 15 \,(\mathrm{mod}\, 16)$,

$$[(-1)\,5^i \bmod 16 : 0 \le i \le 3] \;=\; [15, 11, 7, 3]\,.$$

With this row ordering we can associate a column ordering to get the circulant structure. This can be achieved by reversing the row ordering except for the first element. We obtain the first submatrix for the divisor $2^0 = 1$:

$$\widehat{\boldsymbol{B}}_{16}^{\langle 5\rangle} \;=\; \omega\left(\frac{1}{16}\left[\begin{array}{cccc|cccc} 1 & 13 & 9 & 5 & 15 & 3 & 7 & 11 \\ 5 & 1 & 13 & 9 & 11 & 15 & 3 & 7 \\ 9 & 5 & 1 & 13 & 7 & 11 & 15 & 3 \\ 13 & 9 & 5 & 1 & 3 & 7 & 11 & 15 \\ \hline 15 & 3 & 7 & 11 & 1 & 13 & 9 & 5 \\ 11 & 15 & 3 & 7 & 5 & 1 & 13 & 9 \\ 7 & 11 & 15 & 3 & 9 & 5 & 1 & 13 \\ 3 & 7 & 11 & 15 & 13 & 9 & 5 & 1 \end{array}\right]\right).$$

The submatrices for the other divisors can be constructed similarly, and we get

$$\widehat{\boldsymbol{B}}_8^{\langle 5\rangle} = \omega\left(\frac{1}{8}\left[\begin{array}{cc|cc} 1 & 5 & 7 & 3 \\ 5 & 1 & 3 & 7 \\ \hline 7 & 3 & 1 & 5 \\ 3 & 7 & 5 & 1 \end{array}\right]\right), \widehat{\boldsymbol{B}}_4^{\langle 5\rangle} = \omega\left(\frac{1}{4}\left[\begin{array}{c|c} 1 & 3 \\ \hline 3 & 1 \end{array}\right]\right),$$
$$\boldsymbol{B}_2^{\langle 5\rangle} = \omega\left(\frac{1}{2}\left[\begin{array}{c} 1 \end{array}\right]\right), \boldsymbol{B}_1^{\langle 5\rangle} = \omega(\left[\begin{array}{c} 0 \end{array}\right]).$$

Note that all these matrices (except for the last two trivial ones) have a block-circulant-with-circulant-blocks structure, which is consistent with the two partitions in their multiplicative groups (see (4.10)). Using (4.11), the full matrix looks like this (where italic font means redundancy)

$$\widehat{\boldsymbol{\Omega}}_{16}^{\langle 5\rangle} = \omega\left(\frac{1}{16}\left[\begin{array}{cccc|cccc|cccc|cc|cc} 1 & 13 & 9 & 5 & 15 & 3 & 7 & 11 & 2 & 10 & 14 & 6 & 4 & 12 & 8 & 0 \\ 5 & 1 & 13 & 9 & 11 & 15 & 3 & 7 & 10 & 2 & 6 & 14 & \prime & & & \\ 9 & 5 & 1 & 13 & 7 & 11 & 15 & 3 & & & \prime & \cdot & \prime & & & \\ 13 & 9 & 5 & 1 & 3 & 7 & 11 & 15 & & & \cdot & \prime & \prime & & & \\ \hline 15 & 3 & 7 & 11 & 1 & 13 & 9 & 5 & 14 & 6 & 2 & 10 & 12 & 4 & & \\ 11 & 15 & 3 & 7 & 5 & 1 & 13 & 9 & 6 & 14 & 10 & 2 & & \prime & & \\ 7 & 11 & 15 & 3 & 9 & 5 & 1 & 13 & \prime & \cdot & & & & \prime & & \\ 3 & 7 & 11 & 15 & 13 & 9 & 5 & 1 & \cdot & \prime & & & & \prime & & \end{array}\right)\right).$$

We now apply the symmetry of ω by mapping each $k \in \mathbb{Z}_n$ to $\min(n-k, k)$ and we get the matrix

$$\boldsymbol{\Omega}_{16}^{\langle 5\rangle} = \omega\left(\frac{1}{16}\left[\begin{array}{cccc|cccc|cc|cc|cc} 1 & 3 & 7 & 5 & & & & & 2 & 6 & \cdot & 4 & \prime & 8 & 0 \\ 5 & 1 & 3 & 7 & & & & & 6 & 2 & \cdot & \prime & \prime & & \\ 7 & 5 & 1 & 3 & & & & & \cdot & \cdot & \cdot & \prime & \prime & & \\ 3 & 7 & 5 & 1 & & & & & \cdot & \cdot & \cdot & \prime & \prime & & \\ \hline & & & & & & & & \cdot & \cdot & \cdot & \prime & \prime & & \\ & & & & & & & & \cdot & \cdot & \cdot & \prime & \prime & & \\ & & & & & & & & \cdot & \cdot & \cdot & \prime & \prime & & \\ & & & & & & & & \cdot & \cdot & \cdot & \prime & \prime & & \end{array}\right)\right),$$

as promised in Theorem 4.3. We can then represent this matrix in a reduced form:

$$\boldsymbol{\Omega}_{16}^{\langle 5\rangle} \sim \omega\left(\frac{1}{16}\left[\begin{array}{cccc|cc|c||c|c} 1 & 3 & 7 & 5 & 2 & 6 & 4 & 8 & 0 \\ 5 & 1 & 3 & 7 & 6 & 2 & \prime & & \\ 7 & 5 & 1 & 3 & & \cdot & \prime & & \\ 3 & 7 & 5 & 1 & \cdot & & \prime & & \end{array}\right)\right). \tag{4.12}$$

If we write out the reordering from Theorems 4.2 and 4.3 explicitly by means of permutations $\Pi_g^\top$ and $\Pi_{g^{-1}}$ on the rows and columns induced by g, we obtain

$$\boldsymbol{y} = \boldsymbol{\Omega}_{p^m}\,\boldsymbol{x} \quad \Leftrightarrow \quad \Pi_g^\top\,\boldsymbol{y} = \underbrace{\Pi_g^\top\,\boldsymbol{\Omega}_{p^m}\,\Pi_{g^{-1}}}_{\boldsymbol{\Omega}_{p^m}^{\langle g\rangle}}\,\Pi_{g^{-1}}^\top\,\boldsymbol{x} = \boldsymbol{\Omega}_{p^m}^{\langle g\rangle}\,\Pi_{g^{-1}}^\top\,\boldsymbol{x}.$$

Note that the permutations do not complicate matters, since the starting vectors are all ones and we know that the vectors

$$\begin{aligned}
\boldsymbol{p}_d^{\langle g^{-1}\rangle} &:= \Pi_{g^{-1}}^{\top}\,\boldsymbol{p}_d \;=\; (\mathbf{1}_n + \gamma_d\,\boldsymbol{\Omega}_{p^m}^{\langle g\rangle}(z_d)) \mathbin{.*} \boldsymbol{p}_{d-1}^{\langle g^{-1}\rangle} && \text{(product)},\\
\boldsymbol{p}_{d,\ell}^{\langle g^{-1}\rangle} &:= \Pi_{g^{-1}}^{\top}\,\boldsymbol{p}_{d,\ell} \;=\; \boldsymbol{p}_{d-1,\ell}^{\langle g^{-1}\rangle} + \boldsymbol{\Omega}_{p^m}^{\langle g\rangle}(z_d) \mathbin{.*} \boldsymbol{p}_{d-1,\ell-1}^{\langle g^{-1}\rangle} && \text{(order-dependent)}
\end{aligned}$$

have the same symmetry structure as the rows of the matrix $\boldsymbol{\Omega}_{p^m}^{\langle g\rangle}$, which means that we can keep a reduced version in memory and never have to refer to the unpermuted version. Also, from the resulting vector $\boldsymbol{E}_d^2$ we need only the index of the minimal entry, from which we obtain z_d via the sequence $\langle g\rangle_{p^m}$.

Combining this with Lemma 4.1, we observe that we need only the fast Fourier transforms of the first column of the smaller circulant matrices in the reduced matrix. This requires $\mathcal{O}(n)$ memory. A matrix-vector product with a vector of length $n \times 1$ involves $m+1$ (the number of divisors) smaller matrix-vector products with circulant matrices, with a total cost of

$$\sum_{\ell=0}^{m} \phi(p^{m-\ell}) \log(\phi(p^{m-\ell})) \;=\; \mathcal{O}(n \log(n))\,.$$

We can calculate all these circulant matrix-vector products in turn, starting from the smallest matrix up to the largest matrix, while accumulating the results. In each step the previous result vector is replicated $p-1$ times before the next result vector is added. The summing alone has a memory complexity of $\mathcal{O}(n)$ and a time complexity of $\mathcal{O}(n)$, which is negligible against the $\mathcal{O}(n \log(n))$ cost of the matrix-vector products. Again we specify this as a specialized version of the general case in [14, Theorem 5]. (There is no need for a special summing order as is the case for general n in [14].)

THEOREM 4.4 (fast matrix-vector product $\boldsymbol{\Omega}_{p^m}\,\boldsymbol{p}$). [illegible] $\boldsymbol{\Omega}_{p^m}\,\boldsymbol{p}$ [illegible] $\boldsymbol{p}$ [illegible] (4.7) [illegible] (4.8) [illegible] $\mathcal{O}(n \log(n))$ [illegible] $\mathcal{O}(n)$ [illegible] $\boldsymbol{B}_{p^\ell}^{\langle g\rangle}$ [illegible]

4.3. Exploiting the embedding structure. In Theorems 4.2 and 4.3 we used implicitly the property that a generator g for the group U_{p^m} also generates U_{p^ℓ} for all $\ell \le m$, with the obvious adjustment for $p=2$ where 5 generates U_{2^ℓ} in two partitions for all $\ell \le m$. This actually follows from the fact (see, e.g., [1, section 24.3.4]) that if we have a generator r for U_p, then a generator g for U_{p^ℓ}, $\ell \ge 2$, can be found by taking

$$g \;=\; \begin{cases} r+p & \text{if } r^{p-1} \equiv 1 \,(\mathrm{mod}\, p^2)\,,\\ r & \text{otherwise}\,. \end{cases} \tag{4.13}$$

It can be checked that if one searches for generators in increasing numerical order starting from 2, then the first p for which the smallest generator of U_p does not generate U_{p^2} is as high as 40487 (and the first p for which one of its generators does not generate U_{p^2} is $p=29$). In other words, as long as p is smaller than 40487, the smallest generator of U_p also generates U_{p^ℓ} for all $\ell \ge 1$. Furthermore, the most logical choice in practice is $p=2$ for which we can always pick 5 as a pseudogenerator. This gives us a way to extend a matrix for p^m points to a matrix for p^{m+1} points by simply adding one block in front.

We now show the extension to $\mathbf{\Omega}_{32}^{\langle 5\rangle}$ starting from $\mathbf{\Omega}_{16}^{\langle 5\rangle}$:

$$\mathbf{\Omega}_{32}^{\langle 5\rangle} \;=\; \left[\begin{array}{c|c} \boldsymbol{B}_{32}^{\langle 5\rangle} & \begin{array}{c}\mathbf{\Omega}_{16}^{\langle 5\rangle}\\ \hline \mathbf{\Omega}_{16}^{\langle 5\rangle}\end{array}\end{array}\right],$$

where we have recursively

$$\mathbf{\Omega}_{16}^{\langle 5\rangle} \;=\; \left[\begin{array}{c|c} \boldsymbol{B}_{16}^{\langle 5\rangle} & \begin{array}{c}\mathbf{\Omega}_{8}^{\langle 5\rangle}\\ \hline \mathbf{\Omega}_{8}^{\langle 5\rangle}\end{array}\end{array}\right],\; \mathbf{\Omega}_{8}^{\langle 5\rangle} \;=\; \left[\begin{array}{c|c} \boldsymbol{B}_{8}^{\langle 5\rangle} & \begin{array}{c}\mathbf{\Omega}_{4}^{\langle 5\rangle}\\ \hline \mathbf{\Omega}_{4}^{\langle 5\rangle}\end{array}\end{array}\right],\; \mathbf{\Omega}_{4}^{\langle 5\rangle} \;=\; \left[\begin{array}{c|c} \boldsymbol{B}_{4}^{\langle 5\rangle} & \begin{array}{c}\mathbf{\Omega}_{2}^{\langle 5\rangle}\\ \hline \mathbf{\Omega}_{2}^{\langle 5\rangle}\end{array}\end{array}\right],$$

with $\mathbf{\Omega}_{2}^{\langle 5\rangle} = [\boldsymbol{B}_{2}^{\langle 5\rangle} | \boldsymbol{B}_{1}^{\langle 5\rangle}]$. In reduced form (and a common scaling of $1/32$) we have

$$\mathbf{\Omega}_{32}^{\langle 5\rangle} \sim$$

$$\omega\left(\frac{1}{32}\left[\begin{array}{cccccccc|cccc|cc|c||c|c}
1 & 13 & 9 & 11 & 15 & 3 & 7 & 5 & 2 & 6 & 14 & 10 & 4 & 12 & 8 & 16 & 0\\
5 & 1 & 13 & 9 & 11 & 15 & 3 & 7 & 10 & 2 & 6 & 14 & 12 & 4 & & & \\
7 & 5 & 1 & 13 & 9 & 11 & 15 & 3 & 14 & 10 & 2 & 6 & & & & & \\
3 & 7 & 5 & 1 & 13 & 9 & 11 & 15 & 6 & 14 & 10 & 2 & & & & & \\
15 & 3 & 7 & 5 & 1 & 13 & 9 & 11 & & & & & & & & & \\
11 & 15 & 3 & 7 & 5 & 1 & 13 & 9 & & & & & & & & & \\
9 & 11 & 15 & 3 & 7 & 5 & 1 & 13 & & & & & & & & & \\
13 & 9 & 11 & 15 & 3 & 7 & 5 & 1 & & & & & & & & &
\end{array}\right]\right).$$

In section 4.2 we explained how to sum up the result vectors of a matrix-vector multiplication with the matrix $\mathbf{\Omega}_{p^m}$ in $\mathcal{O}(n\log(n))$ time. In fact, since n is of the form p^m, any method used to sum up the intermediate result vectors would have been sufficient to get at the $\mathcal{O}(n\log(n))$ time complexity (see [14]). By accumulating the result vectors from the smallest matrix upward, we obtain intermediate result vectors which are precisely the matrix-vector products with the matrices $\mathbf{\Omega}_{p^\ell}$ for ℓ going from 0 up to $m-1$. Thus in an attempt to compute $e_{p^m,d}$, we get along the way the values of $e_{p^\ell,d}$ for ℓ from 0 to $m-1$ at no extra cost.

The availability of all these intermediate results is a nice side effect since then the cost for calculating $X_{m_1,m_2,d}$, which involves $e_{p^\ell,d}$ for all ℓ from m_1 to m_2, is no worse than the cost for calculating just $e_{p^{m_2},d}$. This brings us to the main result of this paper.

THEOREM 4.5 (fast construction of embedded lattice rules). $n = p^{m_2}$ p 2 $\mathcal{O}(dn\log(n))$ $\mathcal{O}(n)$ $\mathcal{O}(qn)$ q 1 $\mathcal{O}(dn(\log(n))^2)$

5. Using embedded lattice rules.

5.1. Ordering of lattice points. The main advantage of having a sequence of embedded lattice rules $L_0 \subset L_1 \subset L_2 \subset \cdots$ is that we can increase the number of points without having to throw away already computed function values. When the base is b (not necessarily prime anymore), we increase the number of points by a factor of b at each step along the sequence. For $\ell \geq 1$, let $\Delta L_\ell := L_\ell \setminus L_{\ell-1}$ denote the set of additional points from the point set $L_{\ell-1}$ to L_ℓ. We recall from Property 4 that the points in $L_{\ell-1}$ are generated by the indices k which are multiples of b in $\mathbb{Z}_{b^\ell}$.

Thus we can write

$$L_m \;=\; L_0 \,\cup\, \left(\bigcup_{\ell=1}^{m} \Delta L_\ell \right), \tag{5.1}$$

with $L_0 = \{\mathbf{0}\}$ and

$$\Delta L_\ell \;=\; \left\{ \left\{ \frac{k\boldsymbol{z}}{b^\ell} \right\} : k \in (\mathbb{Z}_{b^\ell} \setminus b\mathbb{Z}_{b^{\ell-1}}) \right\}. \tag{5.2}$$

Letting $m \to \infty$ in (5.1) leads formally to an extensible lattice sequence L_∞ with an infinite number of points. In this paper we consider lattice point sets where the number of points is finite, that is, the union in (5.1) goes only to a predefined upper limit m_2. But, due to the fast construction, this limit can be taken rather large.

Note that the set of additional points ΔL_ℓ from the point set $L_{\ell-1}$ to L_ℓ actually comprises of $b-1$ shifted versions of $L_{\ell-1}$. We put this forward as another property of embedded lattice rules (see also [7, Theorem 2.3]).

[illegible] 5. The lattice point set L_m can be considered as the union of b shifted versions (dependent on $\boldsymbol{z}$) of the lattice point set L_{m-1}, as well as the union of b^2 shifted versions of L_{m-2}, and so on. Trivially, L_m is the union of b^m shifted versions of the initial point $\mathbf{0}$.

Explicitly, we have

$$L_m \;=\; \bigcup_{k_0=0}^{b-1} \cdots \bigcup_{k_\ell=0}^{b-1} \left(L_{m-\ell-1} + \frac{k_\ell \boldsymbol{z}}{b^{m-\ell}} + \cdots + \frac{k_0 \boldsymbol{z}}{b^m} \right) \quad \forall\, \ell = 0, \ldots, m-1\,,$$

where the addition is interpreted as in the definition $P_n + \boldsymbol{\Delta} := \{\{\boldsymbol{t} + \boldsymbol{\Delta}\} : \boldsymbol{t} \in P_n\}$ (the inner braces indicate taking the fractional part).

With the generating vector constructed by Algorithm 2 for n ranging from b^{m_1} to b^{m_2}, we can start with an approximation of the integral using the point set L_{m_1}, and then gradually add blocks of additional points $\Delta L_{m_1+1}, \Delta L_{m_1+2}, \ldots, \Delta L_{m_2}$ until we are satisfied with the error (provided that m_2 is large enough). This usage of the sequence of embedded lattice rules is justified by the knowledge that every lattice point set L_m is good for $m_1 \le m \le m_2$.

A more interesting usage would be to apply the lattice rule one point at a time, i.e., as a low-discrepancy sequence. Since we may end up with an incomplete lattice point set, there is no guarantee that these additional points will be well distributed over the unit cube, and we could even end up with these points all falling in one half of the cube. In fact, we are sure to get a bad ordering of the points if we generate the points from each block ΔL_ℓ according to the natural order of the k indices in (5.2). The solution to this inappropriate ordering of the points is given by replacing k/b^ℓ in (5.2) with the radical inverse function, possibly in combination with Gray code ordering. These schemes provide a better ordering of the numbers k/b^ℓ in the unit interval, and this property is preserved under multiplication by $\boldsymbol{z}$ (and then taking the fractional part) so that the ordering of every one-dimensional projection of the lattice points is good. Both schemes are well-known tools borrowed from the vast literature on digital sequences; see, for example, [12].

The [illegible] function takes an integer in base b and mirrors it around the radix point. For an integer $k = \sum_{\ell=0}^{m-1} k_\ell b^\ell = (k_{m-1} \ldots k_0)_b$ the radical inverse is given as $\phi_b(k) = \sum_{\ell=0}^{m-1} k_\ell b^{-(\ell+1)} = (0.k_0 k_1 \ldots k_{m-1})_b$. The [illegible] variant uses

this same transformation but starts with a Gray code ordering in base b of the k indices (i.e., successive indices differ only in one b-ary digit). The latter is computationally more efficient since then the next index can be obtained from the previous one by a simple digit change and the actual radical inversion can be avoided.

We summarize the usage of these schemes as a property of embedded lattice rules. Note that they preserve the embedding property $L_0 \subset L_1 \subset \cdots$ of the lattice point sets, while providing a better ordering of the points within each block of additional points ΔL_ℓ. More discussions can be found in [7].

PROPERTY 6. The lattice point set L_m can be used as a low-discrepancy sequence if the points are generated in the order

$$\boldsymbol{t}_k \;=\; \{\varphi_b(k)\,\boldsymbol{z}\} \qquad \text{for} \quad k = 0, 1, \ldots, b^m - 1\,,$$

where $\varphi_b(k)$ denotes either the radical inverse $\phi_b(k)$ of k in base b, or the corresponding Gray code variant.

We end this subsection with a simple illustration for the most practical case of $b = 2$:

$$\underbrace{\underbrace{\underbrace{\underbrace{\boldsymbol{t}_0 = \boldsymbol{0}}_{L_0},\ \underbrace{\boldsymbol{t}_1 = \tfrac{\boldsymbol{1}}{\boldsymbol{2}}}_{\Delta L_1}}_{L_1},\ \underbrace{\boldsymbol{t}_2, \boldsymbol{t}_3}_{\Delta L_2}}_{L_2},\ \underbrace{\boldsymbol{t}_4, \boldsymbol{t}_5, \boldsymbol{t}_6, \boldsymbol{t}_7}_{\Delta L_3}}_{L_3},\ \underbrace{\boldsymbol{t}_8, \boldsymbol{t}_9, \boldsymbol{t}_{10}, \boldsymbol{t}_{11}, \boldsymbol{t}_{12}, \boldsymbol{t}_{13}, \boldsymbol{t}_{14}, \boldsymbol{t}_{15}}_{\Delta L_4},\ \ldots\,.}_{L_4}$$

The number of points doubles as we move along the sequence, and the set of additional points ΔL_ℓ is generated by $\{k\boldsymbol{z}/2^\ell\}$ with k running through all the odd numbers up to $2^\ell - 1$. In Table 5.1 we show how the points are ordered under radical inverse and the Gray code variant.

TABLE 5.1
Ordering of lattice points.

	k	Natural order	Radical inverse	Gray code variant
L_0	0	0	0	0
ΔL_1	1	$1/2 = 0.5$	$(0.1)_2 = 0.5$	$(0.1)_2 = 0.5$
ΔL_2	2	$1/4 = 0.25$	$(0.01)_2 = 0.25$	$(0.11)_2 = 0.75$
	3	$3/4 = 0.75$	$(0.11)_2 = 0.75$	$(0.01)_2 = 0.25$
ΔL_3	4	$1/8 = 0.125$	$(0.001)_2 = 0.125$	$(0.011)_2 = 0.375$
	5	$3/8 = 0.375$	$(0.101)_2 = 0.625$	$(0.111)_2 = 0.875$
	6	$5/8 = 0.625$	$(0.011)_2 = 0.375$	$(0.101)_2 = 0.625$
	7	$7/8 = 0.875$	$(0.111)_2 = 0.875$	$(0.001)_2 = 0.125$
ΔL_4	8	$1/16 = 0.0625$	$(0.0001)_2 = 0.0625$	$(0.0011)_2 = 0.1875$
	9	$3/16 = 0.1875$	$(0.1001)_2 = 0.5625$	$(0.1011)_2 = 0.6875$
	10	$5/16 = 0.3125$	$(0.0101)_2 = 0.3125$	$(0.1111)_2 = 0.9375$
	11	$7/16 = 0.4375$	$(0.1101)_2 = 0.8125$	$(0.0111)_2 = 0.4375$
	12	$9/16 = 0.5625$	$(0.0011)_2 = 0.1875$	$(0.0101)_2 = 0.3125$
	13	$11/16 = 0.6875$	$(0.1011)_2 = 0.6875$	$(0.1101)_2 = 0.8125$
	14	$13/16 = 0.8125$	$(0.0111)_2 = 0.4375$	$(0.1001)_2 = 0.5625$
	15	$15/16 = 0.9375$	$(0.1111)_2 = 0.9375$	$(0.0001)_2 = 0.0625$
$\vdots$				

5.2. Error estimation. There is a comprehensive discussion on error estimation in [7] including error estimation based on the embedding structure of the lattice rules. Here we focus only on the simplest form of error estimation using random shifts.

In practice, once we have found a good generating vector $\boldsymbol{z}$, we generate a number of independent random shifts $\boldsymbol{\Delta}_1, \boldsymbol{\Delta}_2, \ldots, \boldsymbol{\Delta}_w$ and compute the approximations $Q_1, Q_2, \ldots, Q_w$, where Q_i is the shifted lattice rule approximation to the integral $I_d(f)$ based on the generating vector $\boldsymbol{z}$ and the shift $\boldsymbol{\Delta}_i$. Then we take the average

$$\overline{Q} := \frac{1}{w}(Q_1 + Q_2 + \cdots + Q_w)$$

as our final approximation to $I_d(f)$. An unbiased estimate for the standard error of our approximation is given by

$$\hat{\sigma} := \sqrt{\frac{1}{w(w-1)} \sum_{i=1}^{w} (Q_i - \overline{Q})^2}\,.$$

Using the Chebyshev inequality we can obtain a $(100 - 100/v^2)\%$ confidence interval $\overline{Q} \pm v\hat{\sigma}$ (with $v > 1$).

6. Numerical experiments.

6.1. Results from CBC construction. We construct good generating vectors of embedded lattice rule up to 360 dimensions using Algorithm 2 for n ranging from 2^{10} to $2^{20} \approx 1$ million (i.e., $b = 2$, $m_1 = 10$, and $m_2 = 20$). Our search criterion is the quantity $X_{m_1,m_2,d}(\boldsymbol{z})$ defined in (1.3), i.e.,

$$X_{10,20,d}(\boldsymbol{z}) \;=\; \max_{10 \le m \le 20} \frac{e_{2^m,d}(\boldsymbol{z})}{e_{2^m,d}(\boldsymbol{z}^{(m)})}\,,$$

where $\boldsymbol{z}^{(m)}$ is the generating vector constructed using Algorithm 1 for a 2^m-point rule (which we refer to as a good "fixed" rule). Let $\boldsymbol{z}^\star$ denote the vector resulting from Algorithm 2. Then the quantity $X_{10,20,d}(\boldsymbol{z}^\star)$ tells us how far off the vector $\boldsymbol{z}^\star$ is compared with the vectors $\boldsymbol{z}^{(m)}$ in terms of the maximum (over m) ratio of the worst-case errors.

We consider weighted Sobolev spaces (see Example 2) with both product weights and order-dependent weights. In particular, we consider equal product weights, decaying product weights, order-2 weights and order-3 weights. Our choices of weights together with the maxima of $X_{10,20,d}$ over all $1 \le d \le 360$ are given in Table 6.1.

TABLE 6.1
Maximum of $X_{10,20,d}(\boldsymbol{z}^\star)$ for various weights.

Weights	$\max_d X_{10,20,d}(\boldsymbol{z}^\star)$
product 1	1.53
product 0.5	1.48
product 0.3	1.47
product 0.1	1.41
product 0.05	1.37
product 0.01	1.30
product 0.9^j	1.50
product 0.75^j	1.42
product 0.5^j	1.30

Weights	$\max_d X_{10,20,d}(\boldsymbol{z}^\star)$
product $1/j$	1.41
product $1/j^2$	1.31
order-2	1.43
order-3 1	1.42
order-3 0.5	1.55
order-3 0.3	1.45
order-3 0.1	1.47
order-3 0.05	1.45
order-3 0.01	1.43

The product-type weights in the table are self-explanatory: "product 0.5" means $\gamma_j = 0.5$ and "product $1/j$" means $\gamma_j = 1/j$. We shall explain briefly our choices of order-2 and order-3 weights. First, since all one-dimensional projections of our lattice rules are left-rectangle rules with all points distinct, the weight Γ_1 in an order-dependent weight setting has no essential effect on the choice of the rule. We thus set $\Gamma_1 = 1$. Second, if we decide to construct a rule based on order-2 weights (i.e., if $\Gamma_\ell = 0$ for $\ell > 2$), then the rule is unique regardless of how we choose Γ_2, since in this case Γ_2 rescales only the worst-case error. Therefore without lost of generality we set $\Gamma_2 = 1$. In other words, such a rule is universally applicable for all order-2 problems. For order-3 weights, the rule depends only on the ratio Γ_3/Γ_2. Hence we use Γ_3 to specify the importance of order-3 interactions compared with order-2 interactions. In our table "order-3 0.1" corresponds to the choice $\Gamma_1 = \Gamma_2 = 1$ and $\Gamma_3 = 0.1$. Note that having equal product weights $\gamma_j = r$ is equivalent to having order-dependent weights with $\Gamma_\ell = r^\ell$. The generating vector for the order-2 rule is given in the appendix. From Table 6.1, the worst-case errors for the order-2 rule in the appendix are within a factor of 1.43 of the worst-case errors obtained with the results from the original CBC construction.

Starting from $X_{10,20,1}(\boldsymbol{z}^\star) = 1$, the empirical observation is that in most cases the numbers $X_{10,20,d}(\boldsymbol{z}^\star)$ increase initially as d increases, and then they decrease from some dimension onward. We see from Table 6.1 that $X_{10,20,d}(\boldsymbol{z}^\star)$ is substantially less than 1.6 for all our choices of weights. Thus we conclude that, in all our experiments, the embedded lattice rules are at most 1.6 times worse than the near-optimal rules constructed from Algorithm 1 for specific values of n. Put differently, we have rules with the optimal rate of convergence, but with a constant which is a factor of at most 1.6 times larger.

Enlarging the range of m reduces the overall quality. Figure 6.1 shows the profile of $X_{10,m_2,d}(\boldsymbol{z}^\star)$ with increasing d for order-2 rules with $m_2 = 15$, 20, and 25.

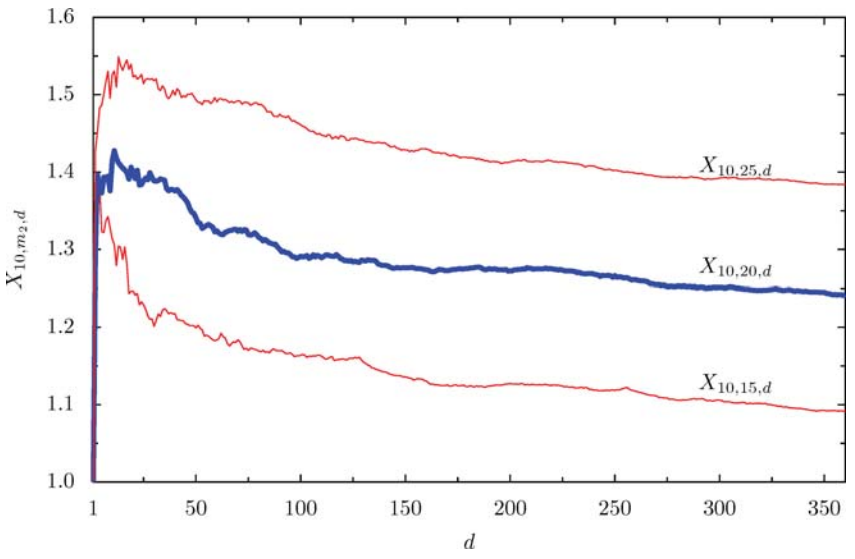


FIG. 6.1. *Profile of $X_{10,m_2,d}(\boldsymbol{z}^\star)$ for order-2 rules with $m_2 = 15$, 20, and 25 as d increases.*

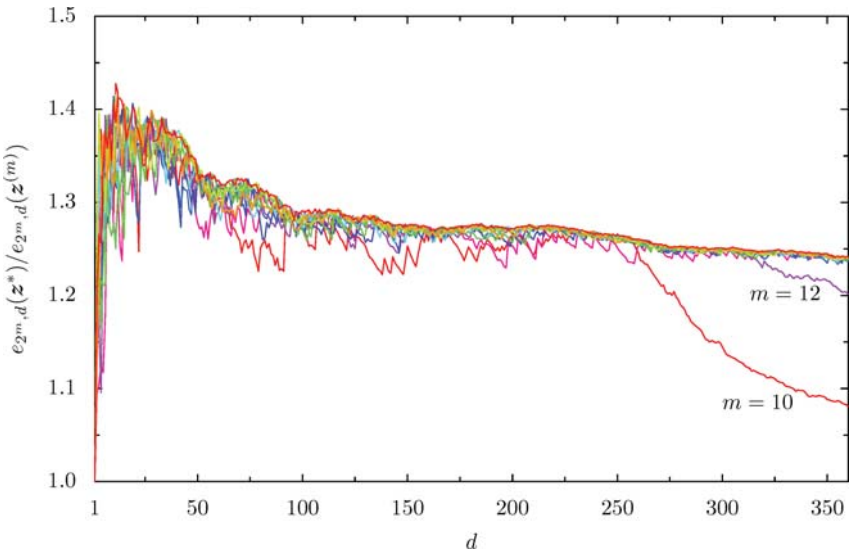


FIG. 6.2. *Initial segments of a good embedded lattice rule.*

It is understandable that by allowing the range of n to grow, the quality of the point set is sacrificed. But we observe that the quality remains very good for all feasible values of m_2.

Using initial segments of a good embedded lattice rule works as expected. The criterion $X_{10,20,d}(\boldsymbol{z}^\star)$ ensures that the resulting vector $\boldsymbol{z}^*$ is good for a 2^m-point rule for all values of m between 10 and 20. In other words, every point set in the embedded sequence $L_{10} \subset L_{11} \subset \cdots \subset L_{20}$ is good. (We refer to the smaller lattice point sets L_ℓ with $\ell < 20$ as the "initial segments" of the point set L_{20}.) In Figure 6.2 we plot the worst-case error ratios

$$\frac{e_{2^m,d}(\boldsymbol{z}^\star)}{e_{2^m,d}(\boldsymbol{z}^{(m)})}$$

for the case of the order-2 rule in the appendix and every $m = 10, \ldots, 20$.

As expected, the maximum of all these graphs corresponds to the graph for $X_{10,20,d}(\boldsymbol{z}^\star)$ in Figure 6.1. For most values of m we do better than $X_{10,20,d}(\boldsymbol{z}^\star)$ promises, but we note that all the individual curves actually stay close to $X_{10,20,d}(\boldsymbol{z}^\star)$. We also see that these curves cross each other in an erratic manner, which is important; otherwise we could redefine the error measure to make use of some assumed ordering. That said, in general it can be observed that the curves for higher m tend to be closer to the top.

For high dimensions we see that the curves for $m = 10$ and $m = 12$ depart from the general trend. This might be explained by observing that the differences between the worst-case errors of different choices for a new component decrease as the dimension increases. In other words, the different choices of generating vectors are equally bad when there are very few points in very high dimensions.

Using initial segments of a good fixed lattice rule does not work. We could wonder if it would be satisfactory to just use the initial segments from a good 2^{20}-point lattice rule with generating vector $\boldsymbol{z}^{(20)}$ constructed by Algorithm 1. (As

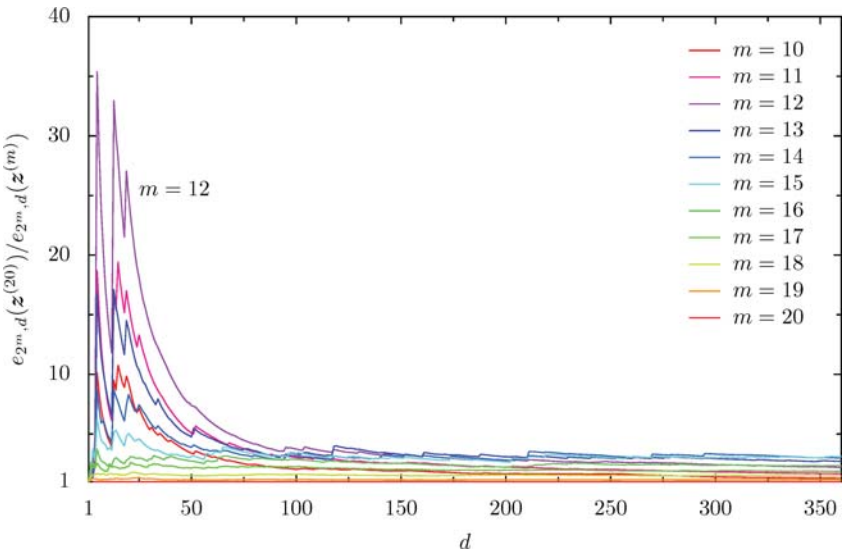


FIG. 6.3. *Initial segments of a good fixed lattice rule.*

before, by initial segments we mean the smaller lattice point sets L_ℓ embedded in L_{20} with $\ell < 20$.) If these initial segments would be any good, then there would be little point to using Algorithm 2. In Figure 6.3 we plot the worst-case error ratios

$$\frac{e_{2^m,d}(\boldsymbol{z}^{(20)})}{e_{2^m,d}(\boldsymbol{z}^{(m)})}$$

for every $m = 10, \ldots, 20$.

It is obvious from the figure that using a large good fixed lattice rule for a maximum number of points $n = 2^{m_2}$ tells one nothing about the goodness of the initial segments of this lattice rule. While $X_{10,20,d}(\boldsymbol{z}^\star)$ is bounded by 1.45 for the good embedded lattice rule, for the fixed lattice rule, the corresponding ratio is 35 times worse than the optimal fixed lattice rule for $m = 12$. Once again there is no direct ordering in these curves, but it is obvious that the curves corresponding to higher m are lower on the graph. For $m = 20$ we have a horizontal line at 1, and the line just above this constant line is the one for $m = 19$. It is not surprising that using half of the points of a good fixed lattice rule would also give a reasonable rule. The maximum of all these curves is clearly very bad. We can draw a very clear conclusion from this graph: the embedded lattice rule constructed by Algorithm 2 works very well in comparison with a more simple-minded strategy.

6.2. Pricing an Asian option. We test the performance of the embedded lattice rules constructed in this paper for calculating the pricing of an Asian call option. This problem is extensively documented, both in the financial literature and in the QMC literature [9, 11, 2]. For an Asian option (also called average option) the payoff is determined by the average value of the underlier (i.e., a stock price) during its lifetime. We will consider a (European style) Asian option with an arithmetic average of d equally spaced stock prices S_j at times $t_j = jT/d$, with expiry time T.

The payoff for such a call option can be written as

$$C(T) \;=\; \max\left(\frac{1}{d}\sum_{j=1}^{d} S_j - K, 0\right) \;=\; \left(\frac{1}{d}\sum_{j=1}^{d} S_j - K\right)^{+},$$

where K is the strike price at the expiry time $T = t_d$. We assume the risk-neutral measure of the Black–Scholes model for the path of the stock prices

$$S_j \;=\; S_0 \exp\left((r - \sigma^2/2)\, t_j + \sigma w_j\right),$$

where S_0 is the initial stock price, r is the risk-free interest rate, σ the volatility of the stock price, and w_j are samples of a Brownian motion at times t_j, i.e., w_j is normally distributed with variance t_j, or $w_j \sim \mathcal{N}(0, t_j)$. This sampled Brownian motion has mean 0 and covariance matrix $\boldsymbol{\Sigma} = [\min(t_i, t_j)]_{i,j=1,\ldots,d}$.

The value of the Asian option at time 0 can then be written as the actualized expected payoff at time T,

$$\begin{aligned} C(0) \;&=\; \mathbb{E}\left(\exp(-rT)\, C(T)\right) \\ &=\; \exp(-rT) \int_{\mathbb{R}^d} \left(\frac{1}{d}\sum_{j=1}^{d} S_0 \exp\left((r - \sigma^2/2)\, t_j + \sigma w_j\right) - K\right)^{+} \mathcal{N}(\boldsymbol{w}; \mathbf{0}, \boldsymbol{\Sigma})\, \mathrm{d}\boldsymbol{w}\,. \end{aligned}$$

We transform this integral to $[0,1]^d$ by using the inverse of the cumulative normal distribution function, Φ^{-1}, and deal with the correlation in our sample points by using a factorization of the covariance matrix, $\boldsymbol{\Sigma} = \boldsymbol{A}^{\top}\boldsymbol{A}$. Then our lattice rule approximation to $C(0)$ is given by

$$\widehat{C}(0) \;=\; \frac{\exp(-rT)}{n} \sum_{k=0}^{n-1} \left(\frac{1}{d}\sum_{j=1}^{d} S_0 \exp\left((r - \sigma^2/2)\, t_j + \sigma w_{k,j}\right) - K\right)^{+},$$

where $\boldsymbol{w}_k = \boldsymbol{A}\, \Phi^{-1}(\boldsymbol{t}_k)$ with lattice point $\boldsymbol{t}_k$ for $k = 0, 1, \ldots, n-1$.

For our particular example we choose the following parameters:

$$S_0 = 100\,, \qquad r = 0.1\,, \qquad \sigma = 0.2\,, \qquad T = 1\,, \qquad K = 100\,, \qquad d = 100\,.$$

We choose the principal component analysis method to factorize the matrix $\boldsymbol{\Sigma}$; that is, $\boldsymbol{A} = [\sqrt{\lambda_1}\boldsymbol{v}_1, \ldots, \sqrt{\lambda_d}\boldsymbol{v}_d]$ where $\lambda_1 \geq \cdots \geq \lambda_d$ are the eigenvalues of $\boldsymbol{\Sigma}$ and $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_d$ are the corresponding unit-length eigenvectors.

In Table 6.2 we present the results obtained with order-2 weights using 10 random shifts. As a comparison we also include results obtained by the Monte Carlo method (where we also used 10 random shifts). In Figure 6.4 we plot the graphs of the standard errors obtained by adding one point at a time. (The values of n in both Table 6.2 and Figure 6.4 correspond to the number of points used from the sequence. Due to the use of 10 random shifts, the actual number of function evaluations in each case is $10n$.) The results for the order-2 rule clearly show a rate of convergence close to $\mathcal{O}(n^{-1})$ while the Monte Carlo error decays like $\mathcal{O}(n^{-1/2})$. Furthermore, the graph assures us that it is quite all right to use the embedded lattice rules as a sequence, since the standard error goes down nicely (with some minor zigzag behavior, as expected) as the number of points is increased one at a time.

TABLE 6.2
Results of the Asian option for Monte Carlo and the order-2 lattice rule.

n	Order-2 lattice rule		Monte Carlo	
	integral	standard error	integral	standard error
1024	7.11310	5.07e-03	7.13816	8.97e-02
2048	7.10685	1.77e-03	7.10988	5.98e-02
4096	7.10252	1.56e-03	7.12438	4.84e-02
8192	7.10261	6.47e-04	7.11989	2.54e-02
16384	7.10304	4.17e-04	7.12116	2.10e-02
32768	7.10324	2.34e-04	7.12404	1.49e-02
65536	7.10280	1.18e-04	7.11410	1.14e-02
131072	7.10293	4.70e-05	7.11427	9.58e-03
262144	7.10287	1.91e-05	7.10980	5.05e-03
524288	7.10285	1.69e-05	7.10636	5.00e-03
1048576	7.10285	8.68e-06	7.10709	1.69e-03

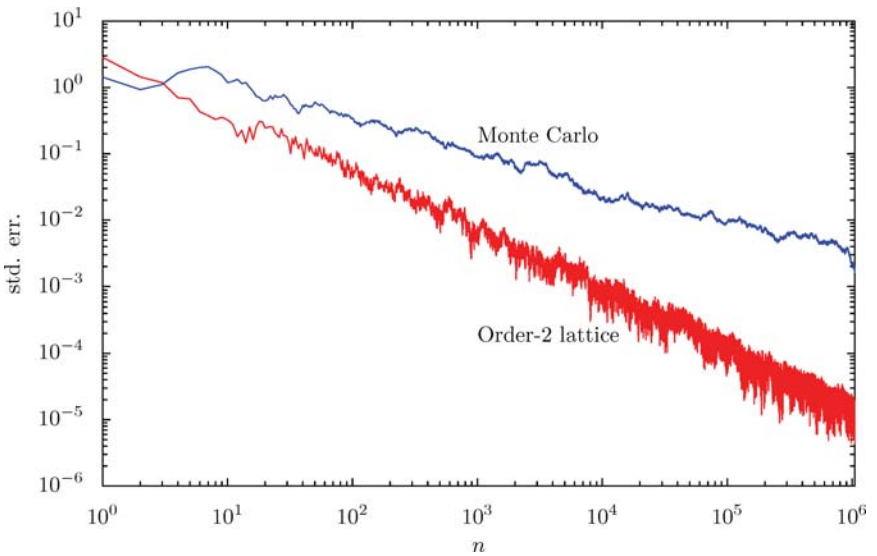


FIG. 6.4. *Standard errors of the Asian option for Monte Carlo and the order-2 lattice rule.*

7. Conclusion. We presented an algorithm for constructing good embedded lattice rules that are extensible in the dimension d and can be used for any number of points n in a predefined range. The algorithm utilized both the CBC construction principle and the fast implementation idea. For product weights and order-dependent weights, the total construction cost is $\mathcal{O}(dn(\log(n))^2)$ operations, with a linear memory dependence on n. Numerical experiments with the unanchored Sobolev space demonstrated that these rules achieve the same optimal rate of convergence $\mathcal{O}(n^{-1+\delta})$, $\delta > 0$, as those constructed for a fixed number of points, and that the implied constant gains only a factor of 1.30 to 1.55.

Appendix. Order-2 embedded lattice rule for $2^{10} \le n \le 2^{20}$.

d	z_d	d	z_d	d	z_d	d	z_d	d	z_d	d	z_d
1	1	61	390955	121	151423	181	455605	241	8601	301	83785
2	182667	62	202915	122	492123	182	227317	242	118219	302	11217
3	302247	63	230815	123	166279	183	251285	243	503827	303	163315
4	433461	64	304301	124	335461	184	295275	244	368993	304	293397
5	160317	65	452315	125	295093	185	476155	245	439439	305	386597
6	94461	66	383101	126	98821	186	291155	246	153211	306	338761
7	481331	67	70519	127	118331	187	294443	247	282303	307	88653
8	252345	68	346393	128	371123	188	31913	248	252999	308	337581
9	358305	69	291605	129	166909	189	518445	249	28617	309	230703
10	221771	70	397773	130	369553	190	481917	250	514327	310	140519
11	48157	71	92713	131	310959	191	326981	251	422201	311	421913
12	489023	72	391775	132	130595	192	488711	252	465011	312	416151
13	438503	73	131613	133	417025	193	447541	253	160907	313	197025
14	399693	74	74351	134	264103	194	39629	254	191687	314	350607
15	200585	75	382127	135	453203	195	394681	255	226205	315	262579
16	169833	76	219343	136	497985	196	379411	256	254941	316	510879
17	308325	77	297125	137	200509	197	335309	257	480375	317	451713
18	247437	78	88545	138	269743	198	93541	258	322857	318	8619
19	281713	79	89837	139	461919	199	188491	259	456621	319	50451
20	424209	80	191863	140	45927	200	152371	260	446175	320	193793
21	244841	81	506647	141	394663	201	408829	261	109049	321	155739
22	205461	82	441649	142	299155	202	217957	262	285305	322	23611
23	336811	83	240063	143	81299	203	65145	263	436731	323	437689
24	359375	84	239067	144	112403	204	462013	264	290763	324	100267
25	86263	85	310875	145	447473	205	220627	265	290439	325	439671
26	370621	86	211625	146	480325	206	368989	266	25363	326	211341
27	422443	87	147791	147	105053	207	273585	267	480371	327	22951
28	284811	88	370849	148	328455	208	373297	268	153337	328	73405
29	231547	89	149445	149	513239	209	285793	269	406899	329	220037
30	360239	90	340329	150	322199	210	405807	270	90863	330	169733
31	505287	91	493031	151	47537	211	63693	271	78537	331	408633
32	355195	92	336897	152	485183	212	382573	272	358757	332	64171
33	52937	93	202595	153	490687	213	84291	273	69087	333	44141
34	344561	94	518247	154	214311	214	444801	274	431749	334	122149
35	286935	95	369599	155	61871	215	226471	275	384083	335	258491
36	312429	96	50453	156	359761	216	166195	276	472419	336	215021
37	513879	97	133655	157	509981	217	170689	277	298375	337	11507
38	171905	98	395941	158	192829	218	368423	278	291499	338	295341
39	50603	99	13871	159	17075	219	509169	279	187231	339	41981
40	441451	100	248123	160	486463	220	243221	280	6967	340	378867
41	164379	101	380725	161	463461	221	191447	281	338357	341	261545
42	139609	102	314879	162	438237	222	200867	282	411965	342	56423
43	371213	103	106903	163	444353	223	194633	283	447817	343	445605
44	152351	104	94505	164	381279	224	226469	284	135463	344	362017
45	138607	105	499197	165	425405	225	413665	285	156061	345	504897
46	441127	106	479671	166	179723	226	148007	286	356943	346	335763
47	157037	107	76553	167	203897	227	284505	287	224431	347	417407
48	510073	108	351609	168	300411	228	459795	288	452203	348	58033
49	281681	109	13815	169	35087	229	324557	289	284195	349	80239
50	380297	110	342069	170	501519	230	422149	290	489071	350	372353
51	208143	111	446599	171	172275	231	368087	291	299537	351	116163
52	497641	112	374429	172	278507	232	166133	292	313173	352	287617
53	482925	113	342011	173	473739	233	401441	293	436265	353	483795
54	233389	114	8365	174	141429	234	44077	294	316447	354	310473
55	238553	115	424079	175	117025	235	457535	295	8353	355	75779
56	121499	116	184381	176	516465	236	122611	296	512723	356	141943
57	137783	117	203853	177	204743	237	27489	297	522699	357	418435
58	463115	118	7937	178	505099	238	392389	298	453557	358	305609
59	168681	119	302301	179	359937	239	327231	299	512829	359	152901
60	70699	120	444747	180	359503	240	87841	300	214315	360	129869

Acknowledgment. The authors thank Professor Ian Sloan for many helpful comments and rewarding discussions.

REFERENCES

[1] M. Abramowitz and I. A. Stegun, eds., *Handbook of Mathematical Functions with Formulas, Graphs and Mathematical Tables*, National Bureau of Standards Applied Mathematics Series 55, U.S. Government Printing Office, Washington, D.C., 1964.

[2] P. P. Boyle, Y. Lai, and K. S. Tan, *Pricing options using lattice rules*, N. Am. Actuar. J., 9 (2005), pp. 50–76.

[3] R. E. Caflisch, W. Morokoff, and A. Owen, *Valuation of mortgage backed securities using Brownian bridges to reduce effective dimension*, J. Comput. Finance, 1 (1997), pp. 27–46.

[4] J. Dick, *On the convergence rate of the component-by-component construction of good lattice rules*, J. Complexity, 20 (2004), pp. 493–522.

[5] J. Dick, I. H. Sloan, X. Wang, and H. Woźniakowski, *Good lattice rules in weighted Korobov spaces with general weights*, Numer. Math., 103 (2006), pp. 63–97.

[6] F. J. Hickernell, *Lattice rules: How well do they measure up?*, in Random and Quasi-Random Point Sets, P. Hellekalek and G. Larcher, eds., Springer-Verlag, New York, 1998, pp. 109–166.

[7] F. J. Hickernell, H. S. Hong, P. L'Écuyer, and C. Lemieux, *Extensible lattice sequences for quasi-Monte Carlo quadrature*, SIAM J. Sci. Comput., 22 (2000), pp. 1117–1138.

[8] F. J. Hickernell and H. Niederreiter, *The existence of good extensible rank-1 lattices*, J. Complexity, 19 (2003), pp. 286–300.

[9] C. Joy, P. P. Boyle, and K. S. Tan, *Quasi-Monte Carlo methods in numerical finance*, Management Science, 42 (1996), pp. 926–938.

[10] F. Y. Kuo, *Component-by-component constructions achieve the optimal rate of convergence for multivariate integration in weighted Korobov and Sobolev spaces*, J. Complexity, 19 (2003), pp. 301–320.

[11] C. Lemieux and P. L'Écuyer, *Efficiency improvement by lattice rules for pricing Asian options*, in Proceedings of the 30th Conference on Winter Simulation, D. J. Medeiros, E. F. Watson, J. Carson, and M. Manivannan, eds., IEEE Computer Society, Los Alamitos, CA, 1998, pp. 579–586.

[12] H. Niederreiter, *Random Number Generation and Quasi-Monte Carlo Methods*, CBMS-NSF Regional Conf. Ser. in Appl. Math. 63, SIAM, Philadelphia, 1992.

[13] D. Nuyens and R. Cools, *Fast algorithms for component-by-component construction of rank-1 lattice rules in shift-invariant reproducing kernel Hilbert spaces*, Math. Comp., 75 (2006), pp. 903–920.

[14] D. Nuyens and R. Cools, *Fast component-by-component construction of rank-1 lattice rules with a non-prime number of points*, J. Complexity, 22 (2006), pp. 4–28.

[15] C. M. Rader, *Discrete Fourier transforms when the number of data samples is prime*, Proc. IEEE, 5 (1968), pp. 1107–1108.

[16] I. H. Sloan, F. Y. Kuo, and S. Joe, *Constructing randomly shifted lattice rules in weighted Sobolev spaces*, SIAM J. Numer. Anal., 40 (2002), pp. 1650–1665.

[17] I. H. Sloan, X. Wang, and H. Woźniakowski, *Finite-order weights imply tractability of multivariate integration*, J. Complexity, 20 (2004), pp. 46–74.

[18] C. F. Van Loan, *Computational Frameworks for the Fast Fourier Transform*, Frontiers Appl. Math. 10, SIAM, Philadelphia, 1992.

[19] X. Wang and K. T. Fang, *The effective dimension and quasi-Monte Carlo integration*, J. Complexity, 19 (2003), pp. 101–124.

SIAM J. SCI. COMPUT.
Vol. 28, No. 6, pp. 2189–2207
© 2006 Society for Industrial and Applied Mathematics

MODELING WATER TRANSPORT ACROSS ELASTIC BOUNDARIES USING AN EXPLICIT JUMP METHOD*

ANITA T. LAYTON†

Abstract. A mathematical model is presented to simulate water and solute transport in a highly viscous fluid with a water-permeable, elastic immersed membrane. In this model, fluid motion is described by Stokes flow, whereas water fluxes across the membrane are driven by transmural pressure and solute concentration differences. The elastic forces, arising from the membrane being distorted from its relaxed configuration, and the transmembrane water fluxes introduce into model solutions discontinuities across the membrane. Such discontinuities are faithfully captured using a second-order explicit jump method [A. Mayo, *SIAM J. Numer. Anal.*, 21 (1984), pp. 285–299], in which jumps in the solution and its derivatives are incorporated into a finite-difference scheme. Numerical results suggest that the method exhibits desirable volume accuracy and mass conservation.

Key words. explicit jump, finite difference, immersed boundary, mass conservation, Stokes flow

AMS subject classifications. 35R05, 65M12, 65N06

DOI. 10.1137/050642198

1. Introduction.

Substantial effort has been directed to developing computational techniques for simulating moving boundaries or interfaces within a viscous fluid domain. Perhaps the most notable example is the immersed boundary method, developed by Peskin [27] for solving the full incompressible Navier–Stokes equations with moving boundaries. The immersed boundary method was originally developed for studying blood flow through a beating heart [26], but has since been applied to a wide variety of problems, including inner ear fluid dynamics [3], bacterial swimming [9], sperm motility in the presence of boundaries [12], ciliary beating [4, 10], and flow through an arteriole [1].

In this paper, we consider a water-permeable but solute-impermeable membrane immersed in a highly viscous fluid. Our goal is to describe a model and a computational technique that accurately simulates the fluid motion, membrane motion, and solute concentration. In the limit of zero Reynolds number, the inertial acceleration and advection terms in the Navier–Stokes equations can be neglected, and the resulting fluid motion can be described using the simpler Stokes equations. Despite their simplicity relative to the full Navier–Stokes equations, the Stokes equations have many applications. Indeed, the Stokes equations with an immersed membrane have been used to simulate the deformation and cleavage of fluid droplets [22] and to simulate biofilms [6].

The method we use to track fluid motion and solute transport is based on a method developed by Mayo for elliptic problems on an irregular domain [24, 25]. Mayo's method is similar to the immersed interface method later developed by LeVeque and Li [18, 19]; see [20] for a review of the immersed interface method. Both methods are second-order Cartesian grid methods (though higher-order immersed interface methods have recently been developed [14, 23]), with the key idea being the incorporation of known jumps in the solution or its derivatives into the finite-difference

*Received by the editors October 8, 2005; accepted for publication (in revised form) June 12, 2006; published electronically December 11, 2006. This work was supported in part by the National Science Foundation through grant DMS-0340654.
http://www.siam.org/journals/sisc/28-6/64219.html
†Department of Mathematics, Duke University, Durham, NC 27708 (alayton@math.duke.edu).

schemes. Following Wiegmann and Bube [31], we will refer to the method used in this study as an "explicit jump method."

The immersed boundary or immersed interfacial problems previously investigated have assumed a water-impermeable boundary or membrane, i.e., one that does not allow water movement across. In such models, fluid velocity is continuous across the membrane, and the membrane moves at the same velocity as the local fluid. In contrast, the membrane considered in this work is assumed to be permeable to water but not to solute; water fluxes are assumed to be driven by transmural pressure and solute concentration differences. Water permeability of the membrane is assumed to be finite. As a result of water movement across the membrane, the fluid volume enclosed by the membrane may evolve over time. In addition, transmural water fluxes induce corresponding changes in local solute concentration. A principal goal of this work is to accurately compute these model variables.

2. Model formulation. The model represents an evolving, closed membrane Γ that is assumed to be water permeable. Γ, which we will refer to interchangeably as a membrane or boundary, is assumed to have zero thickness. Γ separates the fluid domain Ω into two subdomains Ω^+ and Ω^-, where Ω^- is the region enclosed by membrane, whereas Ω^+ lies outside. Thus, $\Omega = \Omega^+ + \Omega^- + \Gamma$. The membrane Γ is assumed to be elastic, so that when it is distorted from its relaxed state by stretching or compressing, it exerts a force on the local fluid.

One solute is represented throughout Ω, and the membrane Γ is assumed to be impermeable to the solute. We assume a dilute solution in which the volume occupied by the solute is negligible.

We assume that the viscosity of the fluid is sufficiently high that its dynamics can be modeled as a creeping flow using the Stokes equations. In two dimensions, the Stokes equations take the form

$$p_x(x,y,t) = \mu(u_{xx}(x,y,t) + u_{yy}(x,y,t)) + F_1(x,y,t), \tag{2.1}$$

$$p_y(x,y,t) = \mu(v_{xx}(x,y,t) + v_{yy}(x,y,t)) + F_2(x,y,t), \tag{2.2}$$

$$u_x(x,y,t) + v_y(x,y,t) = 0, \tag{2.3}$$

where u and v are the fluid velocities in the x and y directions, respectively; p is the pressure; $\mu > 0$ is the fluid viscosity, assumed constant throughout Ω; and F_1 and F_2 are the singular force components in the x and y directions, respectively, and are supported only along Γ (see below). Biperiodic boundary condition is assumed for Ω.

The forces F_1 and F_2 are elastic or tension forces that arise from Γ being stretched or compressed. These singular forces can be written as

$$F_i(\vec{x},t) = \int_0^{L_0} f_i(s,t)\delta(\vec{x} - \vec{X}(s,t))ds \tag{2.4}$$

for $i = 1, 2$. L_0 is the length of Γ at its relaxed state; $\vec{x} \in \Omega$ and $\vec{X}(s,t)$ gives the location of Γ at time t, parameterized by the arclength s at the unstretched state for $0 \le s \le L_0$; f_i is the force strength at this point; and δ is the two-dimensional delta function. The force strength f is given by

$$\vec{f}(s,t) = \frac{\partial}{\partial s}\left(T(s,t)\vec{\tau}(s,t)\right), \tag{2.5}$$

where the tension $T(s,t)$ is given by

$$T(s,t) = T_0 \left(\left| \frac{\partial \vec{X}}{\partial s} \right| - 1 \right). \tag{2.6}$$

The tension coefficient T_0 describes the elastic properties of the membrane and is assumed to be a constant in this model. The vector tangential to Γ is given by $\vec{\tau}(s,t)$, where

$$\vec{\tau}(s,t) = \frac{\partial \vec{X}/\partial s}{\left| \partial \vec{X}/\partial s \right|}. \tag{2.7}$$

Thus, the force density can be computed directly from the location $\vec{X}$ of the membrane Γ. Note that at the relaxed state $\partial \vec{X}/\partial s = 1$ and tension vanishes.

A water-impermeable membrane moves at the velocity of the local fluid. That is, if $\vec{X}(s,t)$ denotes the position of Γ at time t at s, then

$$\frac{\partial}{\partial t} \vec{X}(s,t) = \vec{u}(\vec{X}(s,t),t). \tag{2.8}$$

With a water-permeable membrane, some water is allowed to pass through. To handle this case, let $g(s,t)$ denote transmural water transport in the direction normal to Γ at s, assumed positive for outward-directed fluxes. The water flux g can be driven by hydraulic pressure difference or concentration difference across Γ. The velocity of the membrane is then denoted by $\vec{U} \equiv (U,V)$, which is the difference of the local fluid velocity and the water flux:

$$U(\vec{X}(s,t),t) = u(\vec{X}(s,t),t) + g(s,t)\cos\theta, \tag{2.9}$$

$$V(\vec{X}(s,t),t) = v(\vec{X}(s,t),t) - g(s,t)\sin\theta, \tag{2.10}$$

where θ denotes the angle between the tangent line and the x-axis.

Advection and diffusion of the solute is coupled to the fluid velocity (u,v). The advection-diffusion equation is given for a solute of concentration $c(x,y,t)$ by

$$c_t + Gc + c_x u^* + c_y v^* = \nu(c_{xx} + c_{yy}), \tag{2.11}$$

where $\nu > 0$ is the solute diffusivity, assumed constant throughout Ω. We have suppressed in (2.11) the x, y, and t dependence for brevity. The advective velocity (u^*, v^*) is equal to the fluid velocity except at Γ, where it is set to the membrane velocity, i.e.,

$$u^*(\vec{x},t) = u(\vec{x},t), \quad v^*(\vec{x},t) = v(\vec{x},t), \quad \vec{x} \notin \Gamma, \tag{2.12}$$

$$u^*(\vec{x},t) = U(\vec{x},t), \quad v^*(\vec{x},t) = V(\vec{x},t), \quad \vec{x} \in \Gamma. \tag{2.13}$$

The second term on the left of (2.11) represents the changes in c induced by transmural water fluxes, with G given by

$$G(\vec{x},t) = \int_0^{L_0} g(s,t)\delta\left(\vec{x} - \vec{X}^+(s,t)\right) ds - \int_0^{L_0} g(s,t)\delta\left(\vec{x} - \vec{X}^-(s,t)\right) ds. \tag{2.14}$$

Note that G is zero everywhere except along the two sides of Γ, where it is singular. G can be considered as a pair of fluid source and sink, which have equal magnitude

(but opposite directions or signs), and which are singularly supported along each side of the membrane.

Because the membrane is assumed impermeable to the solute, the total amount of solute, given by $\int_{\Omega^\pm} c(\vec{x},t)d\vec{x}$, remains constant inside and outside the membrane for all time t. As water is driven across the membrane, the influx or efflux of water induces a local change in c near the membrane. For example, on the side of Γ where $g < 0$ (i.e., water loss), the solute is locally concentrated. This increase in c would have been singularly supported along Γ if ν were set to 0. In the case where $\nu > 0$, the change in c is finite and affects the entire subdomain (Ω^+ or Ω^-); see below.

3. Numerical method. The model equations (2.1), (2.2), (2.3), and (2.11) together describe the dynamics of the fluid, the position of the membrane, and the solute concentration as functions of time. In this section we present numerical procedures that, given a set of initial conditions, advance the system in time until a steady state is reached.

In the numerical discretization, let $\Delta t > 0$ denote the time step and let $t_n \equiv n\Delta t$ be the nth time-level for $n = 0, 1, \ldots$. Let h denote the spatial grid interval, and let $x_i \equiv ih$ and $y_j \equiv jh$ be the ith and jth spatial grid points along the x- and y-axes, respectively, where $i, j = 0, 1, 2, \ldots, N$. Let $\psi(x_i, y_j, t_n)$ be denoted by $\psi^n_{i,j}$ for an arbitrary function $\psi(x, y, t)$. Owing to the assumption of biperiodic boundary conditions, $\psi(x_0, y_j, t_n) = \psi(x_N, y_j, t_n)$ and $\psi(x_i, y_0, t_n) = \psi(x_i, y_N, t_n)$, and analogous equality applied to the derivatives of ψ.

The position of the membrane at t_n is represented by a set of boundary markers (X^n_k, Y^n_k) for $k = 0, 1, 2, \ldots, N_k$, where $(X^n_0, Y^n_0) = (X^n_{N_k}, Y^n_{N_k})$ because the membrane is assumed to be a simple closed curve. The kth boundary marker approximates $(X(s_k, t_n), Y(s_k, t_n))$, where $s_k = kL_0/N_k$; that is, the boundary markers are chosen such that in an unstretched or relaxed state, they are equally spaced. Let the value of an arbitrary function ψ on each side of Γ be denoted by $\psi_{b^\pm}$, i.e., $\psi_{b^\pm} = \lim_{\epsilon \to 0^+} \psi(\vec{X} \pm \epsilon\vec{n}, t)$, where $\vec{n}$ is the unit vector normal to the membrane. Let the value of ψ at the kth boundary marker (X^n_k, Y^n_k) be denoted by ψ^n_k.

3.1. Jump conditions. As in many immersed boundary problems, the solution of our model is unsmooth. Unless special care is taken, these discontinuities tend to introduce substantial inaccuracy into the computed solution obtained by means of a finite-difference method. One remedy for achieving better accuracy is the approach used in the immersed boundary method and its variants, in which the Dirac delta function in (2.4) and (2.14) is replaced by an approximate or discrete delta function that is smooth and compact, thereby eliminating the singularity in the forces [27]. However, by distributing the force terms to nearby grid points, the discrete delta function approach approximates the discontinuous solutions as continuous functions with large gradients, and thus does not faithfully represent the jump discontinuity in solutions. In our model, this artificial smoothing is particularly problematic for pressure, inasmuch as its jump discontinuity is required in the computation of transmural water fluxes (see the numerical examples in section 4). Thus, we adopt an alternative approach—the explicit jump method based on Mayo [24]. That method is similar to the immersed interface method [20], both of which generate second-order accuracy in space.

An explicit jump method requires that the jump discontinuities in the solution and its derivatives be specified along the membrane. Those jump conditions are specified below. To compute the fluid motion given by (2.1), (2.2), and (2.3), we follow the approach by LeVeque and Li [19], in which (2.1), (2.2), and (2.3) are reduced to a

sequence of three Poisson problems, one for each variable. For the pressure, one solves

$$\nabla^2 p = \nabla \cdot \vec{F}. \tag{3.1}$$

The derivative of the singular force term $\vec{F}$, a dipole source, in the Poisson equation (3.1) renders p discontinuous, along with its derivatives. To define these jump conditions, we let the net jump discontinuity of a function ψ be

$$[\psi(\vec{X},t)] = \lim_{\epsilon \to 0^+} \psi(\vec{X}^+ + \epsilon\vec{n},t) - \lim_{\epsilon \to 0^+} \psi(\vec{X}^- - \epsilon\vec{n},t). \tag{3.2}$$

With this notation, the jump conditions for pressure are given by

$$[p(s,t)] = f_n(s,t), \quad [p_n(s,t)] = \frac{\partial}{\partial s} f_\tau(s,t), \tag{3.3}$$

where f_n and f_τ denote the normal and tangential components of $\vec{f}$, respectively. The derivation of (3.3) can be found in [19].

Fluid velocity (u,v) is continuous across the membrane; the normal velocity component is continuous as the membrane moves at the same speed as the fluid, and the tangential component is also continuous owing to the presence of viscosity and the no-slip boundary conditions between the membrane and the fluid on each side. However, the normal derivatives of the velocity components may have jump discontinuities:

$$[u] = [v] = 0, \tag{3.4}$$

$$[\mu u_n(s,t)] = f_\tau(s,t)\sin\theta, \quad [\mu v_n(s,t)] = -f_\tau(s,t)\cos\theta, \tag{3.5}$$

where θ denotes the angle between the tangent line and the x-axis. The derivation of (3.5) for a water-impermeable membrane can again be found in [19].

Furthermore, the water fluxes introduce a jump discontinuity in the concentration and its derivatives across the membrane. Unlike the jump conditions for p, u, and v, the jumps in c cannot be represented in terms of the membrane configuration or the elastic forces. Thus, they are estimated by applying extrapolation or one-sided finite-difference to grid-point c values. To approximate $[c]$, we first estimate c at boundary markers $\vec{X}_k$ on the two sides of Γ using a one-sided interpolation,

$$c^{\mathrm{I}}_{k^\pm} = \sum_{i,j} \xi_{i,j} c_{i,j}, \tag{3.6}$$

and using neighboring grid-point values $c_{i,j}$ such that the $x_{i,j}$ all reside on the same subdomain $\Omega^\pm$ and such that the interpolation yields second-order accuracy. The jump $[c_k]$ is then given by

$$[c_k] = c^{\mathrm{I}}_{k^+} - c^{\mathrm{I}}_{k^-}. \tag{3.7}$$

The correction terms needed to obtain a second-order approximation of the derivatives of c, using a finite-difference stencil that crosses the membrane, involve $[c']$ and $[c'']$ (see (A.8) and (A.10)). In most cases, these jumps can be estimated using one-sided finite difference. However, along portions of the membrane one may not find a sufficient number of grid points on one side of Γ to form the one-sided finite-difference stencil. One such example is shown in Figure 3.1. At (x^*, y_j) where Γ crosses the $y = y_j$ line, c and its derivatives are discontinuous. These jumps are needed for a

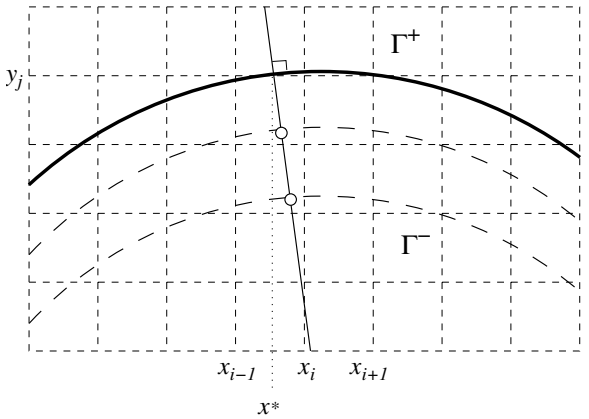


FIG. 3.1. *An example scenario where only one grid point (x_i, y_j) can be found along $y = y_j$ in Γ^-. c_n is approximated at (x^*, y_j) using one-sided finite difference, and using interpolated values (indicated by open circles) and boundary value along the normal line.*

finite-difference approximation of c_x or c_{xx} at (x_i, y_j) to yield high-order accuracy. However, in this example the lone grid point along $y = y_j$ that also lies in Γ^- is (x_i, y_j), and that is not enough to form a second-order, one-sided finite-difference approximation of c_x at (x^*, y_j). One solution is to increase grid resolution around Γ, an approach that was used by Linnick and Fasel [23]. We adopt an alternative approach where, instead of estimating $[c_x]$ and $[c_{xx}]$, we estimate $[c_n]$. To that end, we first approximate c values along normal lines to the membrane at $\vec{X}^{\pm}{}_k^{m+1} \pm lh$, where $l = 0, 1, 2$, by interpolating nearby grid-point c values on the same subdomain $\Omega^{\pm}$. Then we compute $[c_n]$ by estimating c_n at $\vec{X}_k^{\pm}$ using one-sided finite difference (denoted by ${c_n^{\mathrm{FD}}}_{k^{\pm}}$), and then taking the difference; i.e.,

$$[c_{n\,k}] = {c_n^{\mathrm{FD}}}_{k^+} - {c_n^{\mathrm{FD}}}_{k^-}. \tag{3.8}$$

The normal jump at any point $\vec{X}(s)$ along Γ can be obtained by interpolating $[c_{n\,k}]$ values.

The correction terms for the second-order finite-difference operator involve also arclength derivatives of the jumps in functions and in the normal derivative (e.g., $\partial[p]/\partial s$ and $\partial[p_n]/\partial s$, where s denotes the arclength; see the appendix). Those derivatives are obtained by applying second-order finite differencing along s to the values of the jumps at boundary markers.

We advance the solution of the system in time in sequential steps. First, we compute the fluid pressure and velocity by solving (2.1), (2.2), and (2.3). Then we update the solute concentration in a second step using known fluid velocity and water fluxes. The method generates a solution that is first order in time and second order in space.

3.2. Computing fluid and boundary motion. As in [19], we compute p by eliminating the singular source terms from (3.1) to yield $\nabla^2 p = 0$, which we then solve together with the jump conditions specified across the interface (3.3) using the technique of Mayo [24]. Specifically, we solve a difference equation of the form

$$\frac{1}{h^2}\left(p_{i+1,j} + p_{i-1,j} + p_{i,j+1} + p_{i,j-1} - 4p_{i,j}\right) = C_{i,j}. \tag{3.9}$$

The derivation for the correction terms $C_{i,j}$ is shown in the appendix. The Poisson equations are solved by means of fast Fourier transforms.

Once p is computed, (2.1) and (2.2) are independent Poisson problems for u and v, respectively. Neglecting the singular terms as we did for the Poisson problem for p, we obtain

$$\nabla^2 u = \frac{1}{\mu} p_x, \quad \nabla^2 v = \frac{1}{\mu} p_y. \tag{3.10}$$

Equations (3.10) are solved again using the explicit jump method with the jump conditions (3.4) and (3.5), together with the jumps in p_x and p_y, incorporated. The derivatives p_x and p_y are computed using a second-order finite-difference scheme, with the jump conditions (3.3) taken into account, an approach that has been shown to yield second-order accuracy for solution gradient [2].

The position $\vec{X}$ of Γ is determined by the location of a set of boundary markers along Γ; and at a given time t, Γ is represented by a cubic spline through those markers [19]. To advance the membrane by one time step, we move the boundary markers according to

$$\frac{\partial}{\partial t} \vec{X}(x,t) = \vec{U}(\vec{X}(s,t),t), \tag{3.11}$$

where $\vec{U}$ is given in (2.9) and (2.10). To compute $\vec{U}$, the local fluid velocity is determined by interpolating the velocity fields $\vec{u}_{ij}$ at grid points to the kth marker to yield $\vec{u}_k^I$, with care taken to incorporate jumps in their normal derivatives into the interpolation formulae. Then $\vec{U}$ is computed by incorporating g using (2.9) and (2.10), and forward Euler is used to advance the kth marker $\vec{X}_k$ by one time step.

3.3. Computing solute concentration. Once we have obtained the fluid pressure p^{m+1}, velocity $\vec{u}^{m+1}$, and membrane position $\vec{X}^{m+1}$, we proceed to update the solute concentration c, using the explicit jump method and a time-split method as described below. Our strategy is to (1) compute provisional grid-point values $c_{i,j}$ while neglecting transmural water fluxes; (2) compute boundary concentration $c_{b^\pm}$; and (3) incorporate the effects of water fluxes into $c_{i,j}$.

Computing provisional grid-point concentration values $c_{i,j}$. We first consider concentration values at grid points $\vec{x}_{i,j} \notin \Gamma^\pm$. Away from the membrane, $G = 0$. Thus, (2.11) simplifies to

$$c_t = -(c_x u + c_y v) + \nu(c_{xx} + c_{yy}). \tag{3.12}$$

Equation (3.12) is integrated by an explicit update of the advection terms, and, to avoid a prohibitive restriction on Δt, an implicit update of diffusion. That is, we solve for $c_{i,j}^{m+1,*}$ at regular grid points the following elliptic equation:

$$(1 - \nu\nabla^2) c^{m+1,*} = c^m - \Delta t(u^m c_x^m + v^m c_y^m). \tag{3.13}$$

As is done for p and (u, v), (3.13) is solved by incorporating the jumps in c and its derivatives into a finite-difference scheme. However, the implicit treatment of the diffusion means that the jumps $[c^{m+1,*}]$ and $[c_n^{m+1,*}]$ are needed for the explicit jump method. Unlike the jump conditions for p (3.3) and for $\vec{u}$ (3.4)–(3.5), $[c]$ and $[c_n]$ cannot be expressed in terms of the membrane configuration, elastic forces, or the fluid dynamics variables that are already computed. Instead, jump conditions

for concentration can only be computed once c and its derivatives are known at the boundary. Thus, to advance c according to (3.12), we first approximate these jumps by computing a provisional solution $c^{m+1,**}$. To that end, we integrate (3.12) explicitly; i.e., we compute

$$c^{m+1,**} = c^m - \Delta t\big(u^m c_x^m + v^m c_y^m - \nu\nabla^2 c^m\big). \tag{3.14}$$

The corrected finite-difference approximations for c_x^m, c_y^m, and $\nabla^2 c^m$ require jumps at t_m, i.e., $[c^m]$ and $[c_n^m]$, which are known. Once $c^{m+1,**}$ has been computed, estimates for $[c^{m+1,**}]$ and $[c_n^{m+1,**}]$ can be made (see section 3.1). Setting $[c^{m+1,*}] = [c^{m+1,**}]$ and $[c_n^{m+1,*}] = [c_n^{m+1,**}]$, we then solve the implicit diffusion equation (3.13) by incorporating these jumps into the second-order finite difference scheme, a procedure that is analogous to the solution of the three Poisson problems for the fluid pressure and velocity.

Computing boundary concentration values $c_{b^\pm}$. On the immediate sides of the membrane (i.e., $\Gamma^\pm$), concentration is affected by water transport represented by the flux terms Gc in (2.11). To compute boundary concentration, we first rewrite the mass transport equation (2.11) in the Lagrangian form

$$\frac{dc_{b^\pm}}{dt} = \xi g\delta(\vec{0})c_{b^\pm} + \nu(c_{b^\pm xx} + c_{b^\pm yy}), \tag{3.15}$$

where $\xi = \pm 1$ for $\vec{x} \in \Gamma^\pm$, and where $d/dt \equiv \partial/\partial t + u\partial/\partial x + v\partial/\partial y$. As noted previously, there are two "layers" of c_b values, one on each side of Γ (i.e., $\Gamma^\pm$).

Equation (3.15) is integrated in time using a splitting approach as follows. We let

$$c_{b^\pm}^{m+1} = \bar{c}_{b^\pm}^{m+1} + \hat{c}_{b^\pm}^{m+1}, \tag{3.16}$$

where

$$\bar{c}_{b^\pm}^{m+1} = c_{b^\pm}^m + \Delta t\nu(\bar{c}_{b^\pm xx}^{m+1} + \bar{c}_{b^\pm yy}^{m+1}), \tag{3.17}$$

$$\hat{c}_{b^\pm}^{m+1} = \int_{t_m}^{t_{m+1}} \left(\xi g(t)\delta(\vec{0})c_{b^\pm}(t) + \nu(c_{b^\pm xx}(t) + c_{b^\pm yy}(t))\right)\, dt. \tag{3.18}$$

The $\bar{c}_{b^\pm}$ term describes the evolution of the concentration of a bolus that travels along $\Gamma^\pm$, starting with a concentration of $c_{b^\pm}^m$ at t_m and having undergone diffusion for one time step. The $\hat{c}_{b^\pm}$ term represents the effects of transmural water fluxes coupled with solute diffusion. The time-stepping used to obtain $\hat{c}_{b^\pm}^{m+1}$ is discussed below.

Note that $\bar{c}_{b^\pm}^{m+1} = c^{m+1,*}(\vec{X}^\pm)$, the provisional concentration obtained via the solution of (3.12). The advection terms in (3.13) are incorporated into the Lagrangian derivative in (3.15). Thus we approximate $\bar{c}_{b^\pm}^{m+1}$ from the grid-point values $c_{i,j}^{m+1,*}$ using one-sided extrapolation.

To compute $\hat{c}_{b^\pm}^{m+1}$, we employ a time-split approach and treat the Dirac delta and the diffusion operator in (3.18) sequentially. Recall that the solution to

$$\phi_t = \nu(\phi_{xx} + \phi_{yy}), \tag{3.19}$$

with initial condition $\phi(x, y, 0) = \delta(x, y)$, $x, y \in [0, \infty)$, reflective boundaries at $x = 0$ and at $y = 0$, is given by

$$\phi(x, y, t) = \frac{1}{2\pi\nu t} \exp\left(-\frac{x^2 + y^2}{4\nu t}\right). \tag{3.20}$$

We compute $\hat{c}_{b^\pm}^{m+1}$ as the Gaussian solution (3.20) generated by the application of the diffusion operator $\nu(c_{b^\pm xx} + c_{b^\pm yy})$ to a Dirac delta function $\Delta t \xi g \delta(\vec{0}) c_{b^\pm}^m$ for one time step Δt; i.e.,

$$\hat{c}_{b^\pm}^{m+1} = \frac{\Delta t \xi g c_{b^\pm}^m}{2\pi\nu\Delta t}. \tag{3.21}$$

Combining (3.21) with $\bar{c}_{b^\pm}^{m+1}$, we obtain

$$c_{b^\pm}^{m+1} = \bar{c}_{b^\pm}^{m+1} + \frac{\xi g c_{b^\pm}^m}{2\pi\nu}. \tag{3.22}$$

Incorporating water fluxes into grid-point values $c_{i,j}$. The provisional grid-point values $c_{i,j}^{m+1,*}$ do not include the effects of water fluxes, effects that must be incorporated to preserve mass. Indeed, the Gaussian (3.20) is global and nonzero for each subdomain $\Omega^\pm$, although it decays exponentially fast away from the membrane (especial for sufficiently small ν). The final step in updating c is to incorporate the effect of water fluxes, using (3.20), to the provisional solution $c_{i,j}^{m+1,*}$ at regular grid points. To that end,

$$c_{i,j}^{m+1} = c_{i,j}^{m+1,*} + \int_0^{L_o} \frac{\xi g c_{b^\pm}^m}{2\pi\nu} \exp\left(-\frac{(x_j - X^\pm(s))^2 + (y_j - Y^\pm(s))^2}{4\nu\Delta t} \right) ds, \tag{3.23}$$

where $\xi = \pm 1$ for $(x_i, y_j) \in \Omega^\pm$. The integral in (3.23) is approximated using the trapezoidal rule and the boundary marker values $\hat{c}_{b^\pm}^{m+1}$, i.e.,

$$c_{i,j}^{m+1} = c_{i,j}^{m+1,*} + \Delta s \sum_{l=1}^{N_k} \frac{\xi g c_{l^\pm}^m}{2\pi\nu} \exp\left(-\frac{(x_i - X_l^\pm)^2 + (y_j - Y_l^\pm)^2}{4\nu\Delta t} \right). \tag{3.24}$$

Evaluating (3.24) for every pair of grid point and boundary marker requires $\mathcal{O}(N^2 N_k)$ computational time, which likely dominates the overall calculations for a sufficiently refined spatial grid. Thus in practice the summation in (3.24) only includes terms involving grid-point–boundary-marker pairs that are sufficiently close, i.e., for (i, j, k) such that $(x_i - X_k^\pm)^2 + (y_j - Y_k^\pm)^2 < d$. The error introduced by ignoring the contributions from grid points outside of that neighborhood can be shown to be

$$E_d = \int_0^L \frac{\sqrt{\Delta t}|g(s)|(c_{b^+}^m(s) + c_{b^-}^m(s))}{\nu} \left(1 - \operatorname{erf}\left(\frac{d}{2\sqrt{2\pi\Delta t}} \right) \right) ds. \tag{3.25}$$

A smaller Δt or a larger ν increases E_d. Additional errors are present in (3.24) from discretizing the integral in (3.23) using the trapezoidal rule.

4. Numerical examples. In this section we present numerical results obtained for two examples. Our main focus is on the accuracy with which our method captures jumps in the solutions and on the mass-conserving properties of the method. All calculations reported below were performed using Fortran programs, which were executed in double precision on a computer with an Intel Pentium IV 3.2 GHz processor and 1 GB of RAM.

4.1. Hydraulic pressure-driven water fluxes. This example was used by Tu and Peskin to test their immersed boundary method [30], and by LeVeque and Li to test their immersed interface method [19]. The initial membrane is an ellipse

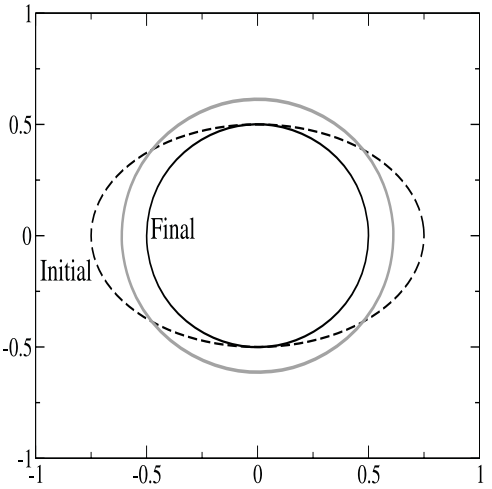


FIG. 4.1. *Membrane configuration at initial time* (*dotted line*) *and steady state* (*solid dark line*) *Gray line shows steady state if there were no water fluxes.*

with major and minor axes set to $a = 0.75$ and $b = 0.5$, respectively (dashed line in Figure 4.1). The unstretched membrane was set to a circle with radius $r_0 = 0.5$ (solid dark line in Figure 4.1). The computational domain was set to $[-1, 1] \times [-1, 1]$. Transmural water fluxes were assumed to be driven by hydraulic pressure [13, 29]

$$g(s,t) = -\sigma[p(s,t)], \tag{4.1}$$

where $\sigma > 0$ represents the water permeability of the membrane; in this example we set $\sigma = 0.01$. Thus, a higher pressure in the enclosed fluid, i.e., $[p] < 0$, will generate outward-directed water fluxes, i.e., $g > 0$, and vice versa. Solute concentration was initialized to $c(\vec{x}, 0) = 1$ in the enclosed fluid (i.e., for $\vec{x} \in \Omega^-$) and to $c(\vec{x}, 0) = 2$ in the outside fluid (i.e., for $\vec{x} \in \Omega^+$).

In the *absence* of transmembrane water fluxes (i.e., if $g(s,t)$ were set to 0), the initial ellipse will converge to a steady state with radius $r_e = \sqrt{ab} \approx 0.6124$, which is larger than for the unstretched membrane but which has the same area as the initial ellipse (gray line in Figure 4.1) owing to the incompressibility of the enclosed fluid. At steady state, p will attain constant values inside and outside the membrane, with a jump $[p] < 0$ (because the membrane was initialized to a stretched state); fluid velocity will vanish everywhere.

In the presence of transmural pressure-driven water fluxes, the higher pressure of the enclosed fluid continues to drive water outward, until the membrane reaches the unstretched position, at which time transmural pressure difference vanishes.

Thus in this case the steady-state membrane configuration is the unstretched one (solid dark line in Figure 4.1). Note that although the fluid is incompressible, the volume of the enclosed fluid is not conserved.

Because the membrane is assumed impermeable to the solute, the total amount of solute (given by $\int_{\Omega^\pm} c(\vec{x}, t) d\vec{x}$) remains unchanged in each of the subdomains Ω^+ and Ω^- for all time. Because the amount of fluid (or area) in each subdomain changes in time (though the total amount of fluid in Ω is constant owing to periodic boundary conditions), c changes. In this case, because the membrane is initialized to a stretched state, at steady state there is a net outward migration of water; thus, c increases in

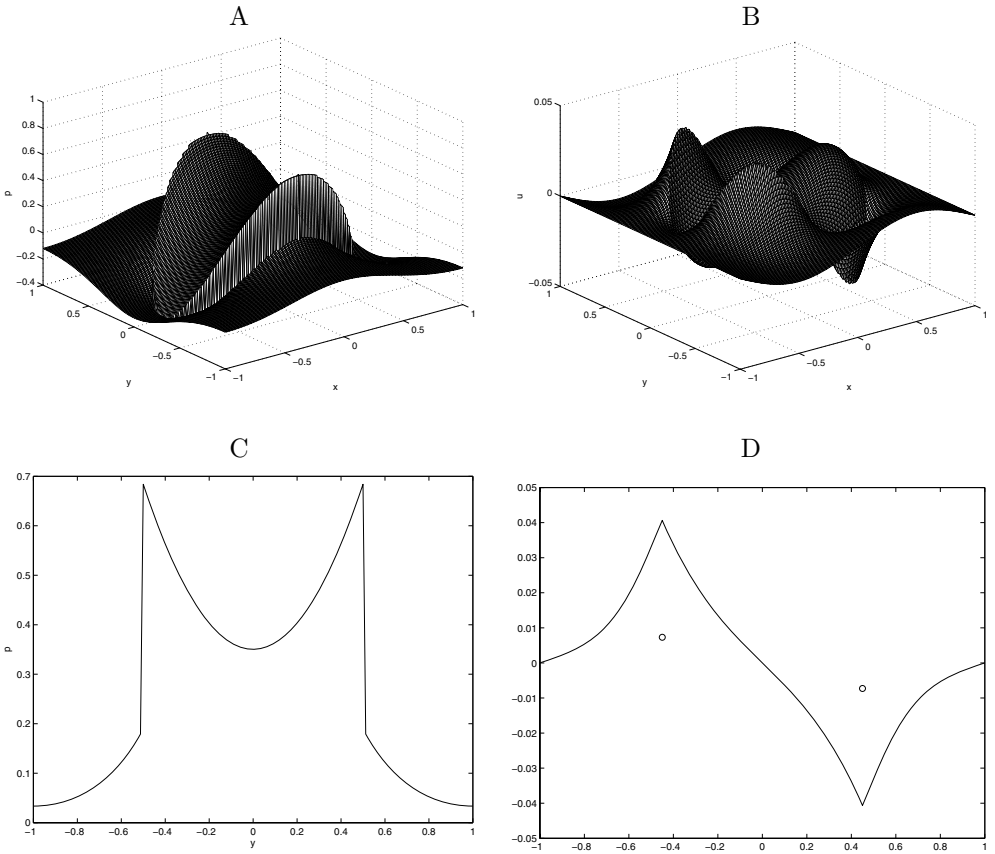


FIG. 4.2. A: *initial pressure field.* B: *initial x-component of velocity field* (u). C: *initial pressure at* $x = 0$. D: *initial* u *at* $y = -0.4$. *Open circles in panel* D *indicate membrane velocity.*

Ω^-. Let $t = \infty$ denote steady state. At steady state, c is constant in each of $\Omega^{\pm}$, with $c(\vec{x}, \infty) = (A^-(0)/A^-(\infty))c(\vec{x}, 0) = 1.5$, where $\vec{x} \in \Omega^-$, $c(\vec{x}, \infty)$ denotes steady-state concentration, and $A^-(0)$ and $A^-(\infty)$ denote initial and steady-state areas, respectively, enclosed by the membrane. An analogous calculation yields a steady-state outside concentration $c(\vec{x}, \infty) = 1.667$, $\vec{x} \in \Omega^+$.

We set the diffusion coefficients to $\mu = 1$ and $\nu = 0.001$. Note that the steady-state solution is independent of μ and ν, although the transient solution depends on their values. The tension coefficient T_0 was set to 0.5. To solve the Stokes equations (2.1), (2.2), and (2.3), we used a time step $\Delta t = 0.1$. Because at each time step the concentration jumps $[c^{m+1,*}]$ and $[c_n^{m+1,*}]$ were estimated using an explicit time-integration, a smaller time step was required to integrate the solute conservation equation (2.11) to maintain numerical stability. Nonetheless that time-step restriction is less than the restriction that would have been imposed by a purely explicit integration of diffusion. Thus, for each fluid step Δt, we took 20 concentration steps using a time step of $\Delta t/20$. During those 20 concentration time steps, p, u, and v were held constant. To obtain an approximate steady-state solution, model equations were integrated to $t = 1000$, at which time $p < 10^{-6}$ in Ω^- (at true steady state $p = 0$).

Given a membrane configuration, pressure and fluid velocity can be computed. Figure 4.2 shows p and u at $t = 0$, in two dimensions (panels A and B), along $x = 0$

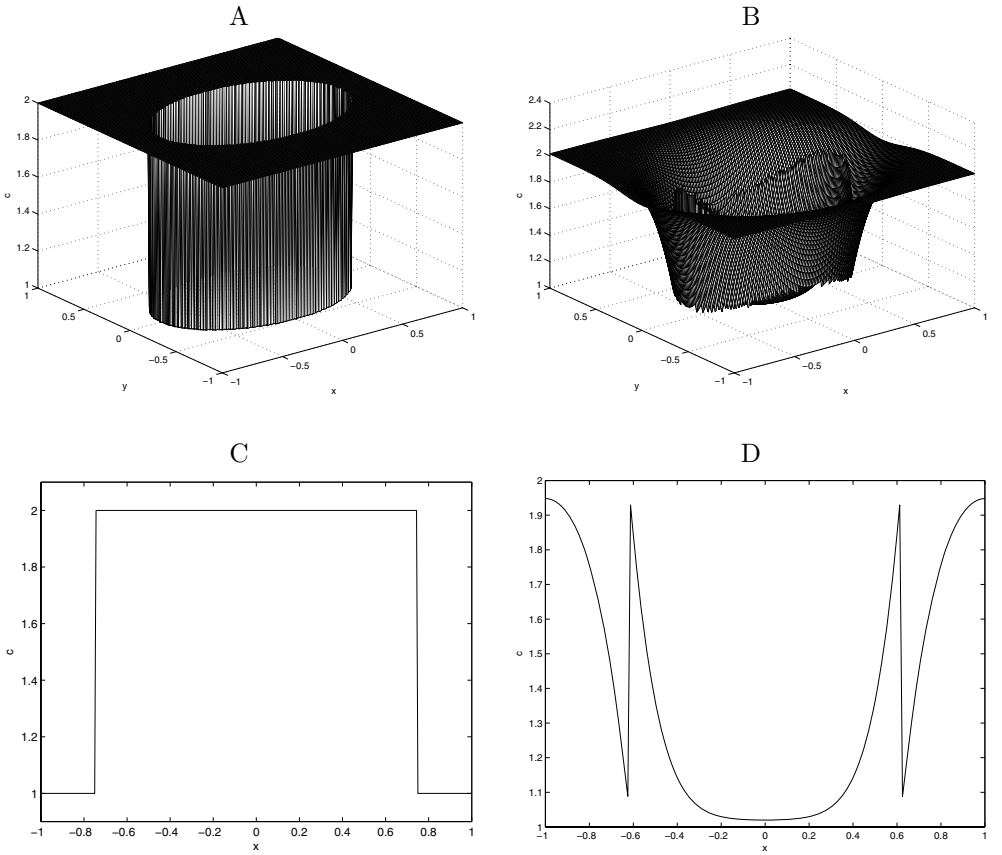


FIG. 4.3. A, C: *concentration at* $t = 0$. B, D: *concentration at* $t = 200$. C, D: *concentration along* $y = 0$.

(panel C), and along $y = -0.4$ (panel D). These results were obtained using $N = 160$. The singular normal elastic force renders p discontinuous across the membrane. In particular, p has jump discontinuities at $(0, \pm 0.5)$ at initial time, as shown in Figure 4.2C. The singularities of p at $\Gamma^{\pm}$, induced by G, are not shown. The net jump discontinuities in pressure and in the normal derivatives of the fluid velocity were captured sharply by the explicit jump method. The velocity at which the membrane moves is shown in Figure 4.2D as open circles.

Figure 4.3 shows solute concentration at $t = 0$ and 200. As previously noted, solute concentration was initialized to be a step function (see panels A and C). Along the portions of the membrane that is stretched (e.g., at $(x, y) = (\pm 0.75, 0)$), water is driven outward, introducing an increase in local concentration within the membrane, and a corresponding decrease on the opposite side of the membrane. Thus, after a sufficiently long integration, interior solute concentration exceeds that of the exterior. As can be noted in Figure 4.3D, the concentration discontinuities at $x = \pm 0.75$ (and elsewhere, not shown) are captured sharply by the explicit jump method. As the simulation proceeds further, interior and exterior concentrations converge to their respective constant values of 1.5 and 1.6667.

Table 4.1 shows volume accuracy and solute conservation results for $N = 80$, 160, 320, and 640. The same N was used to discretize the membrane, i.e., $N_k = N$. The

TABLE 4.1

Convergence results for an example with hydraulic pressure-driven water fluxes. $A^-(\infty)$, steady-state area enclosed by membrane; $M^-(\infty)$, steady-state total solute enclosed by membrane. The errors in $A^-(\infty)$ and $M^-(\infty)$, denoted by error$_{A-}$ and error$_{M-}$, were computed using exact solution $A^-(\infty) = M^-(\infty) = 0.7853981635$. Results show second-order spatial convergence.

N	$A^-(\infty)$	error$_{A-}$	$M^-(\infty)$	error$_{M-}$
80	0.7819365415	3.462E-3	0.5955737106	1.898E-1
160	0.7845370713	8.611E-4	0.7784843941	6.914E-3
320	0.7851768010	2.214E-4	0.7831280790	2.227E-3
640	0.7853426842	5.548E-5	0.7847976120	6.006E-4

steady-state enclosed volume $A^-(\infty)$ was obtained by approximating the integral $\int_{\Omega^-} d\vec{x}$ using a second-order integration method. That method approximates the area of the h^2 squares that the boundary cuts through as a triangle, a trapezoid, or a sum of the areas of a rectangle and a trapezoid, depending on whether one, two, or three of the four corners of the square lie in Ω^-. The amount of enclosed solute $M^-(\infty)$ was estimated analogously using the integral $\int_{\Omega^-} c(\vec{x}, \infty) d\vec{x}$. Until steady state was reached, there was a net efflux of water, driven by the higher interior pressure; thus, the volume enclosed by the membrane is not conserved. Results in Table 4.1 exhibit second-order convergence in space. Analogous volume accuracy and mass conservation results were also obtained for the exterior region Ω^+ and thus for the entire domain Ω. The errors are larger in solute conservation than in volume accuracy, especially for small $N \leq 80$. That difference may be attributable to the substantial solute concentration gradient near the boundary during transient states, which gave rise to significant extrapolation errors when boundary concentration was estimated from grid-point values, e.g., in the computation of $\bar{c}_b^{m+1}$. When the spatial grid failed to sufficiently resolve the concentration gradient near the boundary, the grid-point values $c_{i,j}$ used to extrapolate $\hat{c}_b^{m+1}$ were essentially flat and would thus yield an under- or overestimate, depending on whether the spatial gradient was positive or negative along $\vec{n}$.

4.2. Osmotic and hydraulic pressure-driven water fluxes. In the next example, we represent also water fluxes arising from transmural osmotic or concentration difference:

$$g(s,t) = -\sigma([p(s,t)] - [c(s,t)]). \tag{4.2}$$

At steady state, $g(s, \infty) = 0$, with the transmural pressure and concentration gradients in balance. The previous example in section 4.1, which represents hydraulic pressure-driven fluxes only, yielded a steady-state pressure that is constant everywhere (i.e., $[p] = 0$]) and a piecewise constant steady-state concentration with $[c] = 0.1667$. Had osmotic pressure-driven water fluxes (4.2) been included in the model, that transmural concentration difference would have given rise to an efflux of water, from the interior region of lower concentration to the exterior region of higher concentration. The loss of water would have reduced the interior area, compressed the elastic membrane, and generated an outward-directed tension force. The resulting negative pressure gradient $[p] < 0$ would then favor water fluxes into the interior region, an effect that opposes the concentration difference. Thus, a new steady state would be reached in which the transmural pressure and concentration gradients balance. At steady state, fluid velocity vanishes as before.

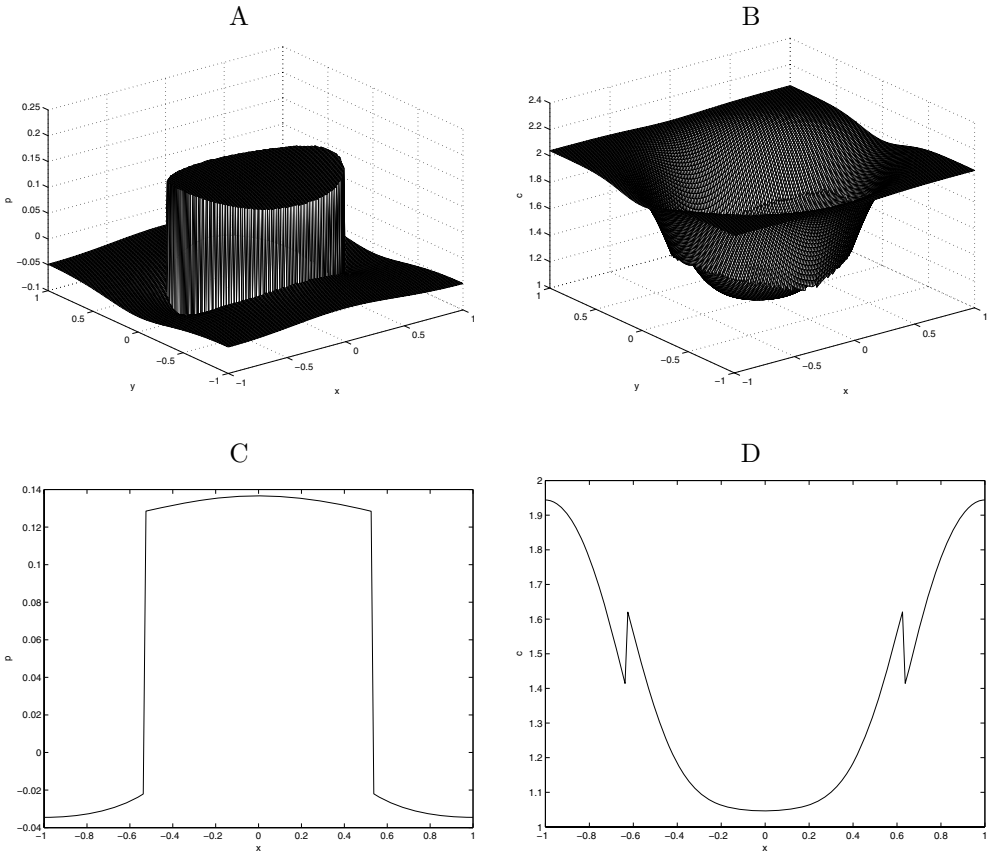


FIG. 4.4. *Pressure and concentration at* $t = 200$. C, D: *along* $y = 0$.

The steady-state balance of transmural pressure and concentration gradients allows us to obtain analytic solutions for steady-state concentrations and pressure:

$$c^+(\infty) - c^-(\infty) = p^+(\infty) - p^-(\infty). \tag{4.3}$$

Applying mass conservation, one obtains that

$$c^+(0)\frac{1-ab}{1-r(\infty)^2} - c^-(0)\frac{ab}{r(\infty)^2} = T_0(r(\infty) - r_0). \tag{4.4}$$

The initial membrane configuration, initial concentration, tension coefficient, membrane permeability, and diffusion coefficients were set to the same values used in the previous example. Using (4.4), we computed steady-state membrane configuration, pressure jump, and concentration to be $r(\infty) = 0.479205$, $[p(\infty)] = -0.0103975$, $c^+(\infty) = 1.622613$, and $c^-(\infty) = 1.633009$. Note that $[c(\infty)] = [p(\infty)]$.

Figure 4.4 shows pressure and solute concentration at $t = 200$, in two dimensions (panels A and B) and along $y = 0$ (panels C and D). Results were obtained for $N = 160$. The jump discontinuities in both pressure and concentration were reproduced sharply by the explicit jump method. Throughout the simulation concentration jump $[c]$ was generally less than the previous example. This difference can be attributed

TABLE 4.2

Convergence results for an example with osmotic and hydraulic pressure-driven water fluxes. Notation is analogous to that used in Table 4.1. *error*$_{A^-}$ *and error*$_{M^-}$ *were computed using exact solution* $A^-(\infty) = 0.7214268871$ *and* $M^-(\infty) = 1.178097245$. *Results show second-order spatial convergence.*

N	$A^-(\infty)$	error$_{A^-}$	$M^-(\infty)$	error$_{M^-}$
80	0.5625704901	1.589E-1	0.8269022387	3.512E-2
160	0.7165083517	4.919E-3	1.167910666	1.019E-2
320	0.7202750053	1.152E-3	1.175660264	2.437E-3
640	0.7211253473	3.015E-4	1.177467860	6.294E-4

to the osmotic water fluxes induced by the concentration gradient, which, taken in isolation, tend to reduce that gradient.

Volume accuracy and solute conservation results for $N = 80$, 160, 320, and 640 are shown in Table 4.2. These results exhibit second-order convergence at sufficiently large N's. Analogous volume accuracy and mass conservation results were also obtained for the exterior region Ω^+. At $N = 80$, there was substantial water and solute loss in Ω^-, and similar gain in Ω^+ (results not shown), owing to the underresolved concentration gradient near Γ as previously noted. Volume accuracy is less robust in this example than in the results shown for the previous example in Table 4.1. This discrepancy can again be attributed to the larger errors in calculating the boundary concentrations, which are required in the computation of water fluxes (but not in the previous example), thus adversely affecting volume accuracy. Nonetheless, with a grid that is sufficiently refined to resolve the near-boundary transient concentration gradient, volume and mass are approximately conserved. For example, for $N \geq 160$, the relative errors in $A^-(\infty)$ and $M^-(\infty)$ are $\leq 1\%$.

5. Discussion. We have presented a model and a computational technique for simulating fluid motion and solute transport in a highly viscous fluid, described by the Stokes equations, with an immersed elastic membrane. This work extends previous immersed boundary models in two significant ways: Previous models have generally assumed water-impermeable membranes. In contrast, our model represents transmural osmotic and hydraulic pressure-driven water fluxes by modifying the boundary velocity. Also, our model simulates the advection and diffusion of a solute, the concentration of which is discontinuous across the membrane. Numerical results suggest that our method adequately preserves mass conservation of the solute.

The original immersed boundary method and many of its variants, e.g., the blob projection method [8], transfer the singular boundary forces onto the underlying fluid using approximate (smooth) Dirac delta functions typically with $\mathcal{O}(h)$ support. This approach does not capture the jump discontinuity in solution (e.g., p) but rather approximates the solution as a continuous function with a large gradient. In a model where hydraulic pressure-driven water fluxes are represented, the transmural jump in pressure is required. For this reason, an explicit jump method is used in this work. Nonetheless, our model equations can be solved by means of the immersed boundary method, the blob projection method, the method of regularized Stokelets by Cortez [7], or the ghost fluid method [15]. The transmural pressure difference may be approximated using pressure values some grid spacings away from the membrane.

Our method computes a solution to the Stokes equations by solving three separate Poisson problems for p, u, and v, with the singular terms neglected and with the jumps incorporated into the finite-difference scheme. This approach is similar to how

LeVeque and Li [19] used the immersed interface method to solve the Stokes equations. However, the way we calculated the correction terms (see the appendix) follows the method by Mayo [24]. The difference between the immersed interface method and Mayo's method lies in the directions along which correction terms are computed: the immersed interface method follows the directions normal and tangential to the interface, whereas in Mayo's method (and in the present work, as well as in [31]) correction terms are computed along the coordinate lines. A previous study has suggested that, at least for the test cases considered there, the solution errors are similar in the two methods [2]. Indeed, our model equation can just as easily be solved by means of the immersed interface method.

The current model assumes biperiodic boundary conditions, which allow us to decouple the Stokes equations into three Poisson problems for p, u, and v. When other boundary conditions are specified (e.g., no-slip conditions at solid walls), boundary conditions for p may be obtained via the introduction of additional variables and equations (see discussion in [19]), or an integral equation can be solved as in [2] for free boundary. Also in [2] Beale and Layton showed how one may solve the Stokes equations with two distinct fluid viscosities on each side of the membrane in free space by combining the integral equation approach with explicit jump methods. The two-fluid case can also be handled by the immersed interface method using an augmented approach [21].

Transmural water fluxes drive solute concentration up or down on the two sides of the membrane. Consequently, large concentration gradients may develop near the membrane at transient states, especially for sufficiently small solute diffusivity ν's, even though the steady-state concentration converges to a stepwise constant function. Thus, a sufficiently refined spatial grid is needed to resolve that concentration gradient, leading to large computation times when a uniform spatial grid is used. Substantial savings in computation time will likely be gained via the use of adaptive mesh refinement. Indeed, adaptive mesh refinement was successfully applied to the immersed boundary method by Roman, Peskin, and Berger [28], and recently to a high-order immersed interface method by Linnick and Fasel [23].

In the current implementation, the membrane position is updated using an explicit time-stepping method. Also, the jumps in c and its normal derivation are first estimated in a provisional step using an explicit integration of the advection-diffusion equation for solute. Thus, with a sufficiently stiff membrane, or with a sufficiently large solute diffusivity, the restriction on time-step size may become stringent. Development of implicit versions of the method, perhaps similar to the implicit or pseudo-implicit versions of the immersed interface methods [17, 19] and of the immersed boundary method [30], is a worthwhile effort.

The present method is second order in space, but only first order in time. We opted for a first-order time-stepping scheme because of the simplicity of a first-order operator splitting, and because of the likelihood (not verified) that order reduction, which is attributable to the stiffness of the membrane and to diffusion, would restrict solution accuracy to first order regardless. Indeed, solution errors appear to be dominated by spatial errors because of the large concentration gradient (see above) and because of the small time step used. Nonetheless, a version of the method that is second or higher order in time, combined with adaptive mesh refinement to resolve the concentration gradient, should be useful. Strang splitting can be used to develop a second-order time-stepping method with operator-splitting. For third- or higher-order methods with operator-splitting, the multi-implicit Picard deferred correction

approach (formerly known as multi-implicit spectral deferred correction [5]) may be used to generate a high-order solution using a low-order time integration method and a low-order splitting, using an iterative procedure.

We regard this model as a step toward a more sophisticated model that simulates the interactions of fluid, solutes, and an elastic, permeable membrane. A solute-permeable membrane can also be represented through the inclusion of a solute flux term in the solute conservation equation (2.11). Additional solutes, which may have different diffusivities and membrane permeabilities, and which may react with each other, can be represented. Also, although in this work we considered only two-dimensional Stokes flows in the zero Reynolds number limit, our methodology can be extended to three dimensions and to the more comprehensive Navier–Stokes equations. These extensions are the focus of an ongoing project.

The methodology developed in this work is useful in many applications, especially in biofluids. One application can be found in the kidney, which exhibits complex dynamics. In particular, the peristaltic contractions of the renal pelvis "milk" the papilla and interrupts flows along the renal tubules by forcing them to collapse [11]. Thus, peristaltic waves likely have a significant impact on the water and solute transport properties of the renal tubules [16]. Our model can be used to simulate flows along renal tubules undergoing peristalsis and to assess the significance of peristalsis in water and solute transport in the kidney.

Appendix. Derivation of correction terms. In this appendix, we derive correction terms for a finite-difference approximation of u_{xx} that gives second-order accuracy when u and its derivatives have jump discontinuities at $x^* = x_i + \theta h$. We consider here the one-dimensional case where $u = u(x)$; generalization to higher dimensions is straightforward.

Suppose we wish to approximate $u_{xx}(x_i)$ using u_{i-1}, u_i, and u_{i+1}. We first consider the case where $0 < \theta \le 1$. We start by deriving the correction terms for a finite-difference approximation for $u_x(x_i + h/2)$ using u_i and u_{i+1}. Let $u_*^R \equiv u(x^{*+})$ and $u_*^L \equiv u(x^{*-})$. With this notation, Taylor's expansion yields

$$u_{i+1} = u_*^R + u_*'^R(1-\theta)h + \frac{1}{2}u_*''^R(1-\theta)^2h^2 + \mathcal{O}(h^3), \tag{A.1}$$

$$u_i = u_*^L - u_*'^L(1-\theta)h + \frac{1}{2}u_*''^L(1-\theta)^2h^2 + \mathcal{O}(h^3). \tag{A.2}$$

Taking the difference, one obtains

$$u_{i+1} - u_i = [u] + (1-\theta)h[u'] + hu_*'^L + \frac{1}{2}(1-\theta)^2h^2[u''] + \frac{1}{2}(1-2\theta)h^2u_*''^L + \mathcal{O}(h^3). \tag{A.3}$$

Using $u_i' = u_*'^L - \theta h u_*''^L + \mathcal{O}(h^2)$ or $hu_*'^L - \theta h^2 u_*''^L = hu_i' + \mathcal{O}(h^3)$, one obtains

$$u_{i+1} - u_i = [u] + (1-\theta)h[u'] + \frac{1}{2}(1-\theta)^2h^2[u''] + hu_i' + \frac{1}{2}h^2u_*''^L + \mathcal{O}(h^3). \tag{A.4}$$

Since $u_*''^L = u_i'' + \mathcal{O}(h)$, (A.4) can be expressed in terms of u' and the jumps:

$$u_{i+1} - u_i = hu_i' + \frac{1}{2}h^2u_i'' + [u] + (1-\theta)h[u'] + \frac{1}{2}(1-\theta)^2h^2[u''] + \mathcal{O}(h^3). \tag{A.5}$$

If there is no jump in $[x_{i-1}, x_i]$, then

$$u_i - u_{i-1} = hu_i' - h^2 u_i'' + \mathcal{O}(h^3). \tag{A.6}$$

Taking the difference equations (A.5) and (A.6), one obtains the correction terms C_i, where

$$u_i'' = \frac{1}{h^2}\left(u_{i+1} - 2u_i + u_{i-1} + C_i\right), \tag{A.7}$$

$$C_i = -[u] - (1-\theta)h[u'] - \frac{1}{2}(1-\theta)^2 h^2 [u'']. \tag{A.8}$$

Similarly if $x^* = -\theta h$, $0 < \theta \leq 1$, then we have

$$u_{i-1} - u_i = -hu_i' + \frac{1}{2}h^2 u_i'' - [u] + (1-\theta)h[u'] - \frac{1}{2}(1-\theta)^2 h^2 [u''] + \mathcal{O}(h^3). \tag{A.9}$$

Combining (A.9) with $u_{i+1} - u_i = hu_i' + h^2 u_i'' + \mathcal{O}(h^3)$, one obtains the correction terms

$$C_i = [u] - (1-\theta)h[u'] + \frac{1}{2}(1-\theta)^2 h^2 [u'']. \tag{A.10}$$

As discussed in section 3.1, the jump conditions are given for the model variables and for their normal derivatives, not their x- and y-derivatives directly. Thus, we express $[u']$ and $[u'']$ in terms of $[u]$, $[u_n]$, and their arclength derivatives. Suppose $\nabla^2 u = f$, $[u] = \alpha$, and $[u_n] = \beta$. Further suppose that the closed boundary is parameterized by the arclength s, such that $\vec{n} = (\dot{y}, -\dot{x})$, $(\ddot{x}, \ddot{y}) = \kappa(\dot{y}, -\dot{x})$, where κ denotes the curvature and the dots indicate derivatives with respect to s. Then one can obtain the relations

$$[u_x] = \dot{x}\dot{\alpha} + \dot{y}\beta, \tag{A.11}$$

$$[u_y] = \dot{y}\dot{\alpha} - \dot{x}\beta, \tag{A.12}$$

$$[u_{xx}] = 2\kappa\dot{x}\dot{y}\dot{\alpha} + \kappa(\dot{y}^2 - \dot{x}^2)\beta + (\dot{x}^2 - \dot{y}^2)\ddot{\alpha} + 2\dot{x}\dot{y}\dot{\beta} + \dot{y}^2[f], \tag{A.13}$$

$$[u_{yy}] = -2\kappa\dot{x}\dot{y}\dot{\alpha} + \kappa(\dot{x}^2 - \dot{y}^2)\beta + (\dot{y}^2 - \dot{x}^2)\ddot{\alpha} - 2\dot{x}\dot{y}\dot{\beta} + \dot{x}^2[f]. \tag{A.14}$$

REFERENCES

[1] K. M. Arthurs, L. C. Moore, C. S. Peskin, E. B. Pitman, and H. E. Layton, *Modeling arteriolar flow and mass transport using the immersed boundary method*, J. Comput. Phys., 147 (1998), pp. 402–440.

[2] J. T. Beale and A. T. Layton, *On the accuracy of finite difference methods for elliptic problems with interfaces*, Comm. Appl. Math. Comput. Sci., 1 (2006), pp. 91–119.

[3] R. P. Beyer, *A computational model of the cochlea using the immersed boundary methods*, J. Comput. Phys., 98 (1992), pp. 145–162.

[4] J. Blake, *Fluid mechanics of ciliary propulsion*, in Computational Modeling in Biological Fluid Dynamics, L. J. Fauci and S. Gueron, eds., Springer-Verlag, New York, 1999, pp. 1–51.

[5] A. Bourlioux, A. T. Layton, and M. L. Minion, *Higher-order multi-implicit spectral deferred correction methods for problems of reacting flow*, J. Comput. Phys., 189 (2003), pp. 351–376.

[6] N. G. Cogan, R. Cortez, and L. J. Fauci, *Modeling physiological resistance in bacterial biofilms*, Bull. Math. Biol., 67 (2005), pp. 831–853.

[7] R. Cortez, *The method of regularized Stokeslets*, SIAM J. Sci. Comput., 23 (2001), pp. 1204–1225.

[8] R. Cortez and M. Minion, *The blob projection method for immersed boundary problems*, J. Comput. Phys., 161 (2000), pp. 428–453.
[9] R. Dillon, L. Fauci, and D. Gaver III, *A microscale model of bacterial swimming, chemotaxis, and substrate transport*, J. Theor. Biol., 177 (1995), pp. 325–340.
[10] R. H. Dillon and L. J. Fauci, *A fluid-structure interaction model of ciliary beating*, in Computational Modeling in Biological Fluid Dynamics, L. J. Fauci and S. Gueron, eds., Springer-Verlag, New York, 1999, pp. 71–80.
[11] T. M. Dwyer and B. Schmidt-Nielsen, *The renal pelvis: Machinery that concentrates urine in the papilla*, News Physiol Sci., 18 (2003), pp. 1–6.
[12] L. Fauci and A. McDonald, *Sperm mobility in the presence of boundaries*, Bull. Math. Biol., 57 (1995), pp. 679–699.
[13] A. Finkelstein, *Water movement through lipid bilayers, pores, and plasma membranes*, Wiley-Interscience, New York, 1987.
[14] K. Ito, Y. Kyei, and Z. Li, *Higher-order, Cartesian grid based finite difference schemes for elliptic equations on irregular domains*, SIAM J. Sci. Comput., 27 (2005), pp. 346–367.
[15] M. Kang, R. Fedkiw, and X. Liu, *A boundary condition capturing method for multiphase incompressible flow*, J. Sci. Comput., 15 (2000), pp. 323–360.
[16] H. E. Layton, *Concentrating urine in the inner medulla of the kidney*, Comments Theor. Biol., 1 (1989), pp. 179–196.
[17] L. Lee and R. J. LeVeque, *An immersed interface method for the incompressible Navier–Stokes equations*, SIAM J. Sci. Comput., 25 (2003), pp. 832–856.
[18] R. J. LeVeque and Z. Li, *The immersed interface method for elliptic equations with discontinuous coefficients and singular sources*, SIAM J. Numer. Anal., 31 (1994), pp. 1019–1044.
[19] R. J. LeVeque and Z. Li, *Immersed interface methods for Stokes flow with elastic boundaries or surface tension*, SIAM J. Sci. Comput., 18 (1997), pp. 709–735.
[20] Z. Li, *An overview of the immersed interface method and its applications*, Taiwanese J. Math., 7 (2003), pp. 1–49.
[21] Z. Li, K. Ito, and M.-C. Lai, *An Augmented Approach for Stokes Equations with a Discontinuous Viscosity and Singular Forces*, NCSU-CRSC Tech. Report CRSC-TR04-23, North Carolina State University, 2004.
[22] Z. Li and S. R. Lubkin, *Numerical analysis of interfacial two-dimensional Stokes flow with discontinuous viscosity and variable surface tension*, Internat. J. Numer. Methods Fluids, 37 (2001), pp. 525–540.
[23] M. N. Linnick and H. F. Fasel, *A high-order immersed interface method for simulating unsteady incompressible flows on irregular domains*, J. Comput. Phys., 204 (2005), pp. 157–192.
[24] A. Mayo, *The fast solution of Poisson's and the biharmonic equations on irregular regions*, SIAM J. Numer. Anal., 21 (1984), pp. 285–299.
[25] A. Mayo and A. Greenbaum, *Fast parallel iterative solution of Poisson's and the biharmonic equations on irregular regions*, SIAM J. Sci. Comput., 13 (1992), pp. 101–118.
[26] C. S. Peskin, *Numerical analysis of blood flow in the heart*, J. Comput. Phys., 25 (1977), pp. 220–252.
[27] C. S. Peskin, *The immersed boundary method*, Acta Numer., 11 (2002), pp. 479–517.
[28] A. M. Roma, C. S. Peskin, and M. J. Berger, *An adaptive version of the immersed boundary method*, J. Comput. Phys., 153 (1999), pp. 509–534.
[29] S. G. Schultz, *Basic Principles of Membrane Transport*, Cambridge University Press, New York, 1980.
[30] C. Tu and C. S. Peskin, *Stability and instability in the computation of flows with moving immersed boundaries: A comparison of three methods*, SIAM J. Sci. Statist. Comput., 13 (1992), pp. 1361–1376.
[31] A. Wiegmann and K. P. Bube, *The explicit-jump immersed interface method: Finite difference methods for PDEs with piecewise smooth solutions*, SIAM J. Numer. Anal., 37 (2000), pp. 827–862.

SIAM J. SCI. COMPUT.
Vol. 28, No. 6, pp. 2208–2228
© 2006 Society for Industrial and Applied Mathematics

ANALYSIS AND COMPARISON OF GEOMETRIC AND ALGEBRAIC MULTIGRID FOR CONVECTION-DIFFUSION EQUATIONS*

CHIN-TIEN WU† AND HOWARD C. ELMAN‡

Abstract. The discrete convection-diffusion equations obtained from streamline diffusion finite element discretization are solved on both uniform meshes and adaptive meshes. Estimates of error reduction rates for both geometric multigrid (GMG) and algebraic multigrid (AMG) are established on uniform rectangular meshes for a model problem. Our analysis shows that GMG with line Gauss–Seidel smoothing and bilinear interpolation converges if $h \gg \epsilon^{2/3}$, and AMG with the same smoother converges more rapidly than GMG if the interpolation constant β in the approximation assumption of AMG satisfies $\beta \ll (\frac{h}{\sqrt{\epsilon}})^{\alpha}$, where $\alpha = \begin{cases} 1, & h<\sqrt{\varepsilon}, \\ 2, & h\geq\sqrt{\varepsilon}. \end{cases}$ On unstructured triangular meshes, the performance of GMG and AMG, both as solvers and as preconditioners for GMRES, are evaluated. Numerical results show that GMRES with AMG preconditioning is a robust and reliable solver on both type of meshes.

Key words. convection-diffusion equations, multigrid, adaptive mesh refinement

AMS subject classifications. Primary, 65N30, 65N50, 65N55; Secondary, 65F10

DOI. 10.1137/060662940

1. Introduction.

The purpose of this paper is to evaluate different solution strategies for the linear systems obtained from discretization of the convection-diffusion equation

$$\begin{cases} -\epsilon \triangle u + b \cdot \nabla u = f & \text{on } \Omega, \\ u = g & \text{on } \partial\Omega, \end{cases} \tag{1.1}$$

where b and f are sufficiently smooth and the domain Ω is convex with Lipschitz boundary $\partial\Omega$. We are interested in the convection-dominated case, i.e., $|b| \gg \epsilon$. In this setting, the solution typically has steep gradients in some parts of the domain. These may take the form of boundary layers caused by Dirichlet conditions on the outflow boundaries or internal layers caused by discontinuities in the inflow boundaries.

Let $\Im_h$ be a given quasi-uniform mesh of triangles on Ω and let V_h be the linear finite element space on $\Im_h$. It is well known that the standard Galerkin finite element discretization on uniform grids produces inaccurate oscillatory solutions to convection-diffusion problems. Here, we will discretize (1.1) on V_h using the streamline diffusion finite element method (SDFEM) [7], a variant of the standard Galerkin method, where extra diffusion in the streamline direction is introduced. The SDFEM formulation entails finding $u_h \in V_h$ such that

$$B_{sd}(u_h, v_h) = f_{sd}(v_h) \quad \text{for all } v_h \in V_h, \tag{1.2}$$

*Received by the editors June 7, 2004; accepted for publication (in revised form) June 13, 2006; published electronically December 11, 2006.
http://www.siam.org/journals/sisc/28-6/66294.html
†Department of Computer Science and Engineering, National Taiwan Ocean University, Keelung Taiwan 20224, Republic of China (ctwu@cs.ntou.edu.tw).
‡Department of Computer Science and Institute for Advanced Computer Studies, University of Maryland, College Park, MD 20742 (elman@cs.umd.edu). This author's work was supported in part by the National Science Foundation under grant DMS 0208015.

where

$$B_{sd}(u_h, v_h) = \epsilon(\nabla u_h, \nabla v_h) + (b \cdot \nabla u_h, v_h) + \sum_{T \in \Im_h} \delta_T (b \cdot \nabla u_h, b \cdot \nabla v_h)_T,$$

$$f_{sd}(v_h) = (f_h, v_h) + \sum_{T \in \Im_h} (f, \delta_T b \cdot \nabla v_h),$$

and δ_T is a stabilization parameter. With a carefully chosen value of δ_T, the streamline diffusion finite element discretization is able to eliminate most oscillations and produce accurate solutions in the regions where no layers are present [7]. As shown in [6], a good choice of δ_T is

$$\delta_T = \begin{cases} \frac{1}{2\|b\|_T} \left(1 - \frac{1}{P_{e_T}}\right) & \text{if } P_{e_T} > 1, \\ 0 & \text{if } P_{e_T} < 1, \end{cases} \tag{1.3}$$

where

$$P_{e_T} = \frac{\|b\|_T \, h_T}{2\epsilon} \quad \text{for } T \in \Im_h \text{ with diameter } h_T$$

is the mesh Peclét number.

This strategy does not produce accurate solutions in regions containing layers that are not resolved by the grid. Accuracy can be achieved at reasonable cost in such regions by adaptive mesh refinement, in which an a posteriori error estimation strategy is used to identify regions where errors are large, and a marking strategy is used to select elements to be refined. An adaptive solution strategy is to start with a coarse mesh, compute the discrete system, solve the linear system, and then locally refine the mesh using the information provided by an a posteriori error estimation. This adaptive solution process can be applied repeatedly until the a posteriori error estimator is less than a prescribed tolerance or the maximal number of mesh refinement steps is reached.

In this paper, we use the local error estimator proposed by Kay and Silvester [8], with the maximum marking strategy [11], and regular mesh refinement strategy for the adaptive refinement process. We are concerned with the costs of performing this computation with emphasis on the costs of solving the discrete systems that arise at each step of the mesh refinement. Let $A_h u_h = f_h$ denote the discrete system of equations derived from (1.2). Since adaptive mesh refinement produces a sequence of nested meshes, multigrid methods are natural candidates for solving this linear system. Here, we would like to explore the effectiveness of the geometric multigrid (GMG) [5] and algebraic multigrid (AMG) [17] methods. Since the linear system is nonsymmetric, Krylov subspace linear solvers such as the generalized minimal residual method (GMRES) [18] are also good choices. In order to accelerate the convergence of the Krylov subspace linear solvers, good preconditioners are needed. We would also like to investigate the performance of GMRES preconditioned by GMG and AMG.

This paper is organized as follows: Details of the error estimator and refinement strategy are given in section 2 where numerical studies that demonstrate the efficiency of the error indicator are presented. In section 3, multigrid algorithms and some estimates of two-level multigrid convergence are analyzed and numerical results that support our theoretical analysis are given. A comparison of the solvers is shown in section 4. This comparison examines both standalone versions of multilevel multigrid as well as versions in which multigrid is used as a preconditioner for GMRES. We also

compare these with unpreconditioned GMRES, and with a preconditioned GMRES algorithm in which the smoother used for multigrid is used as a preconditioner. These experiments are performed using the following two benchmark problems.

Problem 1: *Constant flow with characteristic and downstream layers.* Equation (1.1) is given with the coefficient $b = (\sin(\phi), \cos(\phi))$, where ϕ measures the angle of the flow direction from the y-axis, and the right-hand side $f = 0$ on the domain $\Omega = [-1, 1] \times [-1, 1]$. The Dirichlet boundary condition is set as $g = 1$ on the segments $y = -1 \cap x > 0$ and $x = 1$, and $g = 0$ elsewhere.

Problem 2: *Flow with closed characteristics:* Here, the coefficient vector (b_1, b_2) is $(2y(1-x^2), 2x(1-y^2))$ and the right-hand side $f = 0$ on the domain $\Omega = [-1, 1] \times [-1, 1]$. The Dirichlet boundary condition is $g = 1$ on the segments $y = 1$ and $g = 0$ elsewhere.

These examples are chosen because they exhibit many of the important features of flow problems so that their effect on discretization and multigrid solvers can be identified. In particular, Problem 1 has an exponential boundary layer at the outflow boundary and a characteristic internal layer induced by a discontinuity on the inflow boundary. Both of these can be treated using adaptive mesh refinement. Moreover, by varying the angle ϕ, we can assess the effect of flow direction on performance of solvers. Problem 2 also has a (characteristic) boundary layer caused by discontinuous boundary conditions, and its recirculating flow requires that care be taken in choosing smoothers for multigrid computations.

Finally, in section 5, we draw our conclusions.

2. Adaptive mesh refinement by a posteriori error estimation. One common technique to increase the accuracy of the finite element solution is mesh refinement. Unlike uniform mesh refinement, the adaptive mesh refinement process refines meshes only in the regions where errors between the weak solution of the partial differential equation and the corresponding finite element solution are large. In general, the adaptive mesh refinement process consists of loops of the following form:

$$\underbrace{\textit{Solve}}_{1} \rightarrow \underbrace{\textit{Compute error indicator}}_{2} \rightarrow \underbrace{\textit{Refine mesh}}_{3}$$

In step 3, elements in which the value of the a posteriori error estimator is large are marked for refinement according to some element selection algorithm. On a given triangulation $\Im_h$, for any element $T \in \Im_h$, let η_T be the a posteriori error indicator of element T. A heuristic marking strategy is the *maximum marking strategy* where an element T^* is marked for refinement if

$$\eta_{T^*} > \theta \max_{T \in \Im_h} \eta_T, \tag{2.1}$$

with a prescribed threshold $0 < \theta < 1$. For other marking strategies, we refer to [11]. Once decisions on where to refine are made, commonly used mesh refinement schemes are regular refinement, where a triangle is divided into four triangles equally, and longest side bisection [15], [16], where a triangle is divided into two triangles by adding a new node to the midpoint of the longest edge. The maximum marking strategy and regular refinement scheme are used in our experiments.

A reliable computable a posteriori error estimator in step 2 is the key for the adaptive mesh refinement process to succeed. For the convection-diffusion equation discretized by SDFEM, the first a posteriori error estimation where the error is estimated by computing the residual was proposed by Verfürth [21], and the first a posteriori error estimation where the error is estimated by solving a local problem was

developed by Kay and Silvester [8]. In this study, we use the Kay and Silvester a posteriori error estimator, which we have found to be a more effective choice in [23]. Hereafter, we call this indicator the KS-indicator. First, let us introduce the following abbreviations. Let $\|\cdot\|_{0,\Omega}$ and $\|\cdot\|_{0,T}$ denote the L^2 norm on domain Ω and element T, respectively. Let $\mathcal{E}(T)$ be the set of edges of T and $\omega_T = \cup_{T' \cap T \in \mathcal{E}(T)} T'$, i.e., ω_T is the set of triangles T' that share a common edge with T. Let P_T^0 be the L^2 projection onto the space of polynomials of degree 0 on element T. The interior residual R_T of element T and the interelement flux jump R_E of edge E are defined as follows:

$$
\begin{aligned}
R_T &= (f - b \cdot \nabla u_h)|_T, \\
R_T^0 &= \mathcal{P}_T^0(R_T), \\
R_E &= \begin{cases} [\frac{\partial u_h}{\partial n_E}]_E & \text{if } E \in \Omega, \\ 0 & \text{if } E \in \partial\Omega, \end{cases}
\end{aligned}
$$

where $[\cdot]_E$ is the jump across edge E.[1] Let Φ be the element affine mapping from the physical domain to the computational domain and let χ_i be the nodal basis function of node i. The approximation space is denoted as $Q_T = Q_T \bigoplus B_T$, where

$$
\begin{aligned}
Q_T = span\{\psi_E \circ \Phi^{-1} |\ \psi_E = 4\chi_i\chi_j,\ i,j \text{ are the endpoints of } E \text{ and} \\
E \in \partial T \bigcap (\Omega \bigcup \Gamma_N)\}
\end{aligned}
$$

is the space spanned by quadratic edge bubble functions and

$$
B_T = span\left\{\psi_T \circ \Phi^{-1} |\ \psi_T = 27 \prod_{i=1}^{3} \chi_i\right\}
$$

is the space spanned by cubic interior bubble functions. For details, see [8]. On each element T, the error estimator is then given by $\eta_{h,T} = \|\nabla e_T\|_{0,T}$, where $e_T \in Q_T$ satisfies

$$
\epsilon(\nabla e_T, \nabla v)_T = (R_T^0, v)_T - \frac{1}{2}\epsilon \sum_{E \in \partial T} (R_E, v)_E. \tag{2.2}
$$

Let $e_h = u - u_h$. The a posteriori error estimation is specified as follows:

(global upper bound)

$$
\|\nabla(e_h)\|_{0,\Omega} \le C \left(\sum_{T \in \Im_h} \eta_{h,T}^2 + \sum_{T \in \Im_h} \left(\frac{h}{\epsilon}\right)^2 \left\|R_T - R_T^0\right\|_{0,T}^2 \right)^{1/2} \tag{2.3}
$$

(local lower bound)

$$
\eta_{h,T} \le c \left(\|e_h\|_{0,\omega_T} + \sum_{T \subset \omega_T} \frac{h_T}{\epsilon} \|b \cdot \nabla e_h\|_{0,T} + \sum_{T \subset \omega_T} \frac{h_T}{\epsilon} \left\|R_T - R_T^0\right\|_{0,T} \right), \tag{2.4}
$$

where constants C and c are independent of the diffusion parameter ϵ and mesh size h.

[1] This definition of R_E for $E \in \partial\Omega$ is for Dirichlet boundary conditions. We don't consider Neumann conditions in this study; in case $E \in \partial\Omega$ and a Neumann condition holds, the flux jump R_E would be set to $-2(\frac{\partial u_h}{\partial n_E})$ [8].

The sharpness of the error indicator can be revealed by computing the local effectivity index,

$$E_T = \max_{t \in \Im_h} \frac{\eta_{h,T}}{|u - u_h|_{1,T}},$$

and the global effectivity index,

$$E_\Omega = \frac{(\sum_{T \in \Im_h} \eta_{h,T}^2)^{1/2}}{|u - u_h|_{1,\Omega}},$$

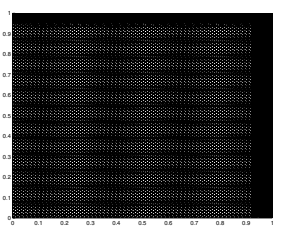
where $|\cdot|_{1,\Omega}$ and $|\cdot|_{1,T}$ represent the H^1 seminorms on domain Ω and element T, respectively. If both indices are close to 1, the error indicator is reliable and efficient. The global upper bound provides an estimate of the error over the whole domain, and the local lower bound locates where the error is large. It has been shown that E_Ω grows in proportion to $\sqrt{Pe}$ and E_T grows in proportion to Pe for the KS-indicator [8]. The following numerical results support this conclusion.

Consider here equation (1.1) on the domain $\Omega = [0,1] \times [0,1]$ with $b = (\beta_1, \beta_2) = (\sin 75^\circ, \cos 75^\circ)$ and the Dirichlet boundary condition is obtained from the analytic solution

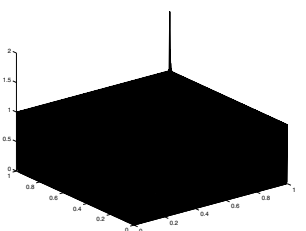
$$u(x,y) = \frac{e^{\beta_1 x/\varepsilon} - 1}{e^{\beta_1/\varepsilon} - 1} + \frac{e^{\beta_2 y/\varepsilon} - 1}{e^{\beta_2/\varepsilon} - 1}. \tag{2.5}$$

Both global effectivity index E_Ω and local effectivity index E_T are computed. In order to see how the effectivity indices change in terms of the diffusion parameter ϵ and mesh size h, the problem is solved over uniform meshes with mesh size $h = \frac{1}{8}, \frac{1}{16}, \frac{1}{32}$, and $\frac{1}{64}$ for $\epsilon = \frac{1}{64}, \frac{1}{256}, \frac{1}{1024}$, and $\frac{1}{4096}$. For accurately approximating the true error $u - u_h$, a very fine mesh $\Im_f$, as shown in Figure 2.1, consisting of 11272 nodes and 21379 elements is generated by first solving the above test problem with $\epsilon = 10^{-3}$ on a 64×64 mesh followed by three mesh refinement steps where a large threshold $\theta = 0.75$ in the maximum marking strategy (2.1) is used to prevent unnecessary over refinement. An approximation of the exact error is then computed on $\Im_f$ by

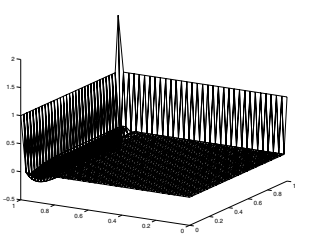
$$|u - u_h|_{1,\Omega^*} = \left(\sum_{\tau \in \Im_f \cup \Omega^*} |u - u_h|_{1,\tau}^2 \right)^{1/2} \quad \text{for } \Omega^* = \Omega \text{ or } T \in \Im_h,$$



(a) Mesh $\Im_f$.



(b) Solution u on $\Im_f$.



(c) SDFEM solution u_h on $\Im_h$, where $h = \frac{1}{32}$.

FIG. 2.1.

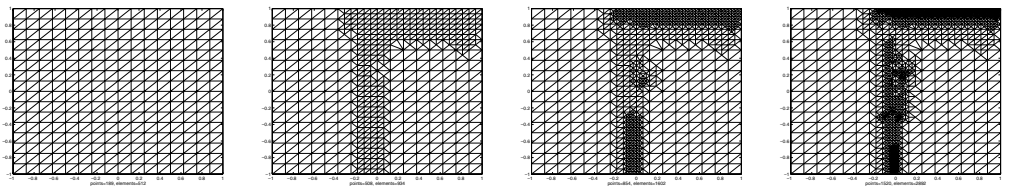
TABLE 2.1
Effectivity indices of the KS-indicator.

ϵ	8×8	16×16	32×32	64×64
$\frac{1}{64}$	1.156	0.951	0.979	1.022
$\frac{1}{256}$	2.044	1.457	1.105	0.929
$\frac{1}{1024}$	4.016	2.772	1.952	1.422
$\frac{1}{4096}$	8.470	5.842	4.075	2.867

(a) Global index E_Ω.

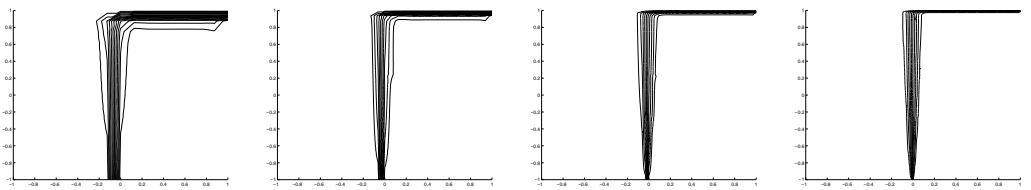
ϵ	8×8	16×16	32×32	64×64
$\frac{1}{64}$	2.504	1.714	1.687	1.557
$\frac{1}{256}$	9.242	4.637	2.536	1.750
$\frac{1}{1024}$	36.95	18.48	9.239	4.629
$\frac{1}{4096}$	147.8	73.90	36.95	18.48

(b) Local index E_T.



(a) $\Im_1$. (b) $\Im_2$. (c) $\Im_3$. (d) $\Im_4$.

FIG. 2.2. *Adaptively refined mesh with threshold value* $\theta = 0.01$.



(a) Solution on $\Im_1$. (b) Solution on $\Im_2$. (c) Solution on $\Im_3$. (d) Solution on $\Im_4$.

FIG. 2.3. *Contour plots of the solutions of Problem* 1 *on adaptively refined meshes starting with a uniform* 16×16 *mesh.*

where the discrete true solution u is evaluated directly from (2.5) on $\Im_f$ and the SDFEM solution u_h is prolonged from $\Im_h$ onto $\Im_f$ by standard bilinear interpolation. In Table 2.1, for $h \gg \epsilon$, E_Ω and E_T decrease by factors approximately of size $\sqrt{2}$ and 2, respectively, as the mesh size is halved. Also, E_Ω and E_T double and quadruple, respectively, when ϵ is reduced by a factor of four. Thus, $E_T = O(P_e)$ and $E_\Omega = O(\sqrt{P_e})$, as shown in [8].

Although there is deterioration in reliability and efficiency for these error indicators, the regions with large errors can still be located when a good threshold value θ is chosen. Next, we give a representative picture of how the adaptive solution process improves the solution quality. We use the KS-indicator for mesh refinement in solving Problem 1 with the diffusion parameter $\epsilon = 10^{-3}$ and $\phi = 0$. From the local lower bound (2.4), we suspect that the value of the indicator $\eta_{h,T}$ in the outflow boundary layer regions is $O(\frac{h}{\epsilon})$ larger than its value in the internal layer regions. Therefore, for the case considered here, a threshold value $\theta = 0.01 \approx \frac{\epsilon}{h}$ is chosen so that both the boundary and internal layer regions can be refined by the maximum marking strategy (2.1). A sequence of adaptively refined meshes is shown in Figure 2.2 in which ele-

ments in both boundary layer and internal layer regions are refined. It can be seen in Figure 2.3 that both the characteristic and downstream layers are resolved accurately by the mesh refinement process.

3. The multigrid algorithms. Given a sequence of meshes $\Im_1, \Im_2, \ldots, \Im_n$, let V_k be the piecewise linear finite element subspace associated with $\Im_k$, let A_k be the matrix defined on $\Im_k$, and let w_k be the initial guess. Let MG_1 represent the direct solver on the coarsest grid $\Im_1$. The general form of the k-level multigrid algorithm is shown in Algorithm 1, where f_k is the right-hand side obtained from discretization of data function f on $\Im_k$, M_k^{-1} represents the smoothing operator, and the operators I_k^{k-1} and I_{k-1}^k represent the grid transfers, restriction and prolongation, between $\Im_{k-1}$ and $\Im_k$. The multigrid V-cycle and W-cycle are defined by choosing $m = 1$ and $m = 2$, respectively.

Unlike in solving self-adjoint elliptic problems, GMG methods usually do not possess mesh-independent convergence for the convection-diffusion equation with dominant convection unless the mesh size $h \ll \epsilon$ or special treatments are employed for the meshes, interpolations, and relaxation [4] when $h \gg \epsilon$. For $h \ll \epsilon$ (and coarse grid size also $< \epsilon$), Bramble et al. [1], [2] and Wang [22] have shown GMG converges with a rate independent of mesh size by using a *compact perturbation technique*. To achieve mesh-independent convergence for $h \gg \epsilon$, Reusken and Olshanskii have recently shown L^2-convergence by using semicoarsening, matrix-dependent prolongation, and line relaxation for Problem 1 discretized by an upwind finite difference method [9], [14], and Szepessy proved L^1-convergence on R^2 by residual damping through large smoothing steps [12]. On the other hand, the convergence of AMG is essentially based on an M-matrix property of the discrete system [17]. No convergence result of algebraic multigrid for the convection-diffusion equation exists to our knowledge. In this section, we describe the versions of GMG and AMG considered here. We restrict our attention to the 2-level V-cycle multigrid and show some convergence results for Problem 1 on uniform rectangular meshes. For this problem, the matrix from SDFEM discretization with stabilization parameter $\delta_T = \frac{h}{2}$ has the form

$$A_k = \begin{bmatrix} D & -U & & & & \\ -L & D & -U & & 0 & \\ & -L & \ddots & \ddots & & \\ & & \ddots & \ddots & -U & \\ & 0 & & -L & D & -U \\ & & & & -L & D \end{bmatrix}, \tag{3.1}$$

Algorithm 1 Multigrid algorithm.

1. set $x_k = w_k$,
2. (presmoothing) $x_k = w_k + M_k^{-1}(g_k - A_k w_k)$,
3. (restriction) $\bar{g}_k = I_k^{k-1}(g_k - A_k x_k)$,
4. (correction) $q_i = MG_{k-1}(q_{i-1}, \bar{g}_k)$ for $1 \le i \le m$, $m = 1$ or 2 and $q_0 = 0$,
5. (prolongation) $\bar{q}_m = I_{k-1}^k q_m$,
6. set $x_k = x_k + \bar{q}_m$,
7. (postsmoothing) $x_k = x_k + M_k^{-1}(g_k - A_k x_k)$,
8. set $y_k = MG_k(w_k, g_k) = x_k$.

where the block tridiagonal matrices

$$
\begin{aligned}
D &= h \times \text{tridiag}\left[\frac{1}{6} - \frac{1}{3}\frac{\epsilon}{h}, \frac{2}{3} + \frac{8}{3}\frac{\epsilon}{h}, \frac{1}{6} - \frac{1}{3}\frac{\epsilon}{h}\right], \\
L &= h \times \text{tridiag}\left[\frac{1}{6} + \frac{1}{3}\frac{\epsilon}{h}, \frac{2}{3} + \frac{1}{3}\frac{\epsilon}{h}, \frac{1}{6} + \frac{1}{3}\frac{\epsilon}{h}\right], \\
U &= \epsilon \times \text{tridiag}\left[\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right].
\end{aligned} \tag{3.2}
$$

In the following, we describe each component of the multigrid methods used here. First, we use the same smoother in both GMG and AMG. On uniform rectangular meshes, one step of a horizontal line Gauss–Seidel (HLGS) smoother is used. The HLGS method uses the block lower triangle of A_k as a splitting operator. (Thus, the HLGS sweep can be viewed as proceeding along horizontal lines, one line at a time, in the vertical direction.) Second, in GMG, (1) the coarse grids are obtained directly from the mesh refinement process; (2) the grid transfer is via linear interpolation for prolongation, and the restriction operator I_k^{k-1} is then taken to be the transpose of the prolongation operator I_{k-1}^k; and (3) the matrix A_k is obtained directly from SDFEM discretization in the refinement process. For AMG, we use the method developed by Ruge and Stüben [17] (AMG): (1) a coarse grid at level $k-1$ is generated by coarsening the fine grid along the so-called strong connection direction of matrix graph $\mathcal{G}$ of $A_k = (a_{i,j})$ where a directive edge $\overrightarrow{e}_{i,j} \in \mathcal{G}$ is defined to be strongly connected from node i to node j if $\mu \le \frac{-a_{i,j}}{\max_{m \neq i}(-a_{i,m})}$ for a given parameter $0 < \mu < 1$; (2) the interpolation operator I_{k-1}^k is defined dynamically by a sophisticated algebraic formula during the AMG coarsening process [17]; and (3) the matrix A_{k-1} for a coarse grid is computed by $I_k^{k-1} A_k I_{k-1}^k$. The above process is repeated until all coarse grids are generated.

Next, we show the convergence behavior of 2-level V-cycle multigrid for Problem 1. Let E_k^s be the error reduction operator of the HLGS smoother and let $E_k^c = I - I_{k-1}^k A_{k-1}^{-1} I_k^{k-1} A_k$ be the coarse grid correction operator. The error reduction operators of multigrid methods can then be written as $E_k^s E^c E_k^s$. The notation E_k^{mg} and E_k^{amg} denotes the error reduction operators of geometric multigrid and the algebraic multigrid, respectively. The notation $\|\cdot\|$ denotes the usual discrete L^2 norm for any vector and matrix. The notation $\|\cdot\|_0$ and $\|\cdot\|_1$ represents the continuous L^2 and H^1 norms. Variants of these norms are specified using matrix notation in (3.17) below.

In order to show convergence, the following auxiliary lemmas are needed.

LEMMA 3.1. *Given two symmetric matrices B_1 and B_2, assume that $B_1, B_2 \ge 0$, B_1 is irreducible, and B_2 is positive definite. The following properties hold.*

1. *There exists a positive eigenvector x^+ of $B_2^{-1}B_1$ such that*

$$B_2^{-1}B_1 x^+ = \rho(B_2^{-1}B_1)x^+. \tag{3.3}$$

2. *If $\alpha I - B_2^{-1}B_1$ is nonsingular and $(\alpha I - B_2^{-1}B_1)^{-1} \ge 0$, then $\rho(B_2^{-1}B_1) < \alpha$.*

Proof. The existence of a positive eigenvector satisfying (3.3) is essentially a generalization of the well-known Perron–Frobenius theorem [20, Theorem 2.7]. Using (3.3), one can prove the second result using a standard argument of the Perron–Frobenius theory; see Theorem 3.16 in [20]. ∎

LEMMA 3.2. *Let L and D be the matrices defined in* (3.2). *For any $\delta \ge (1 + \frac{2\epsilon}{h})\frac{h}{\epsilon}$, the matrix $\delta(D-L) - D$ is an M-matrix.*

Proof. Let us choose $\delta = \frac{h\gamma}{\epsilon}$ for some $\gamma > 0$. From (3.2), $D - L = \frac{\epsilon}{3} \times$ tridiag$[-2, 7, -2]$. Therefore, we have

$$\begin{aligned}\delta(D-L) - D &= h \times \text{tridiag}\left[\frac{-2\gamma}{3}, \frac{7\gamma}{3}, \frac{-2\gamma}{3}\right] - h \times \text{tridiag}\left[\frac{1}{6} - \frac{\epsilon}{3h}, \frac{2}{3} + \frac{8\epsilon}{h}, \frac{1}{6} - \frac{\epsilon}{3h}\right] \\ &= h \times \text{tridiag}\left[-\left(\frac{2\gamma}{3} + \frac{1}{6} - \frac{\epsilon}{3h}\right), \frac{7\gamma}{3} - \frac{2}{3} - \frac{8\epsilon}{3h}, -\left(\frac{2\gamma}{3} + \frac{1}{6} - \frac{\epsilon}{3h}\right)\right].\end{aligned}$$

Since

$$\frac{7\gamma}{3} - \frac{2}{3} - \frac{8\epsilon}{3h} - 2\left(\frac{2\gamma}{3} + \frac{1}{6} - \frac{\epsilon}{3h}\right) = \gamma - 1 - \frac{2\epsilon}{h},$$

clearly, for $\gamma \geq 1 + \frac{2\epsilon}{h}$, the matrix $\delta(D-L) - D$ is irreducible and weakly diagonal dominant. This implies that the matrix $\delta(D-L) - D$ and $(\delta(D-L) - D)^{-1}$ are positive definite. Moreover, since the off-diagonal entries of $(\delta(D-L) - D)$ are all negative, the matrix $\delta(D-L) - D$ is an M-matrix for $\delta \geq (1 + \frac{2\epsilon}{h})\frac{h}{\epsilon}$. ∎

LEMMA 3.3. *For $h \gg \epsilon$, the error reduction matrix E_k^s of the HLGS iterative method for the matrix obtained from SDFEM discretization of Problem* 1 *satisfies the following inequality:*

$$\|E_k^s\| \leq 3\frac{\epsilon}{h^2} \min\left\{\frac{3h}{\sqrt{\epsilon}}, 1\right\}. \tag{3.4}$$

Proof. With A_k from (3.1), E_k^s is the product of the inverse of the block lower triangular part of A_k and the strict block upper triangular part of A_k. By direct computation, one has $E_k^s = G_1 G_2$, where

$$G_1 = \begin{bmatrix} 0 & I & & & \\ 0 & D^{-1}L & \ddots & & \\ \vdots & \vdots & \ddots & I & \\ 0 & (D^{-1}L)^{n-2} & \cdots & D^{-1}L & I \\ 0 & (D^{-1}L)^{n-1} & \cdots & (D^{-1}L)^2 & D^{-1}L \end{bmatrix}, \; G_2 = \begin{bmatrix} 0 & & & \\ & D^{-1}U & & \\ & & \ddots & \\ & & & D^{-1}U \end{bmatrix}.$$

From the Gerschgorin circle theorem, the following inequalities hold for $h \gg \epsilon$:

$$\begin{aligned}\|U\| &= \rho(U) \leq \epsilon, \\ \left\|D^{-1}\right\| &= \rho(D^{-1}) = \frac{1}{\lambda_{min}(D)} < \frac{1}{\frac{h}{3} + \frac{10\epsilon}{3}} \leq \frac{3}{h}.\end{aligned}$$

Therefore, we have

$$\|G_2\| \leq 3\frac{\epsilon}{h}. \tag{3.5}$$

Next, we estimate $\|G_1\|$. First, let us estimate $\left\|D^{-1}L\right\|$. Considering $D^{-1}L = I - D^{-1}(D-L)$, we have

$$\alpha I - D^{-1}L = D^{-1}(D-L) - (1-\alpha)I = (1-\alpha)\left\{D^{-1}\left[\frac{1}{1-\alpha}(D-L) - D\right]\right\}. \tag{3.6}$$

Let us choose α satisfying $\frac{1}{1-\alpha} = \delta = (1 + \frac{2\epsilon}{h})\frac{h}{\epsilon}$. Lemma 3.2 implies that the matrix $\frac{1}{1-\alpha}(D-L)-D$ is an M-matrix. Consequently, $(\frac{1}{1-\alpha}(D-L)-D)^{-1} \geq 0$. Then using (3.6) and the fact that $D \geq 0$, it follows that the matrix $\alpha I - D^{-1}L > 0$. Since D is also positive definite, by Lemma 3.1 we can conclude that

$$\rho(D^{-1}L) < \alpha = 1 - \frac{1}{\delta} = 1 - \frac{\epsilon}{h}\left(\frac{1}{1+2\frac{\epsilon}{h}}\right) < 1 - \frac{\epsilon}{3h}. \tag{3.7}$$

Let $x = (x_1, x_2, \ldots, x_N) \in V_h$, where $x_i \in R^N$ and $\sum_{i=1}^{N} \|x_i\|^2 = 1$. We have that $\|G_1 x\| < \|G_1 y\|$ for $y = (\|x_1\| x^+, \|x_2\| x^+, \ldots, \|x_N\| x^+)$, where x^+ is the Perron–Frobenius eigenvector of $D^{-1}L$. Therefore, the eigenvector corresponding to the maximum eigenvalue has the following form:

$$y = (0, \beta_1 x^+, \beta_2 x^+, \ldots, \beta_{N-1} x^+), \text{ where } \sum_{i=1}^{N-1} \beta_i^2 = 1.$$

By direct computation,

$$\begin{aligned}\|G_1 y\| &\leq \left\{ \sum_{i=1}^{N} \left\| \sum_{k=1}^{i} \beta_k (D^{-1}L)^{i-k} x^+ \right\|^2 \right\}^{1/2} \\ &= \left\{ \sum_{i=1}^{N} \left(\sum_{k=1}^{i} \beta_k l^{i-k} \right)^2 \right\}^{1/2},\end{aligned}$$

where $l = \rho(D^{-1}L)$. Since $l < 1$ and $\sum_{i=1}^{N-1} \beta_i^2 = 1$, the inequalities

$$\|G_1\| \leq \left\{ \sum_{i=1}^{N-1} \left(\sum_{k=1}^{i} \beta_k \right)^2 \right\}^{1/2} \leq \left(\sum_{i=1}^{N-1} i \right)^{1/2} \leq N \tag{3.8}$$

and

$$\begin{aligned}\|G_1\| &\leq \left\{ \sum_{i=1}^{N-1} \left[\sum_{k=1}^{i} \beta_k^2 \right] \left[\sum_{k=1}^{i} (l^{i-k})^2 \right] \right\}^{1/2} \leq \left\{ \sum_{i=1}^{N-1} \sum_{k=1}^{i} (l^{i-k})^2 \right\}^{1/2} \\ &\leq \left\{ \frac{1}{1-l^2} \left(\sum_{i=1}^{N-1} 1 - l^{2i} \right) \right\}^{1/2} \leq \left(\frac{N}{1-l} \right)^{1/2}\end{aligned} \tag{3.9}$$

hold. Recall that $l < 1 - \frac{\epsilon}{3h}$ from (3.7). By combining (3.8) and (3.9), we have

$$\|G_1\| \leq \frac{1}{h} \min\left\{ \frac{3h}{\sqrt{\epsilon}}, 1 \right\}. \tag{3.10}$$

Therefore, from (3.5) and (3.10), inequality (3.4) holds. ∎

LEMMA 3.4 (smoothing property). *Let E_k^s be the error reduction operator of the HLGS iterative method on V_k. The following inequality holds:*

$$\|A_k(E_k^s)\| \leq \epsilon \left(1 + 3\frac{\epsilon}{h_k^2} \min\left\{ \frac{3h_k}{\sqrt{\epsilon}}, 1 \right\} \right). \tag{3.11}$$

Proof. By directly multiplying A_k and E_k^s, we have

$$A_k E_k^s = \begin{bmatrix} 0 & U - U(D^{-1}L)D^{-1}U & -UD^{-1}U & & \\ 0 & -U(D^{-1}L)^2D^{-1}U & \ddots & \ddots & \\ \vdots & \vdots & \ddots & -U(D^{-1}L)D^{-1}U & -UD^{-1}U \\ 0 & -U(D^{-1}L)^{n-1}D^{-1}U & \cdots & -U(D^{-1}L)^2D^{-1}U & -U(D^{-1}L)D^{-1}U \\ 0 & 0 & \cdots & 0 & 0 \end{bmatrix}$$

$$= \operatorname{diag}[U](T_1 - T_2G_2),$$

where

$$T_1 = \begin{bmatrix} 0 & I & & & \\ 0 & 0 & \ddots & & \\ \vdots & \vdots & \ddots & I & \\ 0 & 0 & \cdots & 0 & I \\ 0 & 0 & \cdots & 0 & 0 \end{bmatrix} \text{ and } T_2 = \begin{bmatrix} 0 & D^{-1}L & I & & \\ 0 & (D^{-1}L)^2 & \ddots & \ddots & \\ \vdots & \vdots & \ddots & D^{-1}L & I \\ 0 & (D^{-1}L)^{n-1} & \cdots & (D^{-1}L)^2 & D^{-1}L \\ 0 & 0 & \cdots & 0 & 0 \end{bmatrix}.$$

Using the same argument as in the proof of Lemma 3.3, one can show

$$\|T_2G_2\| \le 3\frac{\epsilon}{h_k^2} \min\left\{\frac{3h_k}{\sqrt{\epsilon}}, 1\right\}.$$

Therefore,

$$\begin{aligned} \|A_k E_k^s\| &\le \epsilon\left(1 + 3\frac{\epsilon}{h_k^2} \min\left\{\frac{3h_k}{\sqrt{\epsilon}}, 1\right\}\right) \\ &= \epsilon\left(1 + 3\frac{\epsilon}{h_k^2} \min\left\{\frac{3h_k}{\sqrt{\epsilon}}, 1\right\}\right). \qquad \square \end{aligned}$$

Using the inequalities in Lemmas 3.3 and 3.4, our multigrid convergence results can be shown. In the following, estimates of the error reduction operators of GMG and AMG are given in Theorems 3.5 and 3.7, respectively.

THEOREM 3.5. *Let E_k^{mg} be the error reduction operator of the 2-level V-cycle geometric multigrid in Algorithm 1 with the HLGS smoother and bilinear interpolation prolongation on a uniform rectangular mesh $\Im_k$ for Problem 1. The following estimate for E_k^{mg} holds:*

$$\|E_k^{mg}\| \le c\frac{\epsilon}{h^{3/2}} \quad \textit{for some constant } c. \tag{3.12}$$

Proof. First, in general, the error reduction operator of GMG can be written as

$$E_k^{mg} = E_k^s E^c E_k^s = E_k^s(I - I_{k-1}^k(I - E_{k-1}^{mg})A_{k-1}^{-1}I_k^{k-1}A_k)E_k^s. \tag{3.13}$$

For 2-level GMG, since $E_{k-1}^{mg} = 0$, we have

$$\begin{aligned} \|E_k^{mg}\| &= \left\|E_k^s(I - I_{k-1}^k A_{k-1}^{-1} I_k^{k-1} A_k)E_k^s\right\| \\ &\le \|E_k^s\| \left\|(I - I_{k-1}^k A_{k-1}^{-1} I_k^{k-1} A_k)E_k^s\right\|. \end{aligned} \tag{3.14}$$

Let $P_k : H^1 \mapsto V_k$ be the projection operator defined by $B_{sd}(P_k u, v) = B_{sd}(u, v)$ for all $v \in V_k$ for all k. We have $I_k^{k-1} A_k = A_{k-1} P_{k-1}$ on V_k. Therefore, let $e^s = E_k^s(e)$ for any $e \in V_k \subset H^1$,

$$\begin{aligned}
\left\|(I - I_{k-1}^k A_{k-1}^{-1} I_k^{k-1} A_k) E_k^s(e)\right\| &= \left\|(I - I_{k-1}^k P_{k-1}) e^s\right\| \\
&\le c h_k^{-2} \left\|(I - P_{k-1}) e^s\right\|_0 \le c h_k^{-2} \frac{1}{\sqrt{\epsilon}} \left\|(I - P_{k-1}) e^s\right\|_1 \\
&\le c h_k^{-2} \sqrt{\frac{h_k}{\epsilon}} |e^s|_1 \quad \text{by the a priori error estimation,} \\
&\le c h_k^{-2} \sqrt{\frac{h_k}{\epsilon^2}} \left\|A_k E_k^s(e)\right\|_0 \quad \text{by the regularity estimate,} \\
&\le c\sqrt{h_k} \left(1 + 3\frac{\epsilon}{h_k^2} \min\left\{\frac{3h_k}{\sqrt{\epsilon}}, 1\right\}\right) \|e\| \quad \text{by (3.11),}
\end{aligned}$$

where c is a constant independent of h and ϵ. By combining the above estimate and (3.4), inequality (3.14) implies

$$\|E_k^{mg}\| \le c \frac{\epsilon}{h^{3/2}} \min\left\{\frac{h}{\sqrt{\epsilon}}, 1\right\} \left(1 + \frac{\epsilon}{h^2} \min\left\{\frac{h}{\sqrt{\epsilon}}, 1\right\}\right) \le c \frac{\epsilon}{h^{3/2}}. \qquad \square$$

Remark 3.6. From inequality (3.12), it is clear that the geometric multigrid converges when $h_k \gg \epsilon^{2/3}$. The above 2-level analysis can be generalized by mathematical induction and estimation of (3.13). In [23], we have found that the multigrid V-cycle converges when $h \gg \sqrt{\epsilon}$.

Next, we analyze AMG convergence. The framework to prove the convergence of algebraic multigrid is based on the *smoothing assumption*,

$$\exists\ \alpha > 0 \text{ such that } \|E_k^s e_h\|_1^2 \le \|e_h\|_1^2 - \alpha \|e_h\|_2^2 \text{ for any } e_h \in V_k, \tag{3.15}$$

and the *approximation assumption*,

$$\min_{e_H \in V_{k-1}} \left\|e_h - I_H^h e_H\right\|_0^2 \le \beta \|e_h\|_1^2 \text{ with } \beta > 0 \text{ independent with } e_h \in V_k, \tag{3.16}$$

where

$$\|v\|_0^2 = \langle Dv, v\rangle, \quad \|v\|_1^2 = \langle A_k v, v\rangle, \quad \text{and} \quad \|v\|_2^2 = \langle D^{-1} A_k v, A_k v\rangle \tag{3.17}$$

for any $v \in V_k$. For problems where the coefficient matrix is an M-matrix, Ruge and Stüben have proved that 2-level AMG converges when both the smoothing assumption and the approximation assumption hold for M-matrices. It has also been shown that the smoothing assumption holds for the usual point Gauss–Seidel relaxation, and the approximation assumption holds for some simple coarse grid selection algorithms

with algebraic interpolation formula [17]. The classical algebraic multigrid coarsening algorithm is designed to satisfy the approximation assumption based on the property of algebraically smooth error, whereby the smooth errors vary slowly along the strong connection directions.

For the discrete convection-diffusion Problem 1, the matrix obtained from SDFEM discretization is nonsymmetric and not an M-matrix. As a result, the usual framework of AMG analysis cannot be applied directly. In Lemma 3.2 of [14], Reusken has shown that there is a constant c independent of ϵ and h such that $\|E^c\| \leq c$ for a two-grid method with a coarse grid from semicoarsening and a matrix-dependent prolongation. In the following, we take this as an assumption and estimate the error reduction rate of AMG in terms of the parameter β in the approximation assumption (3.16) by using (3.4) and (3.11).

THEOREM 3.7. *Assume the approximation assumption* (3.16) *holds. Then the following inequality holds:*

$$\|E_k^{amg}\| \leq \sqrt{3}c\frac{\epsilon^2}{h^3}\min\left\{\frac{h}{\sqrt{\epsilon}},1\right\}\frac{\beta}{\sqrt{h}}, \tag{3.18}$$

where c is some constant independent of ϵ and h.

Proof. For any $e \in V_k$, we have

$$\|E_k^{amg}(e)\| = \|E_k^s E^c E_k^s(e)\| \leq \frac{\epsilon}{h^2}\min\left\{\frac{h}{\sqrt{\epsilon}},1\right\}\|E^c E_k^s(e)\|. \tag{3.19}$$

Let $\bar{e} = E_k^s(e)$. Since $\mathrm{Ker}(E^c) = Rang(I_{k-1}^k)$, we have

$$\begin{aligned}
\|E^c(\bar{e})\| &\leq c\min_{e_H\in V_{k-1}}\left\|\bar{e} - I_{k-1}^k e_H\right\| \leq c\sqrt{\frac{3}{h}}\min_{e_H\in V_{k-1}}\left\|\bar{e} - I_{k-1}^k e_H\right\|_0 \\
&\leq c\sqrt{\frac{3}{h}}\beta\|\bar{e}\|_1 \text{ by approximation assumption (3.16)}, \\
&\leq \sqrt{\frac{3}{h}}\beta(\|A_k E_k^s(e)\|\,\|E_k^s(e)\|)^{1/2} \\
&\leq c\sqrt{\frac{3}{h}}\beta\left(\epsilon\left[1+\frac{\epsilon}{h^2}\min\left\{\frac{h}{\sqrt{\epsilon}},1\right\}\right]\left[\frac{\epsilon}{h^2}\min\left\{\frac{h}{\sqrt{\epsilon}},1\right\}\right]\right)^{1/2}\|e\| \\
&\qquad \text{by (3.4) and (3.11)} \\
&\leq c\sqrt{\frac{3}{h}}\frac{\epsilon}{h}\beta\|e\|.
\end{aligned} \tag{3.20}$$

By combining (3.19) and (3.20), inequality (3.18) holds. ∎

Remark 3.8. Inequality (3.18) can be rewritten as

$$\|E_k^{amg}\| \leq c\frac{\epsilon}{h^{3/2}}\min\left\{\frac{h}{\sqrt{\epsilon}},1\right\}\frac{\beta\epsilon}{h^2} = c\frac{\epsilon}{h^{3/2}}\left(\frac{\sqrt{\epsilon}}{h}\right)^{\alpha}\beta,$$

where $\alpha = \begin{cases} 1, & h<\sqrt{\epsilon}, \\ 2, & h\geq\sqrt{\epsilon}. \end{cases}$ Comparison with the estimate of E_k^{mg} in (3.12) suggests that algebraic multigrid will converge more rapidly than geometric multigrid if the parameter β in the approximation assumption is less than $(\frac{h}{\sqrt{\epsilon}})^{\alpha}$. Although we have not

TABLE 3.1
Comparison of GMG and AMG for Problem 1 *on uniform rectangular meshes.*

ϵ	10^{-1}	10^{-2}	10^{-3}	10^{-4}
$h=\frac{1}{8}$	5	4	2	2
$h=\frac{1}{16}$	5	4	3	2
$h=\frac{1}{32}$	5	5	4	2

(a) GMG iterative steps.

ϵ	10^{-1}	10^{-2}	10^{-3}	10^{-4}
$h=\frac{1}{8}$	4	3	2	2
$h=\frac{1}{16}$	4	3	3	2
$h=\frac{1}{32}$	5	4	3	2

(b) AMG iterative steps.

yet estimated β for the interpolation operator from AMG coarsening, AMG converges more rapidly than GMG, as shown in Table 3.1. Moreover, in Table 3.1, the facts that both GMG and AMG converge more rapidly as ϵ becomes smaller and converge more slowly as the mesh size h decreases are also consistent with our theoretical analysis. In these numerical studies, the stopping tolerance is set to be

$$\|r_m\| \leq 10^{-6}\,\|r_0\|\,, \tag{3.21}$$

where r_0 is the initial residual and r_m is the residual at the mth iteration.

Remark 3.9. Since the row-sum $\sum_j(a_{i,j})$ and column-sum $\sum_i(a_{i,j})$ are equal for the matrix A_k arising from Problem 1, the following equality holds:

$$\|e\|_1^2 = (A_k e, e) = \sum_{i,j} a_{i,j}e_i e_j = \frac{1}{2}\left(\sum_{i,j}(-a_{i,j})(e_i-e_j)^2\right) + \sum_i s_i e_i^2, \tag{3.22}$$

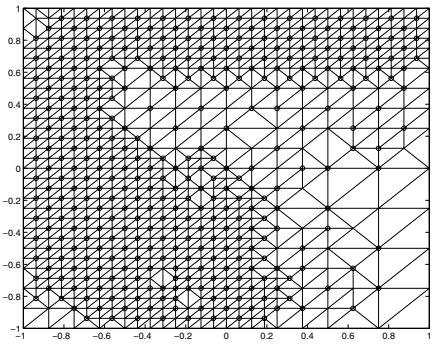
where $s_i = \sum_j(a_{i,j})$. Recall that the algebraically smooth error e_s is characterized by $E_k^s(e_s) \approx e_s$. Clearly, inequalities (3.4) and (3.11) imply

$$\begin{aligned}
&\langle A_k e_s, e_s\rangle \approx \langle A_k E_k^s e_s, E_k^s e_s\rangle \leq \frac{c}{h}\|A_k E_k^s\|\|E_k^s\|\|e_s\|_0^2\\
&\leq c\frac{\varepsilon^2}{h^3}\min\left\{\frac{3h}{\sqrt{\varepsilon}},1\right\}\left(1+\frac{3\varepsilon}{h^2}\min\left\{\frac{3h}{\sqrt{\varepsilon}},1\right\}\right)\|e_s\|_0^2\\
&\leq c\frac{\varepsilon^2}{h^3}\|e_s\|_0^2.
\end{aligned}$$

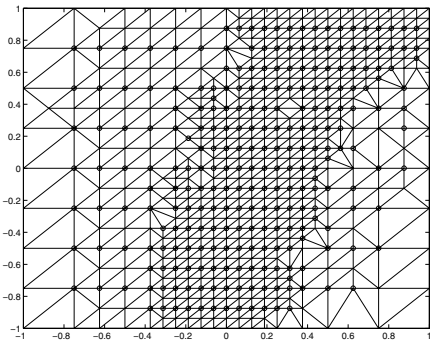
For $h \gg \epsilon^{2/3}$, $\|e_s\|_1^2 \ll \|e_s\|_0^2$ follows from the above estimate. Moreover, by (3.22), this is equivalent to

$$\begin{aligned}
&\frac{1}{2}\sum_i\sum_{j\neq i-1,i+1}(-a_{i,j})(e_i-e_j)^2 + \sum_i s_i e_i^2\\
&\ll \sum_i a_{i,i}e_i^2 + a_{i,i-1}(e_i-e_{i-1})^2 + a_{i,i+1}(e_i-e_{i+1})^2\\
\Rightarrow &\frac{1}{2}\sum_i\sum_{j\neq i-1,i+1}\left(\frac{-a_{i,j}}{a_{i,i}}\right)(e_i-e_j)^2 \ll \sum_i e_i^2 + \frac{1}{4}(e_i-e_{i-1})^2 + \frac{1}{4}(e_i-e_{i+1})^2.
\end{aligned}$$

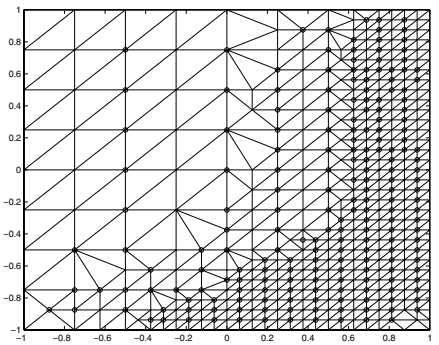
It can be seen from (3.1) that the weights $\frac{a_{i,j}}{a_{i,i}} = O(1)$ for downstream indices j and $O(\frac{\epsilon}{h})$ for upstream. Therefore, we can conclude that the smooth error tends to vary more slowly in downstream directions than in upstream directions. Since the strong



(a) $\phi = -45^\circ$ on an adaptive mesh.



(b) $\phi = 15^\circ$ on an adaptive mesh.



(c) $\phi = 75^\circ$ on an adaptive mesh.

FIG. 3.1. *Coarse grid points selected by AMG coarsening.*

connectivity of the matrix A_k is also along the downstream direction, we expect the AMG coarsening to select a large number of downstream nodes, and the resulting strategy tends to resemble one of semicoarsening.

In the following, we show the AMG coarse grids for the Problem 1 with $\epsilon = 10^{-3}$ and $\phi = -45^\circ$, 15°, and 75°, where the connection parameter is $\mu = 0.25$. Each fine grid is generated by two adaptive mesh refinement steps from an initial 8×8 mesh and the threshold value θ in (2.1) is set to 0.01. In Figure 3.1, the fine grid is plotted and the coarse grid points, selected by AMG coarsening, are marked by $\circ$. Evidently, the AMG coarsening process is sensitive to the flow direction mentioned in Remark 3.9 and generates meshes by means of semicoarsening. The numerical results in Table 3.1, and Table 4.3 and 4.4 of the next section, shows that AMG converges more quickly than GMG for Problem 1. This leads us to conjecture that the algebraically smooth errors are well in the range of the AMG interpolation operator, i.e., the parameter β in the approximation assumption is small. Analysis of the AMG interpolation and approximation assumption will be considered in a future study.

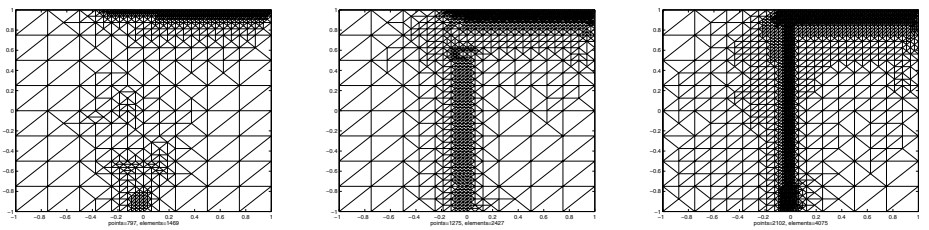
For Problem 2, multigrid convergence is much harder to prove. In fact, the presence of a stagnation point leads to poor performance of multigrid methods when the equation is discretized by a first order upwind scheme [10], [4]. However, many

numerical studies have shown that GMG- and AMG-accelerated Krylov space methods, such as GMRES and BiCGSTAB, achieve nearly mesh-independent convergence [10], [19]. Recently, Ramage has demonstrated that GMRES with GMG preconditioning achieves mesh-independent convergence when SDFEM with an optimal stabilization parameter is employed for discretization on uniform rectangular mesh [13]. In the next section, GMG and AMG are used as a preconditioner for GMRES in some numerical tests.

4. Numerical results. In this section, the performances of different linear solvers, including GMG, AMG, and preconditioned GMRES, for the discrete convection-diffusion equation are compared on both adaptively refined meshes and uniform meshes. We use both Problem 1 with $\phi = 0$ and Problem 2. In both problems, (1.1) is discretized by SDFEM with the stabilization parameter δ_T defined in (1.3) on both a uniform 32×32 mesh and an adaptively refined mesh for $\epsilon = 10^{-2}, 10^{-3}$, and 10^{-4}. Here, a 32×32 mesh is generated from 3 uniform refinements starting with a 4×4 initial mesh, and the adaptively refined mesh is generated by refining an initial 8×8 mesh 4 times, where full multigrid is used with those coarse meshes. The threshold value θ in the maximum marking strategy is chosen such that elements in the regions containing large errors can be refined for both problems.

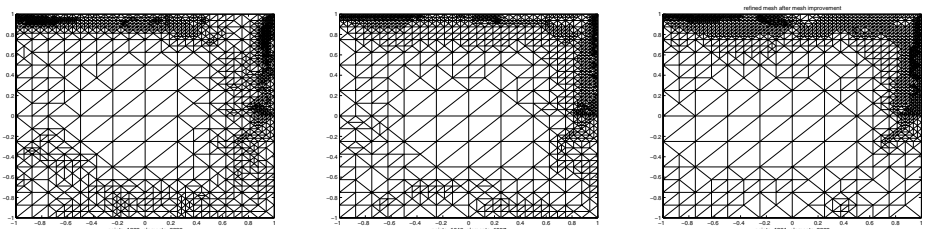
For Problem 1, $\theta = 0.1$, 0.01, and 0.001 for $\epsilon = 10^{-2}, 10^{-3}$, and 10^{-4}, respectively. The adaptive meshes are shown in Figure 4.1. For Problem 2, $\theta = 0.1$ for all ϵ and the adaptive meshes are shown in Figure 4.2.

Since there are no natural "lines" in unstructured meshes or in the coarse meshes of AMG, only point-versions of the Gauss–Seidel method are used for smoothing. Grid points are numbered in lexicographical order where the y-coordinate is the primary



(a) Mesh: $\epsilon = 10^{-2}$. (b) Mesh: $\epsilon = 10^{-3}$. (c) Mesh: $\epsilon = 10^{-4}$.

FIG. 4.1. *Problem* 1: *Adaptively refined meshes after* 4 *refinements starting from an* 8×8 *grid for various* ϵ.



(a) Mesh: $\epsilon = 10^{-2}$. (b) Mesh: $\epsilon = 10^{-3}$. (c) Mesh: $\epsilon = 10^{-4}$.

FIG. 4.2. *Problem* 2: *Adaptively refined meshes for various* ϵ.

key and the x-coordinate is the secondary key. The point Gauss–Seidel method associated with this node ordering is called the forward horizontal Gauss–Seidel (forward-HGS) method. Naturally, one can obtain another lexicographical order where the x-coordinate is the primary key and the y-coordinate is the secondary key. The point Gauss–Seidel method associated with this node ordering is then called the forward vertical Gauss–Seidel (forward-VGS) method. The backward-HGS and backward-VGS methods are obtained from the forward-HGS and forward-VGS methods, respectively, by simply reversing the node ordering. In Problem 1, one step of forward-HGS is used in both presmoothing and postsmoothing. In Problem 2, the smoother consists of four point Gauss–Seidel sweeps (forward-HGS, forward-VGS, backward-HGS, and backward-VGS). Hereafter, this smoother is abbreviated as ADGS (namely, alternating direction Gauss–Seidel).

Both problems are solved by GMG, AMG, GMRES, and right-preconditioned GMRES with zero initial guess, and the stopping tolerance (3.21) is used here for these iterative solvers. The preconditioned GMRES methods with HGS, ADGS, GMG, and AMG preconditioners are denoted as GMRES-HGS, GMRES-ADGS, GMRES-GMG, and GMRES-AMG, respectively. In the following tables, increases in level numbers correspond to finer meshes, where level 1 represents the coarsest mesh in multigrid solver. The notation "-" represents that the number of iterations is greater than 400.

First, Tables 4.1 and 4.2 show some details about the meshes used by GMG and AMG. Table 4.1 shows the number of coarse grid points obtained starting from a uniform 32×32 fine grid of elements. Listed under "GMG" is simply the number of points in the hierarchical set of uniform coarse grids. Listed under "AMG" is the number of grid points derived using the AMG coarsening strategy. Table 4.2 shows what happens when adaptive grid refinement is used. That is, GMG uses the hierarchical set of grids obtained from adaptive refinement. In contrast, AMG starts with the fine grid produced by grid refinement, but then it generates *its own* set of coarse grids using its coarsening strategy. Table 4.1 shows that when AMG starts with a uniform fine mesh, it generates more coarse grid points than are found in the hierarchical coarse grids. On the other hand, Table 4.2 shows that if AMG starts with a fine mesh obtained from adaptive refinement, the coarse grids it generates often contain fewer points than are found in the adaptively refined coarse grids.

Tables 4.3–4.6 examine the iteration counts for both multigrid strategies as a function of the levels at which the solver is used. Tables 4.3 and 4.4 show that both solvers display "textbook" performance on Problem 1 for $\epsilon = 10^{-2}$. As ϵ is reduced, however, performance of GMG decreases and becomes mesh-dependent, especially on uniform grids. In contrast, the performance of AMG is robust and shows weaker

TABLE 4.1
Number of coarse grid points from uniform refinement and AMG coarsening.

	GMG	AMG		
$\log_{10} \frac{1}{\epsilon}$	2, 3, 4	2	3	4
Level = 4	1089	1089	1089	1089
Level = 3	289	480	479	479
Level = 2	81	307	331	231
Level = 1	25	157	108	108

(a) Problem 1.

	GMG	AMG		
$\log_{10} \frac{1}{\epsilon}$	2, 3, 4	2	3	4
Level = 4	1089	1089	1089	1089
Level = 3	289	499	507	513
Level = 2	81	294	283	289
Level = 1	25	174	164	169

(b) Problem 2.

TABLE 4.2
Number of coarse grid points from adaptive refinement and AMG coarsening.

	GMG			AMG		
$\log_{10} \frac{1}{\epsilon}$	2	3	4	2	3	4
Level = 5	797	1275	2102	797	1275	2102
Level = 4	410	649	1047	359	586	997
Level = 3	215	320	528	193	328	547
Level = 2	122	176	239	112	179	308
Level = 1	81	81	81	55	101	153

(a) Problem 1.

	GMG			AMG		
$\log_{10} \frac{1}{\epsilon}$	2	3	4	2	3	4
Level = 5	1233	946	922	1233	946	922
Level = 4	625	632	823	597	496	445
Level = 3	305	373	387	338	291	258
Level = 2	150	196	199	205	168	165
Level = 1	81	81	81	121	95	97

(b) Problem 2.

TABLE 4.3
Problem 1: *Iteration counts for GMG and AMG on uniform meshes.*

Level	GMG	AMG
3	13	7
2	13	6
1	12	6

(a) $\epsilon = 10^{-2}$.

Level	GMG	AMG
3	27	8
2	26	7
1	16	6

(b) $\epsilon = 10^{-3}$.

Level	GMG	AMG
3	51	11
2	35	8
1	17	6

(c) $\epsilon = 10^{-4}$.

TABLE 4.4
Problem 1: *Iteration counts for GMG and AMG on adaptive meshes.*

Level	GMG	AMG
4	9	6
3	8	6
2	7	5
1	7	5

(a) $\epsilon = 10^{-2}$.

Level	GMG	AMG
4	22	6
3	24	6
2	18	6
1	17	6

(b) $\epsilon = 10^{-3}$.

Level	GMG	AMG
4	59	11
3	57	9
2	47	7
1	34	6

(c) $\epsilon = 10^{-4}$.

dependence on ϵ and mesh size. Problem 2, with a recirculating flow, is more difficult for both solvers. On a uniform grid, GMG diverges for $\epsilon = 10^{-4}$, but AMG manages to converge, as shown in Table 4.5. On adaptive meshes, as shown in Table 4.6, textbook-like performance is achieved for both methods. However, AMG is considerably slower than GMG, especially for small $\epsilon = 10^{-4}$ on the finest mesh. Our speculation

TABLE 4.5
Problem 2: *Iteration counts for GMG and AMG on uniform meshes.*

Level	GMG	AMG
3	25	9
2	25	8
1	20	7

(a) $\epsilon = 10^{-2}$.

Level	GMG	AMG
3	179	11
2	135	21
1	43	11

(b) $\epsilon = 10^{-3}$.

Level	GMG	AMG
3	-	52
2	-	43
1	57	13

(c) $\epsilon = 10^{-4}$.

TABLE 4.6
Problem 2: *Iteration counts for GMG and AMG on adaptive meshes.*

Level	GMG	AMG
4	9	12
3	9	11
2	7	10
1	6	7

(a) $\epsilon = 10^{-2}$.

Level	GMG	AMG
4	27	26
3	21	26
2	24	21
1	20	17

(b) $\epsilon = 10^{-3}$.

Level	GMG	AMG
4	30	39
3	32	38
2	40	44
1	30	36

(c) $\epsilon = 10^{-4}$.

TABLE 4.7
Problem 1: *Iteration counts for various GMRES methods on finest grids.*

ϵ	10^{-2}	10^{-3}	10^{-4}
GMG	13	27	51
AMG	7	8	11
GMRES	56	76	95
GMRES-HGS	24	26	36
GMRES-GMG	11	16	22
GMRES-AMG	6	7	9

(a) Iterative steps on uniform mesh.

ϵ	10^{-2}	10^{-3}	10^{-4}
GMG	9	22	59
AMG	6	6	11
GMRES	83	146	-
GMRES-HGS	19	25	48
GMRES-GMG	8	13	27
GMRES-AMG	5	5	9

(b) Iterative steps on adaptive mesh.

is that this is caused by inadequacy of the grid transfer operator used in AMG and the fact that the number of coarse grid points generated by AMG coarsening is only about half the number of coarse grid points used in GMG, as shown in Table 4.2(b). AMG interpolation can be improved by the recently developed AMGe methods [3]. As we will show below, performance can also be remedied using a Krylov space acceleration.

Next, we compare the performance of preconditioned GMRES methods on both the finest uniform mesh and the finest adaptive mesh. Tables 4.7 and 4.8 examine the iteration counts for various versions of preconditioned GMRES. The Krylov space acceleration of the convergence of GMG and AMG is observed in most cases. The numerical results indicate that the AMG-preconditioned GMRES requires the fewest iterations to converge, compared to GMG, AMG, and other versions of GMRES. Particularly in Problem 2, despite the fact that AMG converges more slowly than GMG, GMRES-AMG outperforms GMRES-GMG. This observation suggests that AMG can be a good preconditioner for Krylov space solvers in solving complex flow problems. Moreover, in Table 4.9, iterations of GMRES-AMG on each level of the adaptive meshes are shown for $\epsilon = 10^{-2} \cdots 10^{-4}$. It is clear that the convergence of GMRES-AMG depends only weakly on the mesh size and ϵ on adaptive meshes.

Remark 4.1. Notice a difference between performance of GMG for Problem 1

TABLE 4.8
Problem 2: Iteration counts for various GMRES methods on finest grids.

ϵ	10^{-2}	10^{-3}	10^{-4}
GMG	25	179	-
AMG	9	11	52
GMRES	392	-	-
GMRES-ADGS	33	54	73
GMRES-GMG	11	29	42
GMRES-AMG	6	8	15

(a) Iterative steps on uniform mesh.

ϵ	10^{-2}	10^{-3}	10^{-4}
GMG	9	27	30
AMG	11	26	39
GMRES	199	-	-
GMRES-ADGS	30	37	35
GMRES-GMG	7	12	12
GMRES-AMG	6	8	9

(b) Iterative steps on adaptive mesh.

TABLE 4.9
Iteration counts of GMRES-AMG for various ϵ on adaptive meshes.

Level	$\epsilon =$ 10^{-2}	10^{-3}	10^{-4}
4	5	5	9
3	5	5	8
2	5	5	6
1	4	5	6

(a) Iterative steps in Problem 1.

Level	$\epsilon =$ 10^{-2}	10^{-3}	10^{-4}
4	6	8	9
3	6	7	9
2	5	7	9
1	4	6	7

(b) Iterative steps in Problem 2.

here and the bounds shown in section 3 which suggest convergence is essentially independent of ϵ. The latter result holds only in special settings and depends on having a very accurate smoother. Similar ϵ-independent results for this problem can be found in [9], [12], [14]. In contrast, the better performance of AMG is due to matrix-dependent grid transfer operators and the superior choice of coarse mesh identified by AMG coarsening, for which the smoother is better able to mimic the flow characteristics.

5. Conclusion. In this work, we have explored solution strategies for the convection-diffusion equation using adaptive gridding techniques and a multigrid algorithm. In section 3, the convergence analysis for both GMG with bilinear interpolation and line Gauss–Seidel smoother and AMG with the same smoother not only shows that GMG converges for Problem 1 when $h \gg \epsilon^{2/3}$, but also suggests that AMG converges faster than GMG on uniform rectangular meshes. Although the AMG interpolation and approximation assumption remain to be analyzed, our numerical results support this observation on both triangular and rectangular uniform meshes. In addition, numerical studies in section 4 show that the convergence rates of both GMG and AMG deteriorates as the diffusion parameter ϵ decreases. To overcome this drawback, we have found that Krylov space acceleration significantly improves the convergence of both GMG and AMG, especially when ϵ is small. Moreover, GMRES-AMG converges more rapidly than GMRES-GMG in all of our test cases on both uniform and adaptive meshes.

With adaptive mesh refinement, GMG seems to be a natural choice because the coarse grids, correction operators, and interpolations are ready to be used in GMG. However, GMG convergence is slower than AMG for some examples of both tested problems even though more coarse grid points are employed in GMG. On the other hand, the most important strength of AMG is that it can be treated as a black-box solver. There is no need to compute coarse grids and interpolations explicitly in AMG. The facts that AMG is applicable to a wider range of applications, AMG

converges for all of our test cases, and GMRES-AMG outperforms all other solvers in our numerical studies make AMG very attractive for solving the SDFEM-discretized convection-diffusion problems.

REFERENCES

[1] J. H. Bramble, D. A. Kwak, and J. E. Pasciak, *Uniform convergence of multigrid V-cycle iterations for indefinite and nonsymmetric problems*, SIAM J. Numer. Anal., 31 (1994), pp. 1746–1763.

[2] J. H. Bramble, J. E. Pasciak, and J. Xu, *The analysis of multigrid algorithms for nonsymmetric and indefinite elliptic problems*, Math. Comp., 51 (1988), pp. 398–414.

[3] M. Brezina, A. J. Cleary, R. D. Falgout, V. E. Henson, J. E. Jones, T. A. Manteuffel, S. F. McCormick, and J. W. Ruge, *Algebraic multigrid based on element interpolation (AMGe)*, SIAM J. Sci. Comput., 22 (2000). pp. 1570–1592.

[4] P. M. De Zeeuw, *Matrix-dependent prolongations and restrictions in a blackbox multigrid solver*, J. Comput. Appl. Math., 33 (1990), pp. 1–27.

[5] R. P. Fedorenko, *A relaxation method for solving elliptic difference equations*, USSR Comput. Math. Phys., 1 (1961), pp. 1092–1096.

[6] B. Fischer, A. Ramage, D. Silvester, and A. J. Wathen, *On parameter choice and iterative convergence for stabilised discretisations of advection-diffusion problems*, Comput. Methods Appl. Mech. Engrg., 179 (1999), pp. 179–195.

[7] T. J. R. Hughes, M. Mallet, and A. Mizukami, *A new finite element formulation for computational fluid dynamics:* II. *Beyond SUPG*, Comput. Methods Appl. Mech. Engrg., 54 (1986), pp. 485–501.

[8] D. Kay and D. Silvester, *The reliability of local error estimators for convection-diffusion equations*, IMA J. Numer. Anal, 21 (2001), pp. 107–122.

[9] M. A. Olshanskii and A. Reusken, *Convergence analysis of a multigrid method for a convection-dominated model problem*, SIAM J. Numer. Anal., 42 (2004), pp. 1261–1291.

[10] C. W. Oosterlee and T. Washio, *An evaluation of parallel multigrid as a solver and a preconditioner for singularly perturbed problems*, SIAM J. Sci. Comput., 19 (1998), pp. 87–110.

[11] A. Papastavrou and R. Verfürth, *A posteriori error estimators for stationary convection-diffusion problems: A computational comparison*, Comput. Methods Appl. Mech. Engrg., 189 (2000), pp. 449–462.

[12] I. Persson, K. Samuelsson, and A. Szepessy, *On the convergence of multigrid methods for flow problems*, Electron. Trans. Numer. Anal., 8 (1999), pp. 46–87.

[13] A. Ramage, *A multigrid preconditioner for stabilised discretizations of advection-diffusion problems*, J. Comput. Appl. Math., 110 (1999), pp. 187–203.

[14] A. Reusken, *Convergence analysis of a multigrid method for convection-diffusion equations*, Numer. Math., 91 (2002), pp. 323–349.

[15] M. C. Rivara, *Algorithms for refining triangular grids suitable for adaptive and multigrid techniques*, Internat. J. Numer. Methods Engrg., 20 (1984), pp. 745–756.

[16] M. C. Rivara, *Using longest-side bisection techniques for the automatic refinement of Delaunay triangulation*, Internat. J. Numer. Methods Engrg., 40 (1997), pp. 581–597.

[17] J. W. Ruge and K. Stüben, *Algebraic multigrid*, in Multigrid Methods, Frontiers Appl. Math. 3, S. F. McCormick, ed., SIAM, Philadelphia, 1987, pp. 73–130.

[18] Y. Saad and M. H. Schultz, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 856–869.

[19] U. Trottenberg, C. W. Oosterlee, and A. Schüller, *Multigrid*, Academic Press, San Diego, CA, 2001.

[20] R. S. Varga, *Matrix Iterative Analysis*, Springer-Verlag, Berlin, Heidelberg, 2000.

[21] R. Verfürth, *A posteriori error estimators for convection-diffusion equations*, Numer. Math., 80 (1998), pp. 641–663.

[22] J. Wang, *Convergence analysis of multigrid algorithms for nonselfadjoint and indefinite elliptic problems*, SIAM J. Numer. Anal., 30 (1993), pp. 275–285.

[23] C.-T. Wu, *On the Implementation of an Accurate and Efficient Solver for Convection-Diffusion Equations*, Ph.D. thesis, University of Maryland, College Park, MD, 2003.

SIAM J. SCI. COMPUT.
Vol. 28, No. 6, pp. 2229–2247
© 2006 Society for Industrial and Applied Mathematics

CENTRAL WENO SCHEMES FOR HAMILTON–JACOBI EQUATIONS ON TRIANGULAR MESHES*

DORON LEVY†, SUHAS NAYAK†, CHI-WANG SHU‡, AND YONG-TAO ZHANG§

Abstract. We derive Godunov-type semidiscrete central schemes for Hamilton–Jacobi equations on triangular meshes. High-order schemes are then obtained by combining our new numerical fluxes with high-order WENO reconstructions on triangular meshes. The numerical fluxes are shown to be monotone in certain cases. The accuracy and high-resolution properties of our scheme are demonstrated in a variety of numerical examples.

Key words. Hamilton–Jacobi equations, central schemes, unstructured grids

AMS subject classifications. Primary, 65M06; Secondary, 35L99

DOI. 10.1137/040612002

1. Introduction.

We consider Cauchy problems for Hamilton–Jacobi (HJ) equations of the form

$$\begin{cases} \phi_t(x,t) + H(x,\phi(x,t),\nabla\phi(x,t)) = 0, \quad x \in \Omega \subset \mathbb{R}^2, \\ \phi(x,t=0) = \phi_0(x). \end{cases} \tag{1.1}$$

HJ equations are being used in a variety of applications, such as optimal control problems, differential games, geometric optics, and the calculus of variations.

It is well known that solutions of (1.1) may develop discontinuous derivatives in finite time, and hence it is necessary to interpret solutions of (1.1) in an appropriate weak sense. Such a formulation in terms of the so-called viscosity solutions is due to Crandall, Evans, Ishii, Lions, among others (see [10, 11, 25] and the references therein).

We are interested in approximating solutions of (1.1) on a given conforming triangulation of the domain Ω. While general theory for approximating solutions of (1.1) can be found in the works of Barles, Lions, and Souganidis [4, 33, 26], in this work we focus on Godunov-type schemes for approximating such problems. Godunov-type schemes are based on a global reconstruction which is evolved exactly in time and sampled at the grid nodes at the next time step. A subclass of these schemes are central schemes in which the evolution stage is carried out at points that are located away from the discontinuities. Such a procedure eliminates the necessity of dealing with (generalized) Riemann problems on the cell-interfaces. In addition, systems can be solved componentwise without a characteristic decomposition. These features make central schemes particularly suitable for multidimensional problems and complicated geometries.

*Received by the editors July 20, 2004; accepted for publication (in revised form) June 22, 2006; published electronically December 11, 2006.

http://www.siam.org/journals/sisc/28-6/61200.html

†Department of Mathematics, Stanford University, Stanford, CA 94305-2125 (dlevy@math.stanford.edu, nayak@math.stanford.edu). The work of the first author was supported in part by the National Science Foundation under Career grant DMS-0133511.

‡Division of Applied Mathematics, Brown University, Providence, RI 02912 (shu@dam.brown.edu). The work of this author was supported in part by ARO grant DAAD19-00-1-0405, NSF grant DMS-0207451, and AFOSR grant F49620-02-1-0113.

§Department of Mathematics, University of California, Irvine, CA 92697 (zyt@math.uci.edu).

First- and second-order fully discrete central schemes were introduced by Lin and Tadmor [23, 24] (see also [7]). Their schemes were based on the first-order Lax–Friedrichs scheme [12] and its second-order extension, the Nessyahu–Tadmor scheme [28], for approximating solutions of hyperbolic conservation laws.

Semidiscrete central schemes (on Cartesian meshes) were derived by Kurganov and Tadmor [21]. The main goal there was to reduce the numerical dissipation by estimating the local speeds of propagation of information from the interfaces between neighboring computational cells. The numerical dissipation in these schemes was further reduced by keeping an even more accurate account of the different local speeds [20]. Semidiscrete schemes have a lower numerical dissipation when compared with fully discrete schemes, which make them particularly suitable for long-time simulations, steady-state calculations, and approximating solutions of viscous problems.

The fully discrete schemes of [7, 23, 24] were extended to high-order in [8]. Such extensions were obtained by using suitable weighted essentially nonoscillatory (WENO) reconstructions that were designed following the WENO schemes of Liu, Osher, and Chan [27] and Jiang and Shu [17]. It is important to note that these ideas extend the essentially nonoscillatory (ENO) schemes of Harten et al. [14]. High-order versions of the semidiscrete scheme [21] were obtained by combining the semidiscrete numerical fluxes with WENO reconstructions [9]. This work also provided the theoretical foundation for the monotonicity of the fluxes of [20, 21]. A version of these schemes with reduced numerical dissipation was recently derived in [6].

It is important to note that the progress with central schemes for HJ equations parallels the advances made with upwind schemes for HJ equations. These include the ENO schemes of Osher, Sethian, and Shu [30, 31] and the WENO schemes of Jiang and Peng [16]. Similar methods on triangular meshes include the pioneering works of Abgrall [1, 2], Augoula and Abgrall [3], Barth and Sethian [5], Kossioris, Makridakis, and Souganidis [19], and the recent work of Zhang and Shu [35] on WENO schemes for HJ equations on triangular meshes. The high-order WENO reconstructions on triangular meshes that were used in [35] were based on the results of Hu and Shu [15].

In this paper we present the first semidiscrete *central* scheme for approximating solutions of (1.1) on triangular meshes. Our scheme combines a new numerical flux (that was announced in [22]) with the high-order WENO reconstructions of [35].

We would like to emphasize that the main benefit of the new scheme is in its development as a Godunov-type scheme, in which a global reconstruction is evolved exactly according to the equation. This probably also means that one can expect to obtain for this scheme convergence results in the spirit of the works of Lin and Tadmor [23, 24].

The structure of this paper is as follows: We start in section 2 with the derivation of the numerical flux for HJ equations on triangular meshes. A couple of examples of high-order reconstructions on triangular meshes are then discussed in section 3. Numerical examples that verify the expected order of accuracy of the schemes as well as the high-resolution properties of the resulting solutions are given in section 4. Concluding remarks summarize this work in section 5.

2. Central schemes for Hamilton–Jacobi equations.

2.1. The scheme. To simplify the discussion and the notation, we consider equations of type (1.1) with a Hamiltonian that depends only on $\nabla\phi$, i.e.,

$$\begin{cases} \phi_t(x,t) + H(\nabla\phi(x,t)) = 0, \quad x \in \Omega \subset \mathbb{R}^2, \\ \phi(x, t=0) = \phi_0(x). \end{cases} \tag{2.1}$$

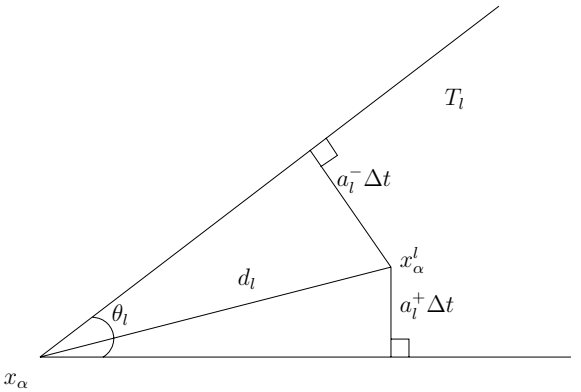


FIG. 2.1. *The evolution point x_α^l that is derived from the maximal local speeds of propagation into T_l, a_l^+, and a_l^-.*

We assume a given triangulation, $\mathfrak{T}$, of Ω, denote the grid points by x_α, and assume that every such point is surrounded by m_α angular sectors T_l^α that are ordered counterclockwise. For simplicity we use the simpler notation $T_l = T_l^\alpha$ (see Figure 2.1).

Given a time-step Δt, we denote the approximate value at time $t^n = n\Delta t$ of $\phi(x_\alpha, t^n)$ by φ_α^n. Assuming that the values of φ_α^n at the grid points x_α are known, we reconstruct a piecewise-polynomial interpolant $\tilde{\varphi}_\alpha$. This interpolant has discontinuous gradients along the cell-interfaces. We denote the approximation of the gradient at x_α that is obtained from the reconstruction in the cell T_l by $\nabla\tilde{\varphi}_{\alpha,l}$. For the purpose of developing the numerical flux there is no need to assume any particular reconstruction. A WENO reconstruction will be described in section 3.

The reconstruction $\tilde{\varphi}_\alpha$ can now be used to estimate the maximal speeds of propagation of information from the cell-interfaces in a direction that is perpendicular to the interfaces. In every sector, T_l, we denote the counterclockwise speed of propagation by a_l^+ and the speed of propagation on the other interface by a_l^- (see Figure 2.1). These speeds can be estimated by

(2.2)
$$a_l^+ = \max\left\{\max_{T_l}\left\{|\nabla H(\nabla\tilde{\varphi}_\alpha(x))\cdot\vec{n}_{l-1,l}|\right\},\ \max_{T_{l-1}}\left\{|\nabla H(\nabla\tilde{\varphi}_\alpha(x))\cdot\vec{n}_{l-1,l}|\right\}\right\},$$

$$a_l^- = \max\left\{\max_{T_l}\left\{|\nabla H(\nabla\tilde{\varphi}_\alpha(x))\cdot\vec{n}_{l+1,l}|\right\},\ \max_{T_{l+1}}\left\{|\nabla H(\nabla\tilde{\varphi}_\alpha(x))\cdot\vec{n}_{l+1,l}|\right\}\right\}.$$

Here, $\vec{n}_{j,l}$ is the normal vector on the interface between T_j and T_l pointing into T_l.

The evolution stage of Godunov-type schemes will be carried out at points x_α^l that are located away from the interfaces, assuming that the time-step is sufficiently small (see Figure 2.1). The distance of the evolution point x_α^l from x_α is denoted by d_l. Clearly, d_l depends on the local speeds of propagation $a_l^\pm$ and on the angle θ_l. A straightforward computation allows us to define d_l as

$$d_l = \Delta t \hat{d}_l, \tag{2.3}$$

where

$$\hat{d}_l^2 = \frac{(a_l^-)^2 + 2a_l^- a_l^+ \cos\theta_l + (a_l^+)^2}{\sin^2\theta_l}. \tag{2.4}$$

We now evolve the interpolant $\tilde{\varphi}(\vec{x}, t^n)$ to the next time-step t^{n+1} at the points x_α^l according to (2.1). Since the evolution points are located away from the propagating discontinuities, the value at the next time-step can be approximated with a first-order Taylor expansion in time, i.e.,

$$\varphi(x_\alpha^l, t^{n+1}) = \tilde{\varphi}(x_\alpha^l, t^n) - \Delta t H(\nabla\tilde{\varphi}(x_\alpha^l, t^n)) + O(\Delta t^2). \tag{2.5}$$

Here, the value of the gradient, $\nabla\tilde{\varphi}(x_\alpha^l, t^n)$, is obtained from the reconstruction, $\tilde{\varphi}$. Expressions of the form (2.5) hold for every evolution point, x_α^l, around x_α.

In order to combine all these values of φ into one value φ_α^{n+1}, we write a convex combination of the values from the different sectors with weights $s_l \geq 0$ that are yet to be determined:

$$\varphi_\alpha^{n+1} = \frac{\sum_{l=1}^{m_\alpha} s_l \varphi(x_\alpha^l, t^{n+1})}{\sum_{l=1}^{m_\alpha} s_l} = \frac{\sum_{l=1}^{m_\alpha} s_l \left[\tilde{\varphi}(x_\alpha^l, t^n) - \Delta t H(\nabla\tilde{\varphi}(x_\alpha^l, t^n)\right]}{\sum_{l=1}^{m_\alpha} s_l}. \tag{2.6}$$

We now define ρ_l to be the unit vector in the direction of x_α^l from x_α and write a Taylor expansion in space:

$$\tilde{\varphi}(x_\alpha^l, t^n) = \tilde{\varphi}(x_\alpha, t^n) + d_l \rho_l \cdot \nabla\tilde{\varphi}(x_\alpha^l, t^n) + O(\Delta t^2).$$

Here by $\nabla\tilde{\varphi}(x_\alpha^l, t^n)$ we refer to the value of the gradient at x_α that is associated with the reconstruction in sector T_l at x_α^l. We may therefore rewrite (2.6) as the fully discrete scheme

$$\varphi_\alpha^{n+1} = \tilde{\varphi}_\alpha^n + \frac{\Delta t}{\sum_{l=1}^{m_\alpha} s_l} \sum_{l=1}^{m_\alpha} s_l \left[\hat{d}_l \rho_l \cdot \nabla\tilde{\varphi}(x_\alpha^l, t^n) - H(\nabla\tilde{\varphi}(x_\alpha^l, t^n))\right]. \tag{2.7}$$

A semidiscrete scheme can now be obtained in the limit $\Delta t \to 0$,

$$\frac{d}{dt}\varphi_\alpha(t) = \lim_{\Delta t \to 0} \frac{\varphi_\alpha^{n+1} - \varphi_\alpha^n}{\Delta t} = \frac{1}{\sum_{l=1}^{m_\alpha} s_l} \sum_{l=1}^{m_\alpha} s_l \left[\hat{d}_l \rho_l \cdot \nabla\tilde{\varphi}_\alpha^l(t) - H(\nabla\tilde{\varphi}_\alpha^l(t))\right], \tag{2.8}$$

where for each l, $\nabla\tilde{\varphi}_\alpha^l(t)$ denotes $\lim_{\Delta t \to 0} \nabla\tilde{\varphi}(x_\alpha^l, t^n)$. All that remains is to determine the coefficients s_l in (2.8). The consistency of the scheme implies that if the value of the gradient is identical in every sector that surrounds x_α, then the numerical Hamiltonian should become the differential Hamiltonian. Hence, we are seeking coefficients s_l, such that

$$\sum_{l=1}^{m_\alpha} s_l \hat{d}_l \rho_l = 0. \tag{2.9}$$

These coefficients can be determined using the results of Abgrall [2]. We denote by $\mu_{l+1/2}$ a unit vector in a direction that is aligned with the interface between the sectors T_l and T_{l+1}, and we assume that $\theta_l < \pi$ (which is indeed the case with a triangulation). It was shown in [2] that

$$\sum_{l=1}^{m_\alpha} \xi_{l+\frac{1}{2}} \mu_{l+\frac{1}{2}} = 0, \tag{2.10}$$

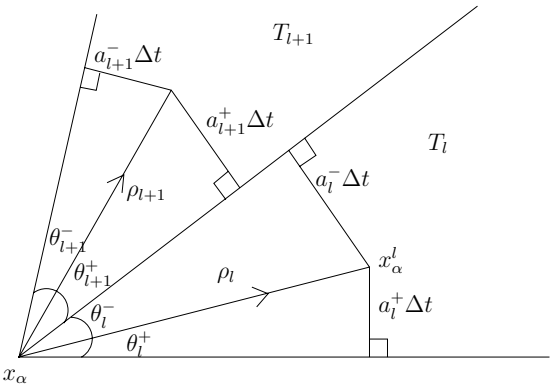


FIG. 2.2. *The angles around* x_α.

provided that

$$\xi_{l+\frac{1}{2}} = \left[\tan\left(\frac{\theta_l}{2}\right) + \tan\left(\frac{\theta_{l+1}}{2}\right)\right]. \tag{2.11}$$

In order to incorporate (2.10)–(2.11) into our framework, we split each angle θ_l into two parts, $\theta_l^{\pm}$, that are defined as

$$\theta_l^{\pm} = \arcsin\frac{a_l^{\pm}}{\hat{d}_l} \tag{2.12}$$

(see Figure 2.2). The consistency condition (2.9) is then satisfied if the weights s_l are taken as $s_l = \beta_l/\hat{d}_l$, where

$$\beta_l = \tan\left(\frac{\theta_l^{+} + \theta_{l-1}^{-}}{2}\right) + \tan\left(\frac{\theta_l^{-} + \theta_{l+1}^{+}}{2}\right).$$

We note that if $\forall l$, $\theta_l^{+} + \theta_{l-1}^{-} \le \pi$, then β_l are guaranteed to be positive. With this notation, a consistent, semidiscrete scheme is given by

$$\frac{d}{dt}\varphi_\alpha(t) = \frac{1}{\sum_{l=1}^{m_\alpha}\frac{\beta_l}{\hat{d}_l}}\sum_{l=1}^{m_\alpha}\beta_l\left[\rho_l\cdot\nabla\tilde{\varphi}_\alpha^l(t) - \frac{H(\nabla\tilde{\varphi}_\alpha^l(t))}{\hat{d}_l}\right]. \tag{2.13}$$

Remarks.

1. It is straightforward to extend scheme (2.13) to more general problems, such as general Hamiltonians of the form present in (1.1) or viscous HJ problems. Examples of these obvious generalizations in the Cartesian case can be found, e.g., in [7, 8, 9, 20, 21].
2. The order of accuracy of the scheme (2.13) is determined by the order of accuracy of the reconstruction $\tilde{\varphi}_\alpha$, and the order of the ODE solver. Due to the known properties of viscosity solutions of HJ equations and the structure of Godunov-type schemes, we assume a global underlying piecewise-smooth reconstruction. We note that, in practice, the final semidiscrete scheme (2.13) uses only values of the gradient computed in the different cells around each grid point x_α. This means that all that we need from the reconstruction are the values of these gradients. High-order reconstructions on triangular meshes are discussed in section 3.

3. There are several ways to simplify the scheme (2.13). One possibility is to replace the different speeds of propagation at every grid point by their maximum, i.e., $a_\alpha = \max_l\{a_l^+, a_l^-\}$. This implies that $\hat{d}_l = a_\alpha/\sin(\theta_l/2)$. In this case (2.13) becomes

$$\frac{d}{dt}\varphi_\alpha(t) = \frac{a_\alpha}{\sum_{l=1}^{m_\alpha}\beta_l \sin\frac{\theta_l}{2}} \sum_{l=1}^{m_\alpha} \beta_l \left[\rho_l \cdot \nabla\tilde{\varphi}_\alpha^l(t) - \frac{\sin\frac{\theta_l}{2}}{a} H(\nabla\tilde{\varphi}_\alpha^l(t))\right]. \tag{2.14}$$

If, in addition, the triangulation of the domain is such that the angles are identical around each point, i.e., $\theta = \theta_l \ \forall\, l$, then (2.14) takes the simpler form

$$\frac{d}{dt}\varphi_\alpha(t) = \frac{1}{m_\alpha} \sum_{l=1}^{m_\alpha} \left[\frac{a_\alpha}{\sin\frac{\theta}{2}} \rho_l \cdot \nabla\tilde{\varphi}_\alpha^l(t) - H(\nabla\tilde{\varphi}_\alpha^l(t))\right]. \tag{2.15}$$

In the special case of a Cartesian grid with equal spacing in the x- and y-directions, the number of angular sectors at each point is $m_\alpha = 4$, and $\sin(\theta/2) = \sin(\pi/4) = \sqrt{2}/2$. If we assume that the velocities are identical in both directions, then (2.15) becomes

$$\begin{aligned}\frac{d}{dt}\varphi_\alpha(t) = {} & \frac{a_\alpha}{2}\left(\varphi_x^+ - \varphi_x^- + \varphi_y^+ - \varphi_y^-\right) \\ & -\frac{1}{4}\left[H\left(\varphi_x^+, \varphi_y^+\right) + H\left(\varphi_x^-, \varphi_y^+\right) + H\left(\varphi_x^+, \varphi_y^-\right) + H\left(\varphi_x^-, \varphi_y^-\right)\right],\end{aligned} \tag{2.16}$$

with the obvious notation; e.g., $H(\varphi_x^+, \varphi_y^+)$ is the Hamiltonian evaluated at the gradient at x_α that is taken from the first quadrant. The scheme (2.16) is identical to the semidiscrete central scheme for Cartesian grids [9, 21].

4. The following Lax–Friedrichs-type scheme on triangular meshes was derived by Abgrall [2]:

$$\frac{d}{dt}\varphi_\alpha(t) = \frac{a}{\pi} \sum_{l=1}^{m_\alpha} \beta_{l+\frac{1}{2}} \mu_{l+\frac{1}{2}} \cdot \left(\frac{\nabla\tilde{\varphi}_\alpha^l(t) + \nabla\tilde{\varphi}_\alpha^{l+1}(t)}{2}\right) - H\left(\frac{\sum_{l=1}^{m_\alpha}\theta_l \nabla\tilde{\varphi}_\alpha^l}{2\pi}\right). \tag{2.17}$$

Here $\mu_{l+1/2}$ is the unit vector in the direction of the interface between the sectors T_l and T_{l+1}, and $\beta_{l+1/2} = \tan(\theta_l/2) + \tan(\theta_{l+1}/2)$. The derivation of (2.17) involved evolution points that were located on the interfaces between the sectors. This resulted in the form of the dissipative term in (2.17) that contains averages of gradients in adjacent sectors. Also, the scheme (2.17) involves a Hamiltonian that is evaluated at the average of the derivatives that are computed in different sectors (with weights that are proportional to the angles). This Hamiltonian term was postulated to be in this form, but could have had other forms. In our case (2.13), this term takes the form of an average over the Hamiltonian that is evaluated in different sectors and is dictated by the derivation of the scheme.

5. We would like to emphasize that the main benefit of the new scheme is in its development as a Godunov-type scheme, in which a global reconstruction is evolved exactly according to the equation. It is a different approach to the "standard" line of schemes for HJ equations in which a monotone flux is

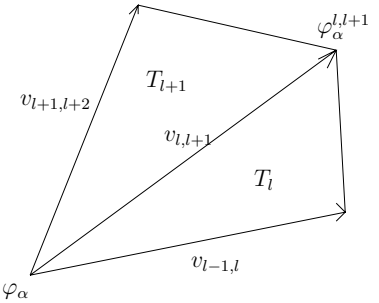


FIG. 2.3. *Grid value at x_α and two triangles containing x_α and grid vectors v.*

combined with a high-order reconstruction. This probably also means that one can expect to obtain for this scheme convergence results in the spirit of the works of Lin and Tadmor [23, 24]. For this purpose, such a construction of a scheme is probably more favorable than traditional constructions.

2.2. Monotonicity. In this section we provide a partial result regarding the monotonicity of our scheme (2.13). Monotonicity is one of the main ingredients in the convergence theory of Barles and Souganidis [4]. Roughly speaking, a stable, consistent, and monotone approximation for an equation that satisfies an underlying comparison principle converges to the viscosity solution.

For simplicity we consider the monotonicity of the scheme in the case where the speeds of propagation do not depend on the gradients of the reconstruction, i.e., they are all equal to a constant that is determined based on a priori bounds. Hence, we assume $a = \max_{l,\pm} a_l^{\pm}$, which also implies that $\theta_l^+ = \theta_l^- = \theta_l/2$.

Monotonicity means that if we rewrite our scheme (2.13) in terms of grid differences,

$$\frac{d}{dt}\varphi_\alpha(t) = F(\varphi_\alpha, \varphi_\alpha - \varphi'), \tag{2.18}$$

where $\varphi_\alpha - \varphi'$ is a vector of grid differences that involve the central node, then F is nonincreasing in all arguments (see, e.g., [4, 11, 29, 31]).

We assume that the gradient of φ in the triangle T_l is constant in that triangle. The normal to the plane connecting the three corners of T_l is given by

$$(v_{l,l+1}^x, v_{l,l+1}^y, \varphi_\alpha^{l,l+1} - \varphi_\alpha) \times (v_{l-1,l}^x, v_{l-1,l}^y, \varphi_\alpha^{l-1,l} - \varphi_\alpha),$$

where $\times$ denotes the cross-product between the vectors. Hence, the gradient of φ in the sector T_l, $\nabla\varphi_\alpha^l$ is given by (see Figure 2.3)

$$\frac{1}{C}\left((\varphi_\alpha - \varphi_\alpha^{l-1,l})v_{l,l+1}^y - (\varphi_\alpha - \varphi_\alpha^{l,l+1})v_{l-1,l}^y, (\varphi_\alpha - \varphi_\alpha^{l,l+1})v_{l-1,l}^x - (\varphi_\alpha - \varphi_\alpha^{l-1,l})v_{l,l+1}^x\right).$$

Here, $C = (v_{l,l+1} \times v_{l-1,l}) \cdot k$ and k is the unit vector pointing out of the plane. The superscripts x and y denote the x- and y-components of the various vectors. We note that C is a negative quantity. Letting $u_1 = \varphi_\alpha - \varphi_\alpha^{l,l+1}$, the derivative of the gradient of φ_α^l with respect to u_1 is given by

$$\frac{\partial}{\partial u_1}\nabla\varphi_\alpha^l = \frac{1}{C}(-v_{l-1,l}^y, v_{l-1,l}^x).$$

This means that

$$\rho_l \cdot \frac{\partial}{\partial u_1} \nabla \varphi_\alpha^l = \frac{1}{C} \rho_l \cdot n_{l-1,l} ||v_{l-1,l}|| = \frac{1}{C} \sin\left(\frac{\theta_l}{2}\right) ||v_{l-1,l}||,$$

where $n_{l-1,l}$ is the unit normal vector pointing into T_l (and is in the plane defined by T_l). Similarly

$$\frac{\partial}{\partial u_1} \nabla \varphi_\alpha^{l+1} = \frac{1}{C'} (v_{l+1,l+2}^y, -v_{l+1,l+2}^x) = \frac{1}{C'} ||v_{l+1,l+2}|| n_{l+2,l+1},$$

where $C' = (v_{l+1,l+2} \times v_{l,l+1}) \cdot k$ and $n_{l+2,l+1}$ is the normal vector from T_{l+2} pointing into T_{l+1}.

Since these are the only two quantities in F that involve u_1, we may evaluate the partial derivative of F with respect to u_1. Letting $K = 1/\sum_{l=1}^{m_\alpha} \frac{\beta_l}{\hat{d}_l}$ (which we note is a positive quantity) we have

$$\begin{aligned}
\frac{\partial F}{\partial u_1} &= \frac{1}{K} \frac{\beta_l ||v_{l-1,l}||}{C \hat{d}_l} \left(\hat{d}_l \rho_l \cdot n_{l-1,l} - \nabla H(\nabla \varphi_\alpha^l) \cdot n_{l-1,l} \right) \\
&\quad + \frac{1}{K} \frac{\beta_{l+1} ||v_{l+1,l+2}||}{C' \hat{d}_{l+1}} \left(\hat{d}_{l+1} \rho_{l+1} \cdot n_{l+2,l+1} - \nabla H(\nabla \varphi_\alpha^{l+1}) \cdot n_{l+2,l+1} \right) \\
&= \frac{1}{K} \frac{\beta_l ||v_{l-1,l}||}{C \hat{d}_l} \left(\hat{d}_l \sin\left(\frac{\theta_l}{2}\right) - \nabla H(\nabla \varphi_\alpha^l) \cdot n_{l-1,l} \right) \\
&\quad + \frac{1}{K} \frac{\beta_{l+1} ||v_{l+1,l+2}||}{C' \hat{d}_{l+1}} \left(\hat{d}_{l+1} \sin\left(\frac{\theta_{l+1}}{2}\right) - \nabla H(\nabla \varphi_\alpha^{l+1}) \cdot n_{l+2,l+1} \right).
\end{aligned} \tag{2.19}$$

We consider only the first term, as an analogous argument holds for the second term. Equation (2.12) implies that $\sin(\theta_l/2) = a/\hat{d}_l$. Also, from (2.2) we know that $\nabla H(\nabla \varphi_\alpha^l) \cdot n_{l-1,l} \le a$, which means that

$$\hat{d}_l \sin\left(\frac{\theta_l}{2}\right) - \nabla H(\nabla \varphi_\alpha^l) \cdot n_{l-1,l} \ge 0.$$

Since $C < 0$ and since we assume that the triangulation is such that the β_l's are positive, we can conclude that the first term in (2.19) is nonpositive. By similar arguments the second term in (2.19) is nonpositive, which means that F is nonincreasing in u_1. The same is true for all the variables of F and hence the scheme is monotone.

3. High-order reconstructions on triangular meshes. In this section we review two third-order reconstructions from [35] (see also [15]). We start with a linear-weights reconstruction. First, we solve an interpolation problem on a large stencil. We then split the large stencil into smaller pieces, obtaining a (low-order) interpolant on each. We conclude with a convex combination of the low-order interpolants that provides the desired (formal) accuracy. The second reconstruction is of WENO type. Here, we replace the linear weights in the convex combination by nonlinear weights. This procedure reduces the spurious oscillations that otherwise develop at singularities. We would like to emphasize that the scheme developed in section 2, i.e., (2.13), does not depend on any particular reconstruction, and the reconstructions described here are provided for the sake of completeness. Extensions to fourth-order are described in [35].

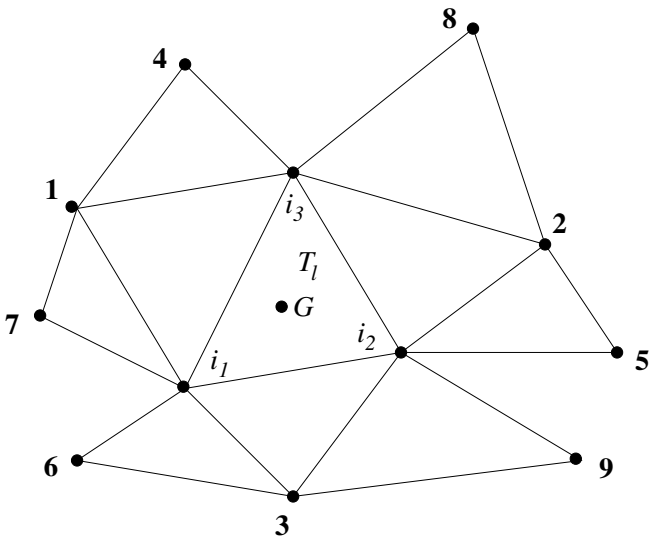


FIG. 3.1. *The nodes used for the large stencil of the third-order reconstruction around T_l. The point G is the barycenter of the cell T_l.*

3.1. A linear-weights reconstruction. We provide an overview of the linear-weights third-order reconstruction from [35]. The problem can be formulated as an interpolation problem, in which the main question is how to choose the interpolation points. The solution is not so obvious, in particular since the geometry of the mesh might not be uniform, which means that stability considerations might imply that it might be better to use different stencils around different triangular sectors.

Assuming a given triangular sector, T_l, the procedure for approximating the components of the derivatives at its three nodes is composed of two steps. First, we look for a large stencil that will provide a stable, third-order approximation of the derivative. There are many ways for choosing such a stencil around any given sector, and we are interested in a compact stencil and a stable reconstruction. In the second step we break the large stencil into several small stencils (all of which are based on grid points that are in the large stencil). Each small stencil provides a second-order approximation of the derivative. These stencils are chosen following compactness and stability considerations. We take sufficiently many such stencils (which will amount to five) such that they can be linearly combined to achieve the desired accuracy of the derivative.

We start by considering an angular sector, T_l, with its three nodes denoted by i_1, i_2, i_3. To obtain a third-order approximation of the derivative at these points, $(\nabla\varphi)_{i_1}, (\nabla\varphi)_{i_2}, (\nabla\varphi)_{i_3}$, we first construct a cubic polynomial. Since we are solving a two-dimensional problem, there are ten free parameters that we have to determine. These will be given by interpolation requirements on a stencil that is yet to be determined. In our numerical interpolation we use normalized coordinates, $(x-x_G)/\sqrt{|T_l|}$, where x_G is the coordinate of the barycenter G and $|T_l|$ is the area of the triangle T_l (see Figure 3.1).

With this in mind, we number the nodes in the neighboring mesh points as $\{1,\dots,9\}$ (in the way that is portrayed in Figure 3.1) and consider the ordered set $W = \{1,\dots,9\}$. The nodes are numbered as in Figure 3.1 in order to avoid biasing the stencil in any particular direction. We note that the geometry of the mesh might

imply that some of these points can be identical. If this is the case, more remote points are added to the list. We omit the details and refer to [35]. We now set a threshold δ and use the following algorithm:

1. Set the interpolation points as $S_0 = \{i_1, i_2, i_3, 1, \ldots, 7\}$.
2. Form the 10×10 interpolation coefficients matrix A and compute its reciprocal condition number, $c(A)$.
3. While $c(A) < \delta$, add the next node in W to S_0 and compute the least squares interpolation coefficients matrix A from the nodes in S_0. Compute $c(A)$.
4. The final S_0 is the large stencil.

Remark. The numerical simulations of [35] showed that at most 12 nodes are needed in order to satisfy the condition $c(A) \geq \delta$ when the threshold is set as $\delta = 10^{-3}$. This is the value that was used in our simulations in section 4.

We denote the interpolation polynomial that is obtained from the stencil S_0 by $p^3(x, y)$. This polynomial, $p^3(x, y)$, can be used to estimate a third-order approximation of the derivatives at the three nodes of T_l. Our goal now is to split the stencil S_0 into several small stencils in such a way that we will be able to recover the third-order accuracy with a convex combination of these smaller stencils. In this case, we will generate five quadratic polynomials, p_s, $s = 1, \ldots, 5$, such that a third-order approximation to the derivatives in the x- and y-directions at $\{i_1, i_2, i_3\}$ will be given by

$$
\begin{aligned}
\frac{\partial}{\partial x}\varphi(x_{i_j}, y_{i_j}) &\approx \frac{\partial}{\partial x}p^3(x_{i_j}, y_{i_j}) = \sum_{s=1}^{5} \gamma_{s,x,i_j} \frac{\partial}{\partial x} p_s(x_{i_j}, y_{i_j}), \qquad j = 1, 2, 3, \\
\frac{\partial}{\partial y}\varphi(x_{i_j}, y_{i_j}) &\approx \frac{\partial}{\partial y}p^3(x_{i_j}, y_{i_j}) = \sum_{s=1}^{5} \gamma_{s,y,i_j} \frac{\partial}{\partial y} p_s(x_{i_j}, y_{i_j}), \qquad j = 1, 2, 3.
\end{aligned} \tag{3.1}
$$

Here, γ_{s,x,i_j} and γ_{s,y,i_j} are the linear weights for the derivatives in the x- and y-directions, respectively. They depend only on the local geometry of the mesh and should satisfy the normalization constraints, $\sum_{s=1}^{5} \gamma_{s,x,i_j} = 1$ and $\sum_{s=1}^{5} \gamma_{s,y,i_j} = 1$. Under these additional conditions, a simple calculation shows that the number of quadratic interpolants has to be greater than or equal to five, which is the reason we set it as five [15, 35].

We are now seeking five small stencils Γ_s, $s = 1, \ldots, 5$, for the target triangle T_l such that $S_0 = \cup_{s=1}^{5} \Gamma_s$. We associate with each such stencil a quadratic polynomial p_s. We note that while the stencils are going to be identical for all the nodes in a given angular sector and in both directions, the linear weights γ_{s,x,i_j} and γ_{s,y,i_j} can be different for each node and for each direction. We summarize the stages given in [35, Procedure 2.2] as follows (for further details see [35]):

1. Obtain a large stencil S_0.
2. For each $s = 1, \ldots, 5$ find a set of candidate small stencils $W_s = \{\Gamma_s^{(r)}, r = 1, \ldots, n_s\}$ in the following way. The nodes i_1, i_2, i_3 are included in every $\Gamma_s^{(r)}$. Let $A_s^{(r)}$ denote the center of $\Gamma_s^{(r)}$, where $A_s^{(r)}$ is given in Table 3.1. Find at least three additional nodes other than i_1, i_2, i_3 such that they have the shortest distance from $A_s^{(r)}$ and the points in $\Gamma_s^{(r)}$ induce an interpolation coefficient matrix A with a good reciprocal condition number ($c(A) \geq \delta$). Based on the experiments in [35] a maximum number of 8 nodes is required to reach the threshold $\delta = 10^{-3}$.
3. Obtain $n_1 \times \cdots \times n_5$ groups of small stencils by taking one small stencil $\Gamma_s^{(r_s)}$ from each W_s, $s = 1, \ldots, 5$. Eliminate the groups that contain the same small stencils and those that do not satisfy the condition $\cup_{s=1}^{5} \Gamma_s^{(r_s)} = S_0$.

TABLE 3.1
The values of $A_s^{(r)}$. The entries refer to the node numbers in Figure 3.1.

s	n_s	$A_s^{(1)}$	$A_s^{(2)}$	$A_s^{(3)}$
1	1	G	–	–
2	3	1	4	7
3	3	2	5	8
4	3	3	6	9
5	≤ 9	$A_5^{(r)}$ are 4–9 and the middle points of 4–8, 5–9, 6–7		

4. For each group $\{\Gamma_s^{(r_s)}, s = 1, \ldots, 5\}$ compute the associated linear weights $\gamma_{s,x,i_j}^{(r_s)}$ and $\gamma_{s,y,i_j}^{(r_s)}$, such that (3.1) is satisfied for $\{i_1, i_2, i_3\}$. Candidate stencils that involve solving ill-posed linear systems are eliminated from further consideration.
5. For the remaining groups, find the minimum value γ_l of all the linear weights $\gamma_{s,x,i_j}^{(r_s)}, \gamma_{s,y,i_j}^{(r_s)}$ on the three vertices $\{i_1, i_2, i_3\}$. The group for which γ_l is the biggest is taken as the final five small stencils for the sector T_l. These five small stencils are denoted by Γ_s, $s = 1, \ldots, 5$.

Remark. Since the linear weights depend on the geometry of the triangulation, some of them might be negative. It is therefore required to take special measures to avoid the stability problems that result from negative weights in the presence of large gradients. In the numerical simulations we apply the technique for handling negative weights in WENO schemes that was recently proposed in [32].

3.2. A WENO reconstruction. In the previous section we showed how to obtain a third-order reconstruction of the derivatives in the x- and y-directions in each grid point. This was done by finding an accurate linear combination of small stencils (each of which results with a second-order reconstruction) such that the overall combination is a third-order reconstruction of the derivative. We now use these results to derive a WENO reconstruction of the derivative. This is done, as usual, by replacing the linear weights by nonlinear weights aiming at reducing the spurious oscillations that might develop in regions that contain discontinuities.

We consider the x-directional derivative at the vertex i of the cell T_l, whose coordinates are (x_i, y_i). Following section 3.1, we denote by $p_s(x, y)$, $s = 1, \ldots, 5$, the sth interpolation polynomial associated with the sth stencil on the cell T_l. A third-order WENO reconstruction for the x-derivative is given in terms of the convex combination

$$(\varphi_x)_i = \sum_{s=1}^{5} w_s \frac{\partial}{\partial x} p_s(x_i, y_i). \tag{3.2}$$

The weights, w_s, associated with $p_s(x, y)$ are given by

$$w_s = \frac{\tilde{w}_s}{\sum_{m=1}^{5} \tilde{w}_m}, \tag{3.3}$$

with

$$\tilde{w}_s = \frac{\gamma_{s,x}}{(\epsilon + \beta_s)^2}, \qquad s = 1, \ldots, 5. \tag{3.4}$$

Here, $\gamma_{s,x}$ is the linear weight associated with stencil s for computing the x-derivative at (x_i, y_i), ϵ is set as a small number to prevent the denominator from vanishing, and β_s is the oscillation indicator associated with the sth stencil.

The oscillatory indicator, β_s, is given by

$$\beta_s = \sum_{|\eta|=2} \int_{T_l} (D^\eta p_s(x,y))^2 dxdy. \tag{3.5}$$

An expression analogous to (3.2) holds for the derivative in the y-direction.

4. Numerical examples. In the numerical examples we use two kinds of triangular meshes. Both are shown in Figure 4.1. The first kind is a "uniform triangular mesh" shown in Figure 4.1 (left). The particular mesh in Figure 4.1 (left) is a coarse mesh with $h = 2/5$, where h is the length of the right-angled side. The second mesh is a "nonuniform triangular mesh" such as the one shown in Figure 4.1 (right). The mesh in the figure is a coarse mesh with $N = 105$ nodes. Refinements of nonuniform meshes are done by splitting each triangle into four similar triangles. The reconstructions we use in all the simulations are the third-order linear and WENO reconstructions from section 3. We refer to the flux (2.13) together with the linear-weights reconstruction of section 3.1 as the "linear scheme." We refer to (2.13) together with the WENO reconstruction of section 3.2 as the "WENO scheme." The ODE solver we used is the third-order strong stability preserving Runge–Kutta (SSP-RK) method [13].

Example 1. A two-dimensional linear equation. Consider the two-dimensional (2D) linear equation

$$\begin{cases} \phi_t + \phi_x + \phi_y = 0, & -2 \le x < 2, -2 \le y < 2, \\ \phi(x,y,0) = \sin(\frac{\pi}{2}(x+y)), \end{cases} \tag{4.1}$$

with periodic boundary conditions.

We solve (4.1) with the linear-weights and the WENO schemes on nonuniform meshes up to time $t = 2$. Since this is a linear problem with constant coefficients, the flux with the local speeds and the flux with the global speeds are identical. The L^1- and L^∞-errors and orders of accuracy, shown in Table 4.1, confirm the expected third-order accuracy.

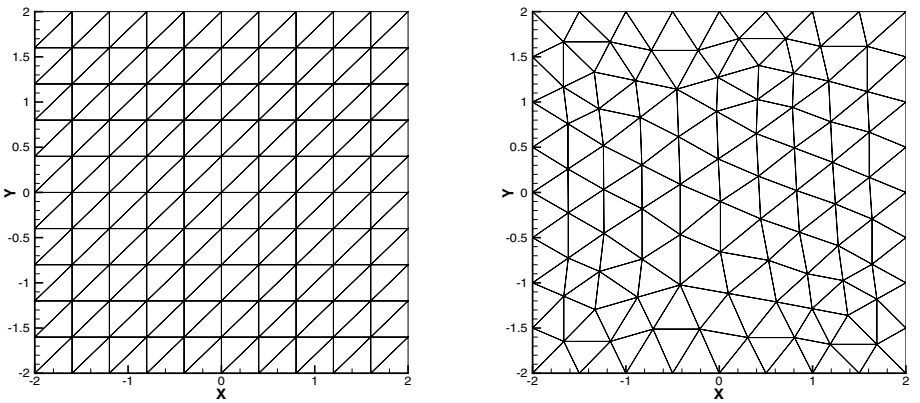


FIG. 4.1. *Left: coarsest uniform mesh with $h = 2/5$. Right: coarsest nonuniform mesh with $N = 105$ nodes.*

TABLE 4.1
Accuracy test for the 2D linear equation (4.1) on nonuniform meshes with the third-order linear and WENO schemes at $t = 2$.

	Linear scheme				WENO scheme			
N	L^1-error	Order	L^∞-error	Order	L^1-error	Order	L^∞-error	Order
105	2.03E-01	—	3.50E-01	—	4.63E-01	—	7.24E-01	—
385	3.16E-02	2.68	6.12E-02	2.51	1.69E-01	1.46	2.62E-01	1.47
1473	4.17E-03	2.92	9.02E-03	2.76	3.32E-02	2.34	7.10E-02	1.88
5761	5.30E-04	2.97	1.20E-03	2.91	5.09E-03	2.70	1.70E-02	2.07
22785	6.68E-05	2.99	1.54E-04	2.96	5.84E-04	3.12	2.53E-03	2.74

TABLE 4.2
Accuracy for the 2D Burgers equation (4.2) on nonuniform meshes with third-order linear and WENO schemes at $t = 0.5/\pi^2$. A global constant speed.

	Linear scheme				WENO scheme			
N	L^1-error	Order	L^∞-error	Order	L^1-error	Order	L^∞-error	Order
105	2.28E-02	—	6.89E-02	—	6.53E-02	—	1.63E-01	—
385	3.80E-03	2.58	1.81E-02	1.93	1.64E-02	2.00	5.94E-02	1.46
1473	5.32E-04	2.83	3.90E-03	2.22	3.50E-03	2.23	1.66E-02	1.84
5761	6.99E-05	2.93	6.99E-04	2.48	5.98E-04	2.55	3.63E-03	2.19
22785	8.96E-06	2.96	8.87E-05	2.98	7.26E-05	3.04	4.95E-04	2.87

TABLE 4.3
Accuracy for the 2D Burgers equation (4.2) on nonuniform meshes with third-order linear and WENO schemes at $t = 0.5/\pi^2$. Local speeds of propagation.

	Linear scheme				WENO scheme			
N	L^1-error	Order	L^∞-error	Order	L^1-error	Order	L^∞-error	Order
105	1.30E-02	—	2.98E-02	—	3.94E-02	—	1.01E-01	—
385	1.87E-03	2.79	6.03E-03	2.31	9.86E-03	2.00	4.95E-02	1.04
1473	2.49E-04	2.91	1.56E-03	1.95	2.24E-03	2.14	1.48E-02	1.74
5761	3.20E-05	2.96	2.23E-04	2.81	3.97E-04	2.50	3.49E-03	2.09
22785	4.05E-06	2.98	3.21E-05	2.80	4.70E-05	3.08	4.88E-04	2.84

Example 2. A 2D Burgers equation. Consider the 2D Burgers equation

$$\begin{cases} \phi_t + \frac{1}{2}(\phi_x + \phi_y + 1)^2 = 0, & -2 \le x < 2, -2 \le y < 2, \\ \phi(x, y, 0) = -\cos\left(\frac{\pi}{2}(x + y)\right), \end{cases} \tag{4.2}$$

augmented with periodic boundary conditions.

We use this example to investigate the difference between fluxes that use a global constant speed and those that use local speeds. A scheme with a global constant speed is obtained from (2.14) when we replace a_α by $a = \max_\alpha a_\alpha$. We solve (4.2) on nonuniform meshes up to time $t = 0.5/\pi^2$. This is before the solution develops any singularities. Table 4.2 shows the L^1- and L^∞-accuracy results that are obtained with the scheme that used a global constant speed. Table 4.3 shows the L^1- and L^∞-errors and orders of accuracy that are obtained when local speeds are taken into account in the numerical flux. In both cases we use linear and WENO schemes. In all cases we verify the expected third-order of accuracy. Indeed, the errors obtained when using a global speed are larger than the error obtained when accounting for local speeds. These results are comparable to those obtained using Abgrall's scheme [2].

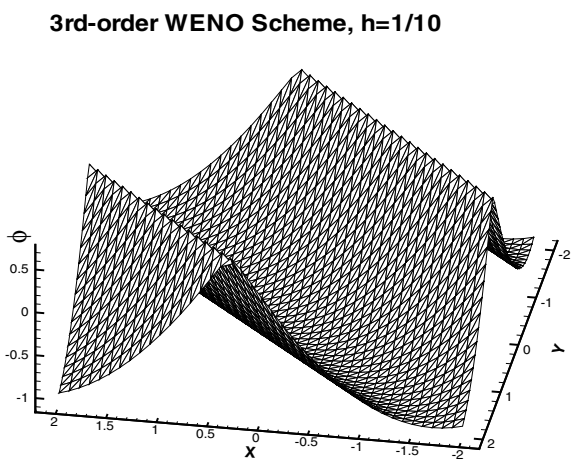


FIG. 4.2. *The 2D Burgers equation* (4.2) *at* $t = 1.5/\pi^2$ *on a uniform mesh with* $h = 1/10$. *The solution is obtained with a third-order WENO scheme and a local speeds flux.*

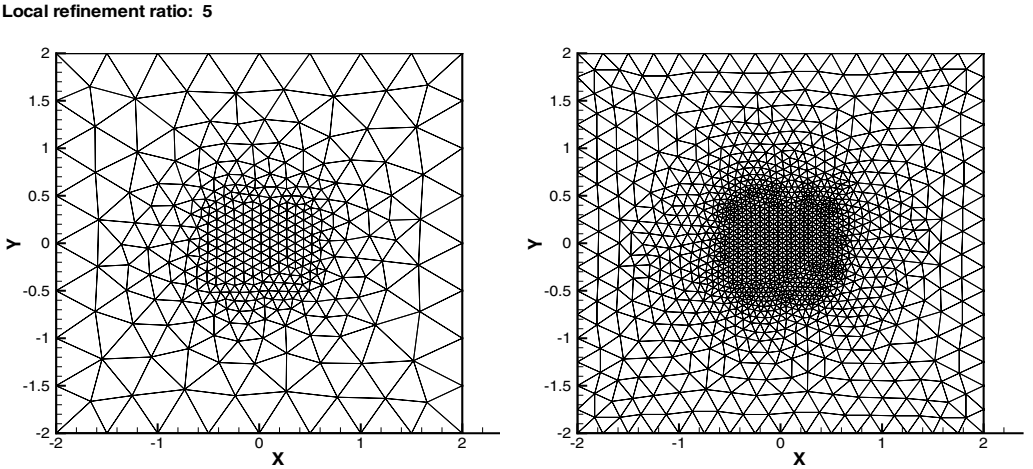


FIG. 4.3. *Left: coarsest very nonuniform mesh with* $N = 353$ *nodes. Right: the refinement of left mesh, with* $N = 1377$ *nodes.*

In Figure 4.2 we plot the solution of (4.2) at time $t = 1.5/\pi^2$, after the solution developed discontinuous derivatives. The solution is obtained with a third-order WENO scheme and a uniform mesh with mesh spacing $h = 1/10$.

We repeat the convergence study, this time using very nonuniform meshes that are shown in Figure 4.3. These meshes are obtained by adding more points in the middle region $[-0.5, 0.5] \times [-0.5, 0.5]$ of the domain $[-2, 2] \times [-2, 2]$.

Table 4.4 includes the accuracy and the minimum and maximum values of local speeds a_l^- for those meshes as shown in Figure 4.3. The same ranges hold for a_l^+. The range of local speeds is from $\approx 10^{-3}$ to ≈ 6.0, which is a relatively large range of such local speeds.

TABLE 4.4

Accuracy and the range of local speeds for the 2D Burgers equation (4.2) on very nonuniform meshes of Figure 4.3 with third-order Linear and WENO schemes at $t = 0.5/\pi^2$. Using the flux with local speeds of propagation.

	Linear scheme				WENO scheme			
N	L^1-error	Order	Min speed	Max speed	L^1-error	Order	Min speed	Max speed
353	7.85E-03	—	1.91E-03	6.04	2.75E-02	—	1.91E-03	5.88
1377	1.11E-03	2.82	1.40E-03	5.93	6.70E-03	2.04	1.40E-03	5.85
5441	1.48E-04	2.90	1.14E-03	5.87	1.48E-03	2.18	1.14E-03	5.85
21633	1.96E-05	2.92	1.01E-03	5.86	2.54E-04	2.54	1.01E-03	5.85

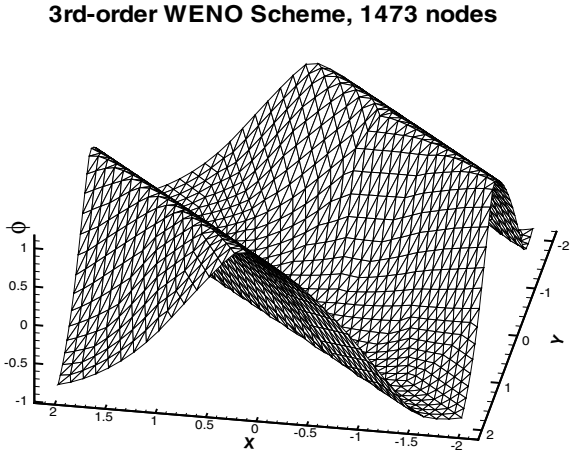


FIG. 4.4. *A 2D HJ equation with a nonconvex flux (4.3) at $t = 1.5/\pi^2$. The solution is obtained with a third-order WENO scheme on a nonuniform mesh with $N = 1473$ nodes and a local speeds flux.*

Example 3. A nonconvex flux. Consider the following HJ equation with a nonconvex flux:

$$\begin{cases} \phi_t - \cos(\phi_x + \phi_y + 1) = 0, & -2 \le x < 2, -2 \le y < 2, \\ \phi(x, y, 0) = -\cos(\frac{\pi(x+y)}{2}), \end{cases} \tag{4.3}$$

augmented with periodic boundary conditions.

In this example we use a nonuniform mesh with $N = 1473$ nodes. At $t = 1.5/\pi^2$ the solution develops a discontinuous derivative. In Figure 4.4 we show the results obtained using the third-order WENO scheme with the local speeds flux.

Example 4. A 2D Riemann problem. Consider the following 2D Riemann problem:

$$\begin{cases} \phi_t + \sin(\phi_x + \phi_y) = 0, & -1 < x < 1, -1 < y < 1, \\ \phi(x, y, 0) = \pi(|y| - |x|). \end{cases} \tag{4.4}$$

We solve (4.4) on a uniform triangular mesh with $40 \times 40 \times 2$ elements. The scheme we use is a third-order WENO scheme with a local speeds flux. Figure 4.5 shows the results obtained at time $t = 1$.

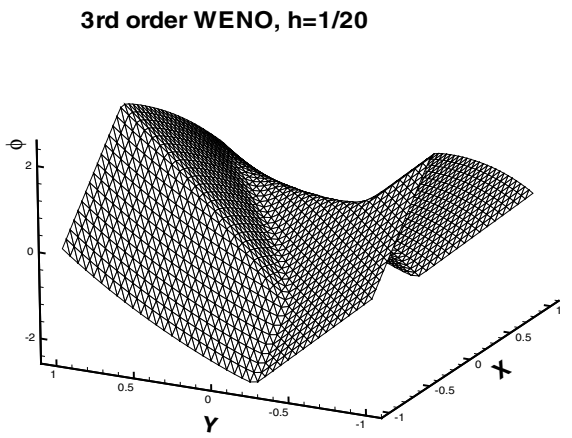


FIG. 4.5. *The 2D Riemann problem* (4.4) *at* $t = 1$. *The solution is obtained with a third-order WENO scheme and a local speeds flux on a uniform triangular mesh with* $h = 1/20$.

Example 5. An optimal control problem. Consider the optimal control problem

$$\begin{cases} \phi_t + (\sin y)\phi_x + (\sin x + \operatorname{sgn}(\phi_y))\phi_y - \frac{1}{2}\sin^2 y - (1-\cos x) = 0, & -\pi < x < \pi, -\pi < y < \pi, \\ \phi(x,y,0) = 0, \end{cases} \tag{4.5}$$

with periodic boundary conditions; see [31]. This problem captures some of the main features and difficulties of control problems, such as the nonsmoothness of the Hamiltonian.

We approximate solutions of (4.5) on a uniform triangular mesh with $40 \times 40 \times 2$ elements using a third-order WENO scheme with a local speeds flux. The solution at $t = 1$ is shown in Figure 4.6 (left). The corresponding optimal control $\omega = \operatorname{sgn}(\phi_y)$ is shown in Figure 4.6 (right).

Example 6. A 2D eikonal equation. Consider the following 2D eikonal equation which arises in geometric optics [18]:

$$\begin{cases} \phi_t + \sqrt{\phi_x^2 + \phi_y^2 + 1} = 0, & 0 \le x < 1, 0 \le y < 1, \\ \phi(x,y,0) = 0.25(\cos(2\pi x) - 1)(\cos(2\pi y) - 1) - 1. \end{cases} \tag{4.6}$$

We approximate solutions of (4.6) using the third-order WENO scheme with a local speeds flux. We use the nonuniform mesh that is shown in Figure 4.7 (left). The solution at $t = 0.6$ is shown in Figure 4.7 (right).

Example 7. A level set equation in a ring. Consider the evolution of the level set function in a ring, as modeled by

$$\begin{cases} \phi_t + sign(\phi_0)(\sqrt{\phi_x^2 + \phi_y^2} - 1) = 0, & \frac{1}{2} < \sqrt{x^2 + y^2} < 1, \\ \phi(x,y,0) = \phi_0(x,y). \end{cases} \tag{4.7}$$

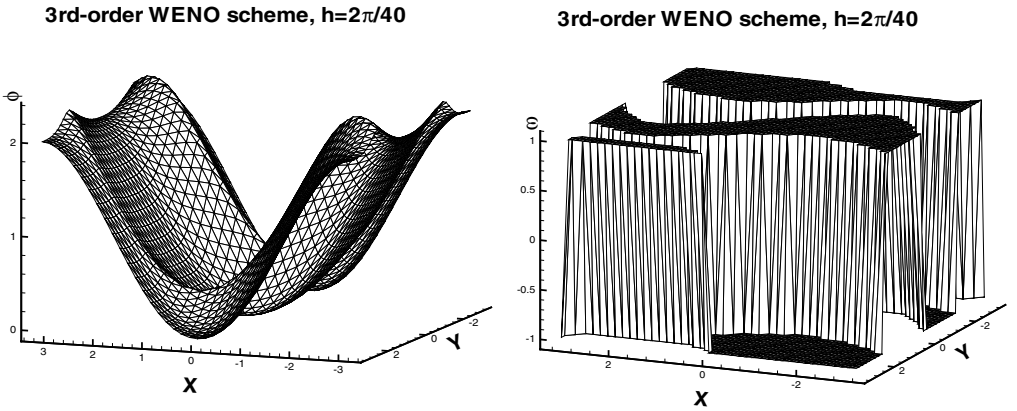


FIG. 4.6. *An optimal control problem* (4.5) *at* $t = 1$ *solved on a uniform triangular mesh with* $h = 2\pi/40$. *The solution is obtained with a third-order WENO scheme and a local speeds flux. Left: the solution. Right: the optimal control* $\omega = sgn(\phi_y)$.

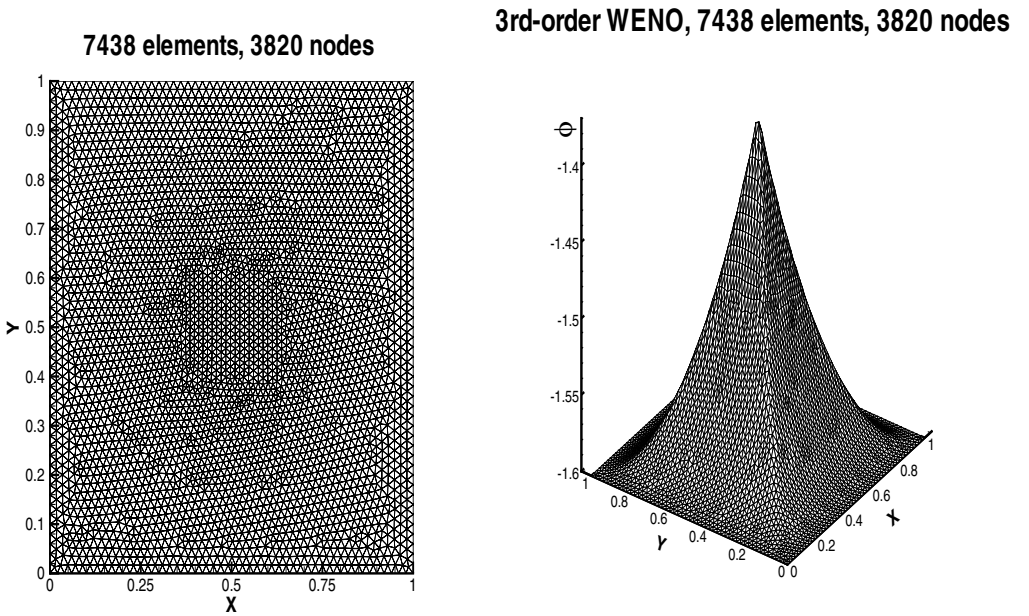


FIG. 4.7. *The* 2*D eikonal equation* (4.6). *Left: the nonuniform mesh used in this example. Right: the solution at* $t = 0.6$ *obtained with the third-order WENO scheme with a local speeds flux.*

This problem is from [34]. The solution ϕ to (4.7) has the same zero level set as ϕ_0, and the steady-state solution is the distance function to that zero level curve. In this example, the exact steady-state solution is the distance function to the inner boundary of the domain. We compute the time-dependent problem to reach the steady-state solution. The exact solution of the steady state is the boundary condition. The mesh with 1420 triangles is shown in Figure 4.8 (left), and the numerical result on that mesh using our third-order scheme is shown in Figure 4.8 (right).

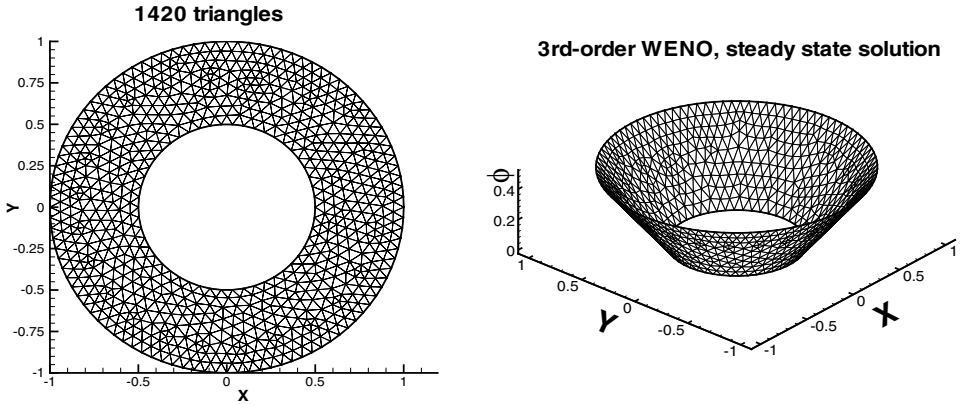


FIG. 4.8. *Solving the level set equation* (4.7) *in a ring. Left: the mesh with* 1420 *triangles. Right: the steady-state solution obtained with our third-order scheme.*

5. Conclusion. In this paper we derived the first central scheme for Hamilton–Jacobi equations on unstructured grids. Similarly to any other Godunov-type method, this scheme is obtained by an exact evolution of a reconstruction which is then projected back onto the mesh points. The order of accuracy of the scheme is determined by the accuracy of the reconstruction and the accuracy of the ODE solver. The reconstructions we used were the third-order linear and WENO reconstructions on triangular meshes [35].

While we have proved the monotonicity of the scheme (with a piecewise-linear reconstruction) in the special case of a global constant local speeds of propagation, we believe that the scheme is monotone in the general case (for a proper choice of the speeds where some global bounds are taken into consideration, see [31]). This point remains as an open problem which we hope to address in the future.

Acknowledgment. We would like to thank Adam Oberman for helpful discussions.

REFERENCES

[1] R. ABGRALL, *On essentially nonoscillatory schemes on unstructured meshes: Analysis and implementation*, J. Comput. Phys., 114 (1994), pp. 45–54.
[2] R. ABGRALL, *Numerical discretization of the first-order Hamilton–Jacobi equation on triangular meshes*, Commun. Pure Appl. Math., 49 (1996), pp. 1339–1373.
[3] S. AUGOULA AND R. ABGRALL, *High order numerical discretization for Hamilton–Jacobi equations on triangular meshes*, J. Sci. Comput., 15 (2000), pp. 197–229.
[4] G. BARLES AND P. E. SOUGANIDIS, *Convergence of approximation schemes for fully nonlinear second order equations*, Asymptot. Anal., 4 (1991), pp. 271–283.
[5] T. BARTH AND J. SETHIAN, *Numerical schemes for the Hamilton–Jacobi and level set equations on triangulated domains*, J. Comput. Phys., 145 (1998), pp. 1–40.
[6] S. BRYSON, A. KURGANOV, D. LEVY, AND G. PETROVA, *Semi-discrete central-upwind schemes with reduced dissipation for Hamilton–Jacobi equations*, IMA J. Numer. Anal., 25 (2005), pp. 87–112.
[7] S. BRYSON AND D. LEVY, *Central schemes for multidimensional Hamilton–Jacobi equations*, SIAM J. Sci. Comput., 25 (2003), pp. 767–791.
[8] S. BRYSON AND D. LEVY, *High-order central WENO schemes for multidimensional Hamilton–Jacobi equations*, SIAM J. Numer. Anal., 41 (2003), pp. 1339–1369.
[9] S. BRYSON AND D. LEVY, *High-order semi-discrete central-upwind schemes for multidimensional Hamilton–Jacobi equations*, J. Comput. Phys., 189 (2003), pp. 63–87.

[10] M. G. Crandall, H. Ishii, and P.-L. Lions, *User's guide to viscosity solutions of second order partial differential equations*, Bull. Amer. Math. Soc., 27 (1992), pp. 1–67.
[11] M. G. Crandall and P.-L. Lions, *Two approximations of solutions of Hamilton–Jacobi equations*, Math. Comp., 43 (1984), pp. 1–19.
[12] K. O. Friedrichs and P. D. Lax, *Systems of conservation equations with a convex extension*, Proc. Nat. Acad. Sci., 68 (1971), pp. 1686–1688.
[13] S. Gottlieb, C.-W. Shu, and E. Tadmor, *Strong stability-preserving high-order time discretization methods*, SIAM Rev., 43 (2001), pp. 89–112.
[14] A. Harten, B. Engquist, S. Osher, and S. Chakravarthy, *Uniformly high order accurate essentially nonoscillatory schemes* III, J. Comput. Phys., 71 (1987), pp. 231–303.
[15] C. Hu and C.-W. Shu, *Weighted essentially nonoscillatory schemes on triangular meshes*, J. Comput. Phys., 150 (1999), pp. 97–127.
[16] G.-S. Jiang and D. Peng, *Weighted ENO schemes for Hamilton–Jacobi equations*, SIAM J. Sci. Comput., 21 (2000), pp. 2126–2143.
[17] G.-S. Jiang and C.-W. Shu, *Efficient implementation of weighted ENO schemes*, J. Comput. Phys., 126 (1996), pp. 202–228.
[18] S. Jin and Z. Xin, *Numerical passage from systems of conservation laws to Hamilton–Jacobi equations, and relaxation schemes*, SIAM J. Numer. Anal., 35 (1998), pp. 2385–2404.
[19] G. Kossioris, Ch. Makridakis, and P. E. Souganidis, *Finite volume schemes for Hamilton–Jacobi equations*, Numer. Math., 83 (1999), pp. 427–442.
[20] A. Kurganov, S. Noelle, and G. Petrova, *Semidiscrete central-upwind schemes for hyperbolic conservation laws and Hamilton–Jacobi equations*, SIAM J. Sci. Comput., 23 (2001), pp. 707–740.
[21] A. Kurganov and E. Tadmor, *New high-resolution semi-discrete schemes for Hamilton–Jacobi equations*, J. Comput. Phys., 160 (2000), pp. 241–282.
[22] D. Levy and S. Nayak, *Central schemes for Hamilton–Jacobi on unstructured grids*, in Proceedings of the 5th European Conference on Numerical Mathematics and Advanced Applications (ENUMATH 2003), Prague, Czech Republic, Springer-Verlag, Berlin, 2004, pp. 623–630.
[23] C.-T. Lin and E. Tadmor, *L^1-stability and error estimates for approximate Hamilton–Jacobi solutions*, Numer. Math., 87 (2001), pp. 701–735.
[24] C.-T. Lin and E. Tadmor, *High-resolution nonoscillatory central schemes for Hamilton–Jacobi Equations*, SIAM J. Sci. Comput., 21 (2000), pp. 2163–2186.
[25] P. L. Lions, *Generalized Solutions of Hamilton–Jacobi Equations*, Pitman, London, 1982.
[26] P. L. Lions and P. E. Souganidis, *Convergence of MUSCL and filtered schemes for scalar conservation laws and Hamilton–Jacobi equations*, Numer. Math., 69 (1995), pp. 441–470.
[27] X.-D. Liu, S. Osher, and T. Chan, *Weighted essentially nonoscillatory schemes*, J. Comput. Phys., 115 (1994), pp. 200–212.
[28] H. Nessyahu and E. Tadmor, *Nonoscillatory central differencing for hyperbolic conservation laws*, J. Comput. Phys., 87 (1990), pp. 408–463.
[29] A. M. Oberman, *Convergent difference schemes for degenerate elliptic and parabolic equations: Hamilton–Jacobi equations and free boundary problems*, SIAM J. Numer. Anal., 44 (2006), pp. 879–895.
[30] S. Osher and J. Sethian, *Fronts propagating with curvature dependent speed: Algorithms based on Hamilton–Jacobi formulations*, J. Comput. Phys., 79 (1988), pp. 12–49.
[31] S. Osher and C.-W. Shu, *High-order essentially nonoscillatory schemes for Hamilton–Jacobi equations*, SIAM J. Numer. Anal., 28 (1991), pp. 907–922.
[32] J. Shi, C. Hu, and C.-W. Shu, *A technique of treating negative weights in WENO schemes*, J. Comput. Phys., 175 (2002), pp. 108–127.
[33] P. E. Souganidis, *Approximation schemes for viscosity solutions of Hamilton–Jacobi equations*, J. Differential Equations, 59 (1985), pp. 1–43.
[34] M. Sussman, P. Smereka, and S. Osher, *A level set approach for computing solution to incompressible two-phase flow*, J. Comput. Phys., 114 (1994), pp. 146–159.
[35] Y.-T. Zhang and C.-W. Shu, *High-order WENO schemes for Hamilton–Jacobi equations on triangular meshes*, SIAM J. Sci. Comput., 24 (2003), pp. 1005–1030.

SIAM J. SCI. COMPUT.
Vol. 28, No. 6, pp. 2248–2265
© 2006 Society for Industrial and Applied Mathematics

SEMI-IMPLICIT COVOLUME METHOD IN 3D IMAGE SEGMENTATION*

S. CORSARO†, K. MIKULA‡, A. SARTI§, AND F. SGALLARI¶

Abstract. We introduce a three-dimensional (3D) semi-implicit complementary volume numerical scheme for solving the level set formulation of (Riemannian) mean curvature flow problem. We apply the scheme to segmentation of objects (with interrupted edges) in 3D images. The method is unconditionally stable and efficient regarding computational times. The study of its experimental order of convergence on 3D examples shows its second order accuracy for smooth solutions and first order accuracy for highly singular solutions with vanishing gradients as arising in image segmentation.

Key words. partial differential equations, numerical methods, image segmentation, semi-implicit scheme, complementary volume method

AMS subject classifications. 65M60, 65M12, 68U10

DOI. 10.1137/060651203

1. Introduction.

In this paper we introduce a fast and stable computational method for a three-dimensional (3D) image segmentation based on a solution of the following Riemannian mean curvature flow equation:

$$u_t = \sqrt{\varepsilon^2 + |\nabla u|^2}\nabla . \left(g(|\nabla G_\sigma * I^0|) \frac{\nabla u}{\sqrt{\varepsilon^2 + |\nabla u|^2}} \right). \tag{1.1}$$

Here, $u(t,x)$ is an unknown; we call it *segmentation function* or *subjective surface* [31, 32, 33], defined in $Q_T \equiv [0,T] \times \Omega$. The computational domain $\Omega \subset \mathbb{R}^d$, $d = 3$, is bounded by a Lipschitz continuous boundary $\partial\Omega$. The endpoint of the interval $[0,T]$ represents a time when the final segmentation result is obtained. In practice it is chosen a posteriori; the segmentation is stopped when a change of solution in time (e.g., in L_2-norm) is less than a prescribed threshold. The Riemannian mean curvature flow (1.1) is weighted by a function g giving edge detection information from the segmented image I^0. The function $g : \mathbb{R}_0^+ \to \mathbb{R}^+$ is nonincreasing, $g(0) = 1$, admitting $g(s) \to 0$ for $s \to \infty$, e.g., $g(s) = 1/(1 + Ks^2)$, $K \geq 0$ [29]. $G_\sigma \in C^\infty(\mathbb{R}^d)$ is a smoothing kernel, e.g., the Gauss function, which is used in presmoothing of image gradients by the convolution. We will use the notation $g^0 = g(|\nabla G_\sigma * I^0|)$, and, due to properties of convolution, $1 \geq g^0 \geq \delta_\sigma > 0$ [8, 17]. The model parameter $1 \geq \varepsilon > 0$ shifts the model from the mean curvature flow of graphs ($\varepsilon = 1$) to the mean curvature flow of level sets ($\varepsilon = 0$) [15]. For segmentation of objects without corrupted boundaries (e.g., due to a noise or occlusion) the choice $\varepsilon = 1$ is optimal.

*Received by the editors January 30, 2006; accepted for publication (in revised form) June 27, 2006; published electronically December 11, 2006. This work was supported by NATO Collaborative Linkage grant PST.CLG.979123, and by grants VEGA 1/3321/06 and APVT-20-040902.
http://www.siam.org/journals/sisc/28-6/65120.html

†CNR, Institute for High-Performance Computing and Networking (ICAR), Via P. Castellino 111, 80131 Napoli, Italy (stefania.corsaro@dma.unina.it).

‡Department of Mathematics, Slovak University of Technology, Radlinského 11, 813 68 Bratislava, Slovakia (mikula@vox.svf.stuba.sk).

§DEIS, University of Bologna, Via Risorgimento 2, 40136 Bologna, Italy (asarti@deis.unibo.it).

¶Department of Mathematics, University of Bologna, Piazza di Porta S. Donato 5, 40127 Bologna, Italy (sgallari@dm.unibo.it).

When segmenting an object with gaps in encompassing edges, then smaller ε, close to zero, should be used.

Together with the variational approaches following the seminal idea of Mumford and Shah [25], the curve and surface evolution models are powerful techniques used in image segmentation [1, 19, 5, 22, 6, 7, 20, 24]. Equation (1.1) can be formally understood as an Evans–Spruck ε-regularization [15]

$$|\nabla u| \approx |\nabla u|_\varepsilon = \sqrt{\varepsilon^2 + |\nabla u|^2} \tag{1.2}$$

of the segmentation equation

$$u_t = |\nabla u| \nabla . \left(g^0 \, \frac{\nabla u}{|\nabla u|} \right), \tag{1.3}$$

which is called geodesic or conformal mean curvature flow equation in the level set formulation [6, 7, 20].

In [15], the ε-regularization (1.2) was used as a tool to prove existence of the viscosity solution [12, 10] of the mean curvature motion in the level set formulation given by (1.3) with $g^0 \equiv 1$ [28]. The basic idea is that if the motion of a graph by mean curvature is rescaled by a factor $\frac{1}{\varepsilon}$, and letting $\varepsilon \to 0$, it gives the evolution of level sets. Equations (1.3) and (1.1), respectively, were studied analytically in [20, 7] and [32, 11], respectively.

In (1.3), where $0 < g^0 \leq 1$, the motion of level sets is influenced by the image features expressed in g^0. In the case $\varepsilon = 1$, (1.1) gives the motion of a graph by mean curvature with respect to a specific Riemann metric depending on g^0. Rescaling this motion by $\frac{1}{\varepsilon}$ (or, correspondingly, rescaling the Riemann metric) helps to complete the interrupted edges. The idea to use such special Riemannian mean curvature flow of graphs to extract the so-called subjective contours [18] and to segment images with interrupted edges originates in [31, 32, 33]; such an approach is called the *subjective surface method*. Unlike in (1.3), the subjective surface method does not move one particular (e.g., zero) level set to the boundary of the segmented object, but it moves all the level sets there. The forming discontinuity (shock) in the evolving graph of solution of (1.1) is used to detect object boundaries. Thus the standard level set methods preventing discontinuity formation by reinitialization of the front and applying, e.g., higher order (ENO or WENO) schemes cannot be used here.

Concerning numerical solution, in all previous approaches, the level set type equation (1.3), respectively (1.1), was rewritten into the advection-diffusion form

$$u_t = g^0 |\nabla u| \nabla . \left(\frac{\nabla u}{|\nabla u|} \right) + \nabla g^0 . \nabla u, \tag{1.4}$$

where $|\nabla u|_\varepsilon$ is used instead of $|\nabla u|$ in the case of (1.1). Then this form of equation was solved by a finite difference scheme using upwind in the advection term and explicit time stepping in the curvature term [6, 7, 20, 31, 32, 33]. As it will be discussed in section 3.3, such an approach is computationally inefficient mainly due to severe stability restrictions both in advection and curvature parts.

In spite of the previous methods based on (1.4), our 3D covolume technique suggested in this paper relies on discretization of (1.3) or, more precisely, of its ε-regularization (1.1). Using formulation (1.1) and employing its variational structure we develop an unconditionally stable semi-implicit time discretization scheme which gives the image segmentation in a computationally fast way. The main goal of this paper is to suggest such a method for high-scale 3D image segmentation problems and

show its stability, computational efficiency, and experimental convergence properties. Since the method can be used for any $\varepsilon > 0$ with $\varepsilon \to 0$, it can be applied to efficient solution of level set flows also in any other mean curvature driven applications.

The covolume technique (also called complementary volume or finite volume element method) is a combination of finite element (see, e.g., [4]) and finite volume methods; see, e.g., [21, 23]. The discrete equations are derived using the finite volume methodology, i.e., integrating equation in the so-called control (complementary, finite) volume. The control volumes are constructed as elements of a dual (complementary) grid to a primal finite element triangulation (tetrahedral grid in the 3D case). Employing the methodology of the finite element method, the piecewise linear representation of solution on a tetrahedral grid is used to evaluate in a clear and simple way the nonlinear quantities, the absolute value of gradients, in the numerical solution of (1.1).

A two-dimensional (2D) covolume technique for solving the level set formulation of mean curvature flow was first introduced in [35]. There also a particular nonlinear fully implicit time discretization scheme was given. The nonlinear scheme was designed in such a way that the basic properties of a continuous solution, namely L_∞- and $W^{1,1}$-stability, are fulfilled also for discrete solutions.

In [16] a linear semi-implicit 2D covolume level set method was given and studied. The same basic stability properties were proven for the semi-implicit scheme. Since it solves in every discrete time step linear (instead of nonlinear) algebraic systems, it gives solution in a more efficient way. In [16], the semi-implicit scheme was applied to 2D image filtering problem represented by the nonlinear diffusion level set type equation [2]

$$u_t = g(|\nabla G_\sigma * u|)|\nabla u| \nabla . \left(\frac{\nabla u}{|\nabla u|} \right). \tag{1.5}$$

In this paper we present for the first time a semi-implicit covolume scheme in three dimensions for solving level set type problems and apply the scheme to a different high-scale application—3D image segmentation. The previous 2D covolume strategy is not straightforwardly extended to the 3D case, but a new primal-dual grid construction is developed respecting discrete voxel structure of 3D images. Our new covolume scheme is related to 3D image data representation known in computer graphics as a centered-cubic-lattice method [9]. From the numerical point of view, it avoids any grid orientation effect, which is important for simulation of curve and surface evolution on a fixed grid; we illustrate this phenomenon in section 3.3. Although (1.5), studied in [16], and (1.1) and (1.3), respectively, look similar, their structure is different. In (1.1) and (1.3) the weighting term $g^0 = g(|\nabla G_\sigma * I^0|)$ is inside the divergence which brings a new advective phenomenon into the model; cf. (1.4). The advective term given by the vector field $-\nabla g^0$ is very important in the segmentation context; it drives all level sets to the image edges. Without the convective term (let us think about (1.5) with g^0 instead of $g(|\nabla G_\sigma * u|)$ in the right-hand side), the model is unable to segment nonconvex objects, as discussed in section 3.3, and so the method from [16] cannot be applied to image segmentation. Fortunately, the semi-implicit covolume strategy handles the weighting term inside divergence naturally, without going to advection-diffusion form (1.4), and allows us to develop a fast and stable numerical scheme for 3D image segmentation.

The paper is organized as follows. In section 2 we introduce in detail our 3D semi-implicit covolume method. In section 3 we study its experimental order of convergence in nontrivial examples of the level set evolution and thus its reliability in

simulations. We also discuss segmentation experiments with artificial and real (medical) 3D images and compare the method with other approaches. In this paper we concentrate mainly on algorithmic and computational aspects of the method. A full theoretical convergence analysis is not a simple task, and it is out of scope of this paper; however, it is a subject of our current research.

2. Computational method. We present our method formally in the discretization of (1.3), although we always use its ε-regularization (1.1) with a specific $\varepsilon > 0$. The notation is simpler in the case of (1.3), and it will be clear where regularization appears in the numerical scheme.

Equation (1.1) is accompanied either with Dirichlet boundary conditions

$$u(t,x) = u^D \quad \text{in } [0,T] \times \partial\Omega, \tag{2.1}$$

or Neumann boundary conditions

$$\frac{\partial u}{\partial s}(t,x) = 0 \quad \text{in } [0,T] \times \partial\Omega, \tag{2.2}$$

where s is unit normal to $\partial\Omega$, and with the initial condition

$$u(0,x) = u^0(x) \quad \text{in } \Omega. \tag{2.3}$$

In the image segmentation we use Dirichlet boundary conditions, and without loss of generality we may assume $u^D = 0$. The zero Neumann boundary conditions are often used in computations of interface motions in free boundary problems (see, e.g., [34, 27]) or in morphological image smoothing (see, e.g., [2, 16]). In this paper, we use Dirichlet boundary conditions in computing 3D image segmentations and in comparisons with known exact solutions u. The zero Neumann boundary conditions are used here in testing the accuracy of the method when dealing with the known motion of a particular level set of function u without knowing the exact evolution of the entire function u. In 3D image segmentation, the computational domain Ω is usually a subdomain of the image domain; it should include the segmented object. Along with the boundary conditions an initial condition (2.3) is also prescribed, and we assume that the initial function is bounded, i.e., $u^0 \in L_\infty(\Omega)$.

2.1. Semi-implicit time discretization. First, we choose a uniform discrete time step τ and a variance σ of the smoothing kernel G_σ. Then we replace the time derivative in (1.3) by backward difference. The nonlinear terms of the equation are treated from the previous time step, while the linear ones are considered on the current time level; this means semi-implicitness of the time discretization. By such an approach we get our semi-discrete in time scheme:

Let τ and σ be fixed numbers, g^0 be given by an image I^0, and u^0 be a given initial segmentation function. Then, for every discrete time moment $t_n = n\tau$, $n = 1, \dots N$, we look for a function u^n, solution of the equation

$$\frac{1}{|\nabla u^{n-1}|}\frac{u^n - u^{n-1}}{\tau} = \nabla.\left(g^0\ \frac{\nabla u^n}{|\nabla u^{n-1}|}\right). \tag{2.4}$$

2.2. Covolume spatial discretization in three dimensions. A 3D digital image is given on a structure of voxels with cubic shape, in general. Since discrete values of image intensity I^0 are given in voxels and they influence the model, we will

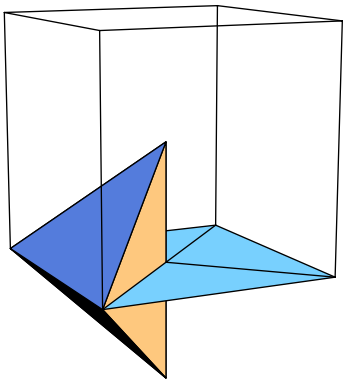


FIG. 2.1. *A tetrahedron of our primal grid together with its intersection with the bottom face of voxel covolume is shown.*

relate spatially discrete approximations of the segmentation function u also to the voxel structure, more precisely, to voxel centers. In every discrete time step t_n of the method (2.4) we have to evaluate gradient of the segmentation function at the previous step $|\nabla u^{n-1}|$. Toward that goal we put a 3D tetrahedral grid into the voxel structure and take a piecewise linear approximation of the segmentation function on such a grid. Such an approach will give a constant value of gradient in tetrahedra (which is the main feature of the covolume [35, 16] and linear finite element [13, 14] methods in solving mean curvature flow in the level set formulation), allowing simple, clear, and fast construction of a fully discrete system of equations.

The formal construction of our covolumes will be given in the next paragraph, and we will see that the covolume mesh corresponds back to the image voxel structure, which is reasonable in image processing applications. On the other hand, the construction of covolume mesh has to use a 3D tetrahedral finite element grid to which it is complementary. This will be possible using the following approach. First, every cubic voxel is split into six pyramids with vertex given by the voxel center and base surfaces given by the voxel boundary faces. The neighboring pyramids of neighboring voxels are joined together to form an octahedron which is then split into four tetrahedra using diagonals of the voxel boundary face—see Figure 2.1. In such a way we get our 3*D tetrahedral grid*. Two nodes of every tetrahedron correspond to centers of neighboring voxels, and further two nodes correspond to voxel boundary vertices; every tetrahedron intersects the common face of neighboring voxels. Let us note that due to our image processing application in mind, we restrict all further considerations only to this type of grid. In our method, only the centers of voxels will represent degree of freedom nodes (DF nodes); i.e., solving the equation at a new time step we update the segmentation function only in these DF nodes. Additional nodes of tetrahedra will not represent degrees of freedom; we will call them nondegree of freedom nodes (NDF nodes), and they will be used in piecewise linear representation of segmentation function. Let a function u be given by discrete values in the voxel centers, i.e., in DF nodes. Then in NDF nodes we take the average value of the neighboring DF nodal values. By such defined values in NDF nodes a piecewise linear approximation u_h of u on the tetrahedral grid is built.

For the tetrahedral grid $\mathcal{T}_h$, given by the previous construction, we construct a covolume (dual) mesh. We modify the approach given in [35, 16] in such a way that

our covolume mesh will consist of cells p associated only with DF nodes p of $\mathcal{T}_h$, let us say $p = 1, \ldots, M$. Since there will be one-to-one correspondence between covolumes and DF nodes, to avoid any confusion, we use the same notation for them. For each DF node p of $\mathcal{T}_h$ let C_p denote the set of all DF nodes q connected to the node p by an edge. This edge will be denoted by σ_{pq} and its length by h_{pq}. Then every *covolume* p is bounded by the planes e_{pq} that bisect and are perpendicular to the edges $\sigma_{pq}, q \in C_p$. By this construction, if e_{pq} intersects σ_{pq} in its center, the covolume mesh corresponds exactly to the voxel structure of the image inside the computational domain Ω where the segmentation is provided. Then the covolume boundary faces do cross in NDF nodes. So we can also say that NDF nodes correspond to zero-measure covolumes, and thus they do not add additional equations to discrete model, and they do not represent degrees of freedom in the covolume method. We denote by $\mathcal{E}_{pq}$ the set of tetrahedra having σ_{pq} as an edge. In our situation (see Figure 2.1) every $\mathcal{E}_{pq}$ consists of four tetrahedra. For each $T \in \mathcal{E}_{pq}$ let c_{pq}^T be the area of the portion of e_{pq} that is in T, i.e., $c_{pq}^T = m(e_{pq} \cap T)$, where m is the measure in $\mathbb{R}^{d-1}$. Let $\mathcal{N}_p$ be the set of all tetrahedra that have DF node p as a vertex. Let u_h be a piecewise linear function on $\mathcal{T}_h$. We will denote a constant value of $|\nabla u_h|$ on $T \in \mathcal{T}_h$ by $|\nabla u_T|$ and define regularized gradients by

$$|\nabla u_T|_\varepsilon = \sqrt{\varepsilon^2 + |\nabla u_T|^2}. \tag{2.5}$$

We will use notation $u_p = u_h(x_p)$, where x_p is the coordinate of DF node p of $\mathcal{T}_h$.

With these notations, we are ready to derive covolume spatial discretization. As it is usual in finite volume methods [21, 23], we integrate (2.4) over every covolume p, $i = 1, \ldots, M$, and using divergence theorem we get an integral formulation

$$\int_p \frac{1}{|\nabla u^{n-1}|} \frac{u^n - u^{n-1}}{\tau} dx = \sum_{q \in C_p} \int_{e_{pq}} \frac{g^0}{|\nabla u^{n-1}|} \frac{\partial u^n}{\partial \nu} ds. \tag{2.6}$$

The exact "fluxes" $\int_{e_{pq}} \frac{g^0}{|\nabla u^{n-1}|} \frac{\partial u^n}{\partial \nu} ds$ in the right-hand side and "capacity function" $\frac{1}{|\nabla u^{n-1}|}$ in the left-hand side (see, e.g., [21]) will be approximated numerically using piecewise linear reconstruction of u^{n-1} on triangulation $\mathcal{T}_h$. If we denote g_T^0 approximation of g^0 on a tetrahedron $T \in \mathcal{T}_h$, then for the approximation of the right-hand side of (2.6) we get

$$\sum_{q \in C_p} \left(\sum_{T \in \mathcal{E}_{pq}} c_{pq}^T \frac{g_T^0}{|\nabla u_T^{n-1}|} \right) \frac{u_q^n - u_p^n}{h_{pq}}, \tag{2.7}$$

and the left-hand side of (2.6) is approximated by

$$M_p m(p) \frac{u_p^n - u_p^{n-1}}{\tau}, \tag{2.8}$$

where $m(p)$ is the measure in $\mathbb{R}^d$ of covolume p and M_p is an approximation of the capacity function inside the finite volume p. For that goal we use either the averaging of the gradients proposed by Walkington [35], i.e.,

$$M_p = \frac{1}{|\nabla u_p^{n-1}|}, \quad |\nabla u_p^{n-1}| = \sum_{T \in \mathcal{N}_p} \frac{m(T \cap p)}{m(p)} |\nabla u_T^{n-1}|, \tag{2.9}$$

or

$$M_p = \sum_{T \in \mathcal{N}_p} \frac{m(T \cap p)}{m(p)} \frac{1}{|\nabla u_T^{n-1}|}, \tag{2.10}$$

which is close to finite element approximation with the mass lumping. Then the regularization of both approximations of the capacity function is given either by

$$M_p^\varepsilon = \frac{1}{|\nabla u_p^{n-1}|_\varepsilon} \tag{2.11}$$

or by

$$M_p^\varepsilon = \sum_{T \in \mathcal{N}_p} \frac{m(T \cap p)}{m(p)} \frac{1}{|\nabla u_T^{n-1}|_\varepsilon}, \tag{2.12}$$

and if we define coefficients (where the ε-regularization is taken into account)

$$b_p^{n-1} = M_p^\varepsilon m(p), \tag{2.13}$$

$$a_{pq}^{n-1} = \frac{1}{h_{pq}} \sum_{T \in \mathcal{E}_{pq}} c_{pq}^T \frac{g_T^0}{|\nabla u_T^{n-1}|_\varepsilon}, \tag{2.14}$$

we get from (2.7)-(2.8) our fully discrete semi-implicit covolume scheme:
Let u_p^0, $p = 1, \dots, M$, be given discrete initial values of the segmentation function. Then, for $n = 1, \dots, N$ we look for u_p^n, $p = 1, \dots, M$, satisfying

$$b_p^{n-1}\ u_p^n + \tau \sum_{q \in C_p} a_{pq}^{n-1}(u_p^n - u_q^n) = b_p^{n-1}\ u_p^{n-1}. \tag{2.15}$$

Applying either Neumann or Dirichlet boundary conditions, our fully discrete scheme gives the system of linear equations with the matrix which is a symmetric and diagonally dominant M-matrix. Using a similar technique as in [16] we get the following.

THEOREM. *There exists unique solution $(u_1^n, \dots, u_M^n)$ of the scheme (2.15) for any $\tau > 0$, $\varepsilon > 0$ and for every $n = 1, \dots, N$. The system matrix is a symmetric and diagonally dominant M-matrix. For any $\tau > 0$, $\varepsilon > 0$ the following L_∞ stability estimate holds:*

$$\min_p\ u_p^0 \le \min_p\ u_p^n \le \max_p\ u_p^n \le \max_p\ u_p^0, \quad 1 \le n \le N. \tag{2.16}$$

One can solve the system (2.15) by any efficient preconditioned linear iterative solver suitable for sparse, symmetric, diagonally dominant M-matrices, e.g., by SOR (successive overrelaxation) or PCG (preconditioned conjugate gradient) methods [30]. In the experiments presented in section 3 we use the SOR method. The iterative process is stopped if $R^{(l)} < \text{TOL}\ R^{(0)}$, where $R^{(l)}$ represents the squared residual in the lth iteration. Reporting experimental order of convergence we use TOL=10^{-6}. In the segmentation one can increase TOL, e.g., to 10^{-2}, in order to use fewer SOR iterations and consequently to decrease overall segmentation time. No significant influence to a precision of the segmentation result is usually observed.

The evaluation of g_T^0 included in coefficients (2.14) can be done in several ways. First, we may replace the convolution by the weighted average to get $I_\sigma^0 := G_\sigma * I^0$

(see, e.g., [23]) and then relate discrete values of I^0_σ to voxel centers. Then, as above, we may construct its piecewise linear representation on a grid and get a constant value of $g^0_T \equiv g(|\nabla I^0_\sigma|)$ on every tetrahedron $T \in \mathcal{T}_h$. Another possibility is to solve a numerically linear heat equation for time t corresponding to variance σ with initial datum given by I^0 (see, e.g., [4]) by the same method as above. The convolution represents a preliminary smoothing of the data. It is also a theoretical tool to have bounded gradients and thus a strictly positive weighting coefficient g^0. In practice, the evaluation of gradients on a fixed discrete grid (e.g., described above) always gives bounded values. So, working on a fixed grid, one can also avoid the convolution, especially if preliminary denoising is not needed or not desirable. Then it is possible to work directly with gradients of piecewise linear representation of I^0 in evaluation of g^0_T.

3. Discussion of numerical results. In this section we study experimental order of convergence of the method in case of mean curvature driven level set flow (i.e., for (1.1) with $g \equiv 1$ and $\varepsilon \to 0$), and then we present some 3D segmentation examples. The comparison with nontrivial exact solutions shows the second order accuracy for smooth (or mildly singular) solutions and the first order accuracy for highly singular solutions (i.e., when gradient is vanishing on a large subset of a domain and discontinuity set of the gradient field is nontrivial). It means that the method is accurate in computing interface motions, and it is reliable also for computing evolutions including flat regions as arising in the subjective surface based segmentation method.

3.1. Experimental order of convergence. In the first example we test the method using the exact solution [26]

$$u(x, y, z, t) = (x^2 + y^2 + z^2 - 1)/4 + t \tag{3.1}$$

of the level set equation

$$u_t = |\nabla u| \nabla . \left(\frac{\nabla u}{|\nabla u|} \right) \tag{3.2}$$

and consider Dirichlet boundary conditions given by this exact solution.

This problem, and all further computed examples in this subsection, are solved in the spatial domain $\Omega = [-1.25, 1.25]^3$ and in the time interval $T = 0.16$. We have taken subsequent grid refinement with $M = n^3$ covolumes (DF nodes), $n = 10, 20, 40, 80, 160$. So the grid size is $h = 2.5/n$. The time step τ is chosen proportionally to h^2 and we measure errors in $L_\infty((0, T), L_2(\Omega))$-norm (cf. [13]), which is natural for testing the schemes for solving parabolic problems.

Let us assume that the error of the scheme in some norm is proportional to some power of the grid size, i.e., $Error(h) = Ch^\alpha$, with a constant C. Then halving the grid size we have $Error(h/2) = C(h/2)^\alpha$ from where we can simply extract

$$\alpha = \log_2(Error(h)/Error(h/2)). \tag{3.3}$$

The α is called the *experimental order of convergence* (EOC) and can be determined by comparing numerical solutions and exact solutions on subsequently refined grids.

In Table 3.1 we report errors in $L_\infty((0, T), L_2(\Omega))$-norm for refined grids and for several choices of ε. In all cases we observe $\alpha = 2$, where the coupling $\varepsilon \approx h^2$ seems optimal (see also other examples), but as one can see, choosing ε even smaller can get even smaller errors on coarse grids.

TABLE 3.1
Errors in $L_\infty((0,T),L_2(\Omega))$-norm, and EOC comparing numerical and exact solution (3.1).

n	h	$\varepsilon = h^2$	EOC	$\varepsilon = h$	EOC	$\varepsilon = 10^{-6}$	EOC
10	0.25	2.9208e-2		6.5190e-2		2.4936e-2	
20	0.125	5.5133e-3	2.40	2.0893e-2	1.64	5.2251e-3	2.25
40	0.0625	1.3106e-3	2.07	5.4823e-3	1.93	1.2939e-3	2.01
80	0.03125	3.2371e-4	2.01	1.3679e-3	2.00	3.2270e-4	2.00
160	0.015625	8.0548e-5	2.00	3.4086e-4	2.00	8.0485e-5	2.00

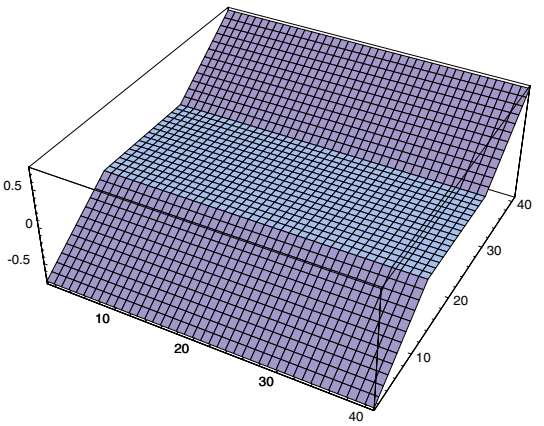


FIG. 3.1. *Exact solution* (3.4) *remaining unchanged under mean curvature flow (plot at $z = 0$).*

Next an interesting example comes from [14] and is given by (see Figure 3.1)

$$
\begin{aligned}
u(x,y,z,t) &= x + 0.5, \ x \le -0.5 \\
&= 0, \ -0.5 \le x \le 0.5 \\
&= x - 0.5, \ x \ge 0.5.
\end{aligned} \tag{3.4}
$$

The level sets of u are planes, with zero mean curvature, and so the solution remains unaltered by the flow. There is also a large part of the solution where gradient is vanishing; however, the set of discontinuity in gradient is relatively simple (two planes) regarding a possible orientation of our grid (we can make it parallel). Mathematically, it is a trivial example, but it is a good test for a numerical scheme. One can observe (see Table 3.2) that the error of the scheme in this example is proportional just to regularization parameter ε, and thus it can be made as small as desirable. This is a simple consequence of the *consistency* of our scheme in the sense that it gives exact solution for any linear initial function and for any choice of regularization parameter ε, any grid size h, and any size of time step τ. Such a property can be checked by inspection of the scheme (2.15). For the linear function, the gradients in all tetrahedra are the same, and so we get the system with the same structure as given by the backward Euler scheme for solving a linear diffusion equation with constant coefficients, and such a scheme does not alter a steady state.

In the next example we compare our numerical solution with the highly singular solution given by

$$u(x,y,z,t) = \min((x^2 + y^2 + z^2 - 1)/4 + t, 0). \tag{3.5}$$

The initial function and numerical result at $T = 0.16$ are plotted in Figure 3.2. We

TABLE 3.2
Errors in $L_\infty((0,T),L_2(\Omega))$-norm, and EOC comparing numerical and exact solution (3.4).

n	h	Error, $\varepsilon = h^2$	EOC	Error, $\varepsilon = 10^{-6}$
10	0.25	5.2252e-2		0.9114e-6
20	0.125	1.5088e-3	1.79	1.0309e-6
40	0.0625	4.0474e-3	1.89	1.0648e-6
80	0.03125	1.0118e-3	2.00	1.0473e-6

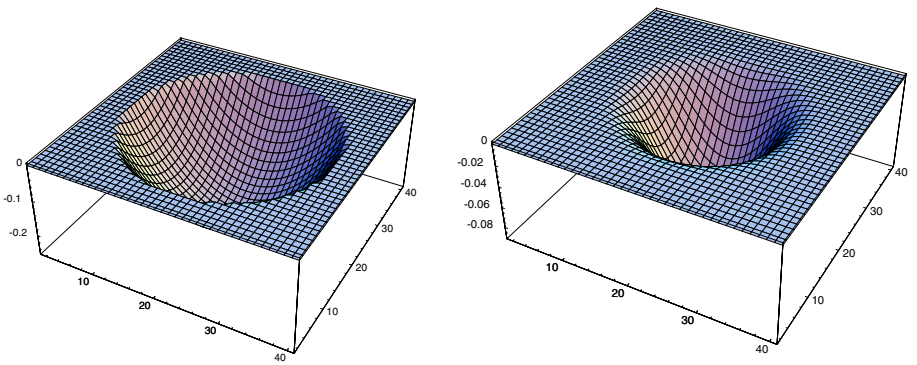


FIG. 3.2. *Numerical solution corresponding to exact solution* (3.5) *plotted at times $t = 0$ (left) and $t = 0.16$ and for $z = 0$.*

TABLE 3.3
Errors in $L_\infty((0,T),L_2(\Omega))$-norm, and EOC comparing numerical and exact solution (3.5).

n	h	Error, $\varepsilon = h^2$	EOC
10	0.25	6.9571e-2	
20	0.125	4.2686e-2	0.70
40	0.0625	2.2049e-2	0.95
80	0.03125	1.1030e-2	0.99
160	0.015625	5.5544e-3	0.99

see only slight smoothing in numerical solution along singularity; numerical solution converges to viscosity solution with $\alpha = 1$, as reported in Table 3.3.

In the last example we test experimental order of convergence comparing the numerical and exact evolution of one particular level set. Namely, we use the exact solution given by shrinking sphere with exact radius $r(t) = \sqrt{r(0) - 4t}$, starting with $r(0) = 1$. Our initial level set function is a 3D cone (signed distance function) having zero value on the unit sphere. We consider zero Neumann boundary conditions, and so we do not know the exact solution for evolution of the whole level set function. At every time step we evaluate $L_2(S^2)$-norm of error, where S^2 is the unit sphere, comparing the exact shrinking sphere and the numerical zero level set, and then we take $L_\infty((0,T),L_2(S^2))$-norm for the overall error in time interval $T = 0.16$. Figure 3.3 represents the initial sphere and the numerical zero level set at $T = 0.16$. Table 3.4 documents the higher order rate of convergence, where again the coupling $\varepsilon = h^2$ is used. Similarly to the first example, different couplings or choice of a fixed small ε give similar errors and convergence rates.

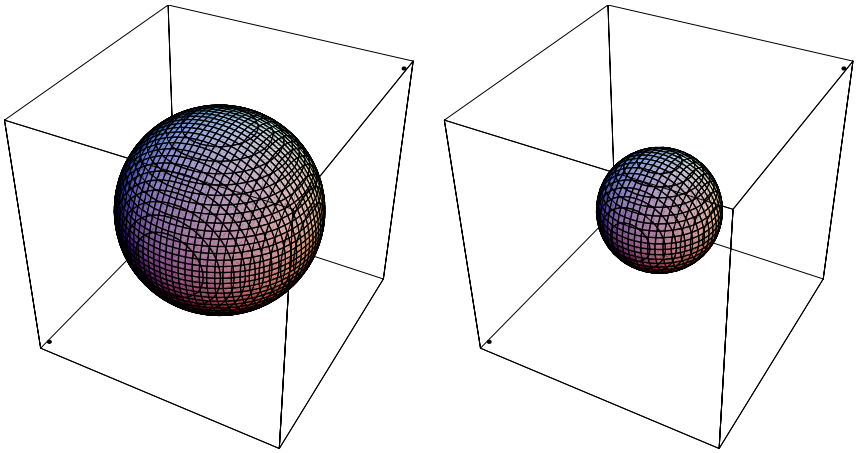


FIG. 3.3. *Numerical solution corresponding to exact shrinking sphere with radius* $r(t) = \sqrt{r(0) - 4t}$, $r(0) = 1$, *plotted at times* $t = 0$ *(left) and* $t = 0.16$.

TABLE 3.4
Errors in $L_\infty((0,T), L_2(S^2))$*-norm, and EOC comparing numerical zero level sets and the exact shrinking sphere.*

n	h	Error, $\varepsilon = h^2$	EOC
10	0.25	7.7228e-2	
20	0.125	2.8732e-2	1.42
40	0.0625	7.8269e-3	1.87
80	0.03125	4.7901e-4	4.03
160	0.015625	9.9568e-5	2.26

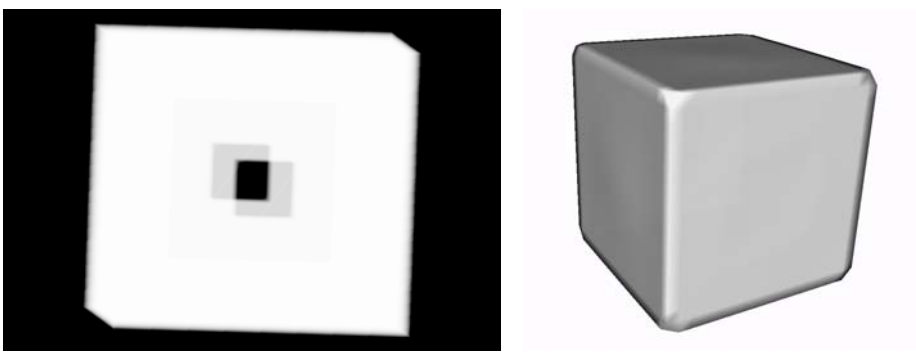


FIG. 3.4. *Segmentation (right) of a* 3*D cube with two holes (left).*

3.2. Segmentation examples. Here we present two 3D segmentation examples. The first one is simple; in the image with resolution 40^3 voxels is an object, a cube with two holes (see Figure 3.4(left)); the holes are in front and back faces. Figure 3.4 gives a reconstructed surface with the perfect hole filling (see Figure 3.4(right)). Due to the advection term all level sets are attracted to the faces of the cube. Due to the mean curvature flow type diffusion the level sets continue inside the gaps in the form of planar surfaces. Since in the holes area $g^0 \equiv 1$ and $\nabla g^0 \equiv 0$ just the mean curvature motion of level sets plays the role inside the gaps, and the planar

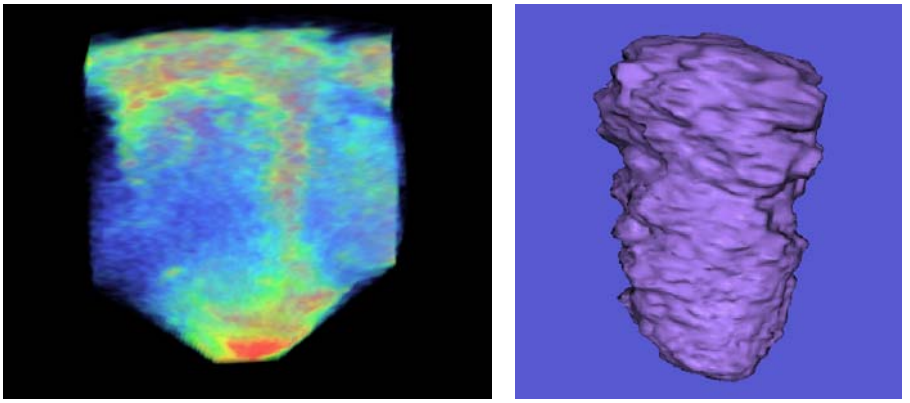


FIG. 3.5. *Volume rendering of original 3D data set (left) and segmentation of the ventricle (right).*

filling is then a consequence of the basic property of mean curvature flow, namely the minimization of surface area. Segmentation of an object of this size takes few seconds (on a 2.4GHz PC); one time step takes 0.15 sec, and we need few tens of time steps to find the result. A change in L_2-norm of numerical solutions in subsequent time steps is used to stop the segmentation process. We check whether

$$\sqrt{\sum_p m(p)\ (u_p^n - u_p^{n-1})^2} < \delta \tag{3.6}$$

with a prescribed threshold δ. In Figure 3.4(right) we visualize isosurface of the segmentation function close to its maximum. In all segmentation examples computed by our semi-implicit scheme we take $K = 1$, $h = \frac{1}{n_1}$, n_1 number of voxels in the horizontal direction, $\tau \approx 10 \times h^2$, and $\delta = 10^{-5}$. We start all computations with the initial function given as a peak centered in a "focus point" inside the segmented object. At a sphere with center s and radius R the initial function can be described by $u^0(x) = \frac{1}{|x-s|+v}$, where s corresponds to the focus point and $\frac{1}{v}$ gives a maximum of u^0. Outside that sphere we take values of u^0 equal to $\frac{1}{R+v}$. R usually corresponds to the halved inner diameter of the image domain. For small objects a smaller R can be used to speed up computations. In this example the maximum of the initial segmentation function (the focus point s) is taken in the image center.

The second example is given by 3D echocardiography of size $81 \times 87 \times 166$. As one can see from volume rendering visualization (Figure 3.5(left)), the 3D image is very noisy; however, the surface of the ventricle is observable. How noisy the image intensity is can be seen also from Figure 3.6, where one can see 2D image intensity and its graph in one 2D slice. Now we start the segmentation process with few "points of view" inside the object of interest, i.e., the initial segmentation function having several maxima, and let it evolve until the difference in L_2-norm of two subsequent time steps is less than the prescribed threshold. Then we look to a 2D slice with relatively good boundary edges (Figure 3.6), where we see accumulation of level sets along the inner boundary of the ventricle (Figure 3.6(left)). The largest gap in the histogram (Figure 3.7(left)) indicates the shock in the segmentation function, and so we choose one level inside the gap and plot it inside the slice (Figure 3.7(right)). We can check how this level set looks like in other noisy slices (Figure 3.8) and then make

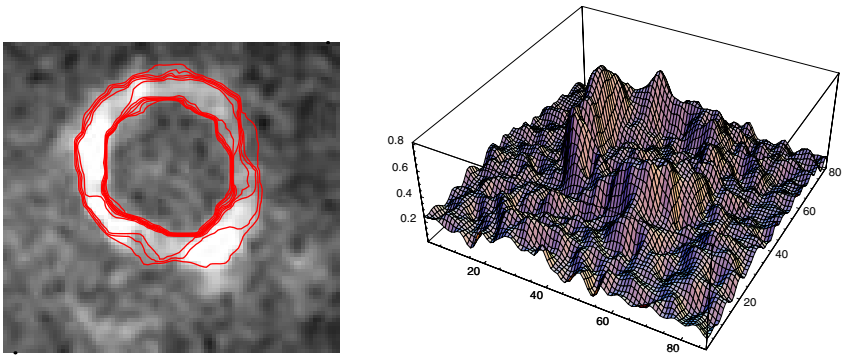


FIG. 3.6. *Plot of accumulated level sets in the slice* $k = 130$ *(left) and* $3D$ *graphical view of this slice (right).*

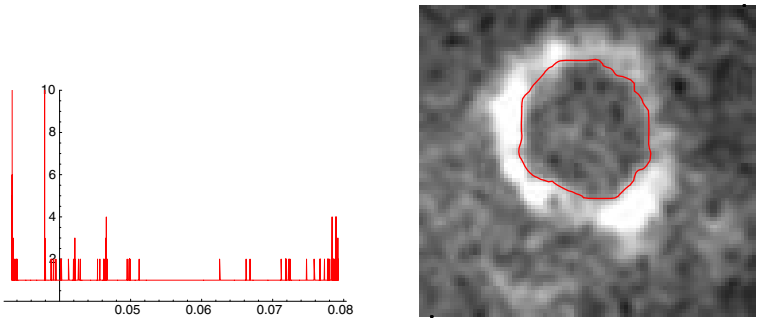


FIG. 3.7. *The histogram of the segmentation function in the slice* $k = 130$ *(middle); image intensity with level set* $u = 0.052$ *(right). Visualization of the* $3D$ *surface in Figure* 3.5 *is done with the same level set.*

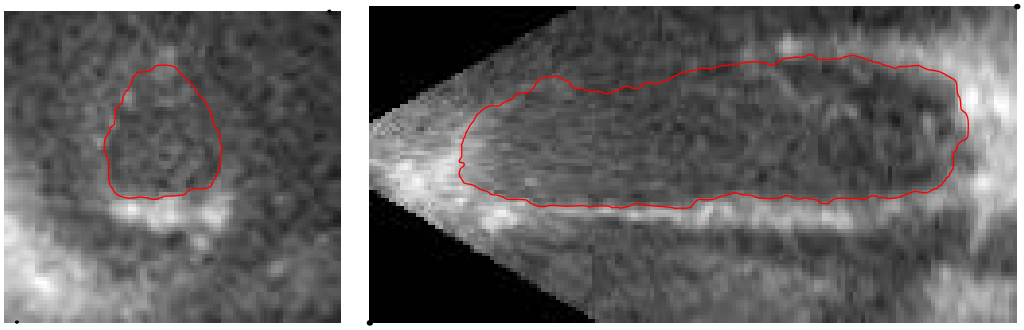


FIG. 3.8. *Plot of image intensity together with level line* 0.052 *in two other slices* $k = 100$ *(left) and* $j = 40$ *(right).*

a 3D isosurface visualization (Figure 3.5(right)) which gives realistic representation of the left ventricle.

The computation of one time step takes 3 seconds on a 2.4GHz one processor PC and takes 200 steps to finish segmentation, and so the overall computing time for this image size is in a range of few minutes. The MPI parallel implementation [3] of our 3D segmentation scheme has been built under the support of the Project HPC-EUROPA at CINECA SuperComputing Center, Bologna, and so the segmentation

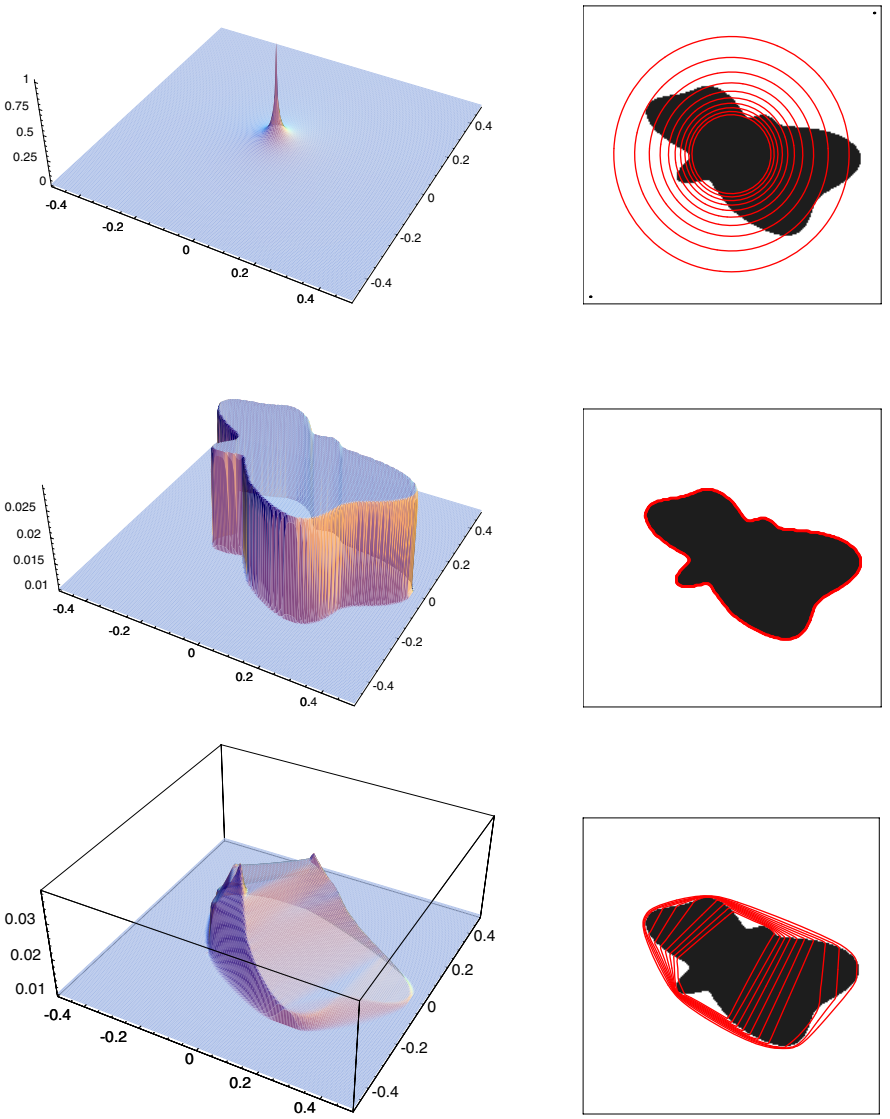


FIG. 3.9. *Successful subjective surface based segmentation* (1.4) *of a "batman" image, in the middle. Nonsuccessful segmentation only by slowed mean curvature motion* (1.5), *in the bottom.*

time is even speeding up depending on the number of available processors (e.g., in Linux cluster). The MPI parallelization of our algorithm is straightforward, because it uses classical structures of scientific computing methodology for solving partial differential equations and linear algebra problems (see, e.g., [3]).

3.3. Comparison with different models and methods. First, we show the difference between segmentation model (1.4) and (1.5) with g^0 instead of $g(|\nabla G_\sigma * u|)$; i.e., we document the role of convective term which appears in the model if g^0 is inside the divergence term. This comparison also shows that the method from [16] cannot be straightforwardly transferred from image smoothing to image segmentation problems.

Let us consider a simple nonconvex black object on a white background as plotted in Figure 3.9. If we start with the initial function plotted in the top left subfigure, isolines of which are plotted in the top right subfigure, and apply model (1.4), we end up with results as in the middle subfigures, while applying (1.5) we end up in the long run with the bottom subfigures. In the middle we see accumulation of all level lines on the object boundary due to the convective term, and any of these level lines can be used to represent segmentation of the object. In the case of (1.5), the motion of level lines is slowed down along the edges; however, level lines shrinking from outside, due to minimizing length (property of mean curvature motion), cannot go inside the nonconvex regions, and they accumulate on a convex envelope of the object.

Next we compare computational efficiency of our semi-implicit scheme with standard methods based on classical level set strategy and explicit time stepping. Such explicit schemes are usually based on formulation (1.4), where, moreover, all derivatives are expanded to get mean curvature and advection terms. Then, in three dimensions, (1.1) is written in the form

$$\begin{aligned} u_t = g^0((\varepsilon + u_{x_2}^2 + u_{x_3}^2)u_{x_1x_1} + (\varepsilon + u_{x_1}^2 + u_{x_3}^2)u_{x_2x_2} + (\varepsilon + u_{x_1}^2 + u_{x_2}^2)u_{x_3x_3} \\ -2u_{x_1}u_{x_2}u_{x_1x_2} - 2u_{x_1}u_{x_3}u_{x_1x_3} - 2u_{x_2}u_{x_3}u_{x_2x_3})/ \\ (\varepsilon + u_{x_1}^2 + u_{x_2}^2 + u_{x_3}^2) + g_{x_1}^0 u_{x_1} + g_{x_2}^0 u_{x_2} + g_{x_3}^0 u_{x_3}, \end{aligned} \tag{3.7}$$

where u_s means partial derivative of a function u with respect to a variable s. The explicit time stepping and central finite difference approximation is used in the mean curvature term, and upwind finite difference approximation is used in convective term [33]. For the computational comparison we use a representative 3D example where we segment a sphere with four holes (white object on black background in 80^3 voxels image); the 2D cut by equatorial plane is given in the upper left of Figure 3.10. In the upper right of Figure 3.10 we see the reconstructed sphere, and in the bottom part of Figure 3.10 we see the 2D cut of the shock profile of the segmentation function which is used to extract the upper sphere. This state of the segmentation function is achieved after 1450 time steps using discretization [33] of (3.7). Although CPU time for one explicit time update is small, due to stability reasons, both in curvature and convective parts, we have to choose a small time step, $\tau = 0.00002$, and need many of them to achieve a piecewise constant profile of the segmentation function. So the overall CPU time is 380 secs. In spite of that our semi-implicit discretization (2.15) fulfills unconditionally the discrete minimum-maximum principle (L_∞-stability); i.e., no spurious oscillations appear for any length of discrete time step. We get the same 3D segmentation result in 28 steps with $\tau = 0.002$. Although one step of the semi-implicit scheme is more complex, we have to solve the linear system, the overall CPU time is 45 secs, and there is a benefit of almost 10 times speed-up. This speed-up ratio is quite a common observation in many 3D segmentation comparisons of semi-implicit and explicit schemes.

At the end we illustrate removing a grid orientation effect which can be observed in numerical results of previously studied algorithms [35, 16]. There either "left oriented" or "right oriented" triangulations (see Figure 3.11) have been used for solving the level-set-like problems, and, especially on coarse grids, the influence of grid orientation can be observable. To show this phenomenon in a transparent way we present a 2D experiment where the initial unit circle is moving by mean curvature and expanding with unit speed in the outer normal direction at the same time. In such a case, $Fm(p)$, $F = -1$ is added to the right-hand side of the fully discrete scheme (2.15). Since there is a balance between curvature and driving force the unit circle is the

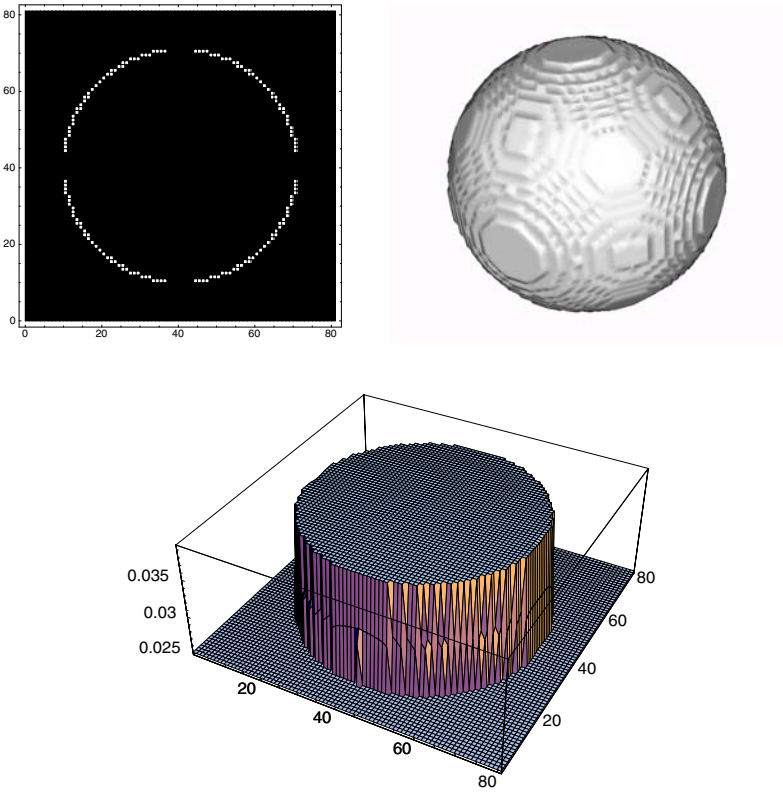


FIG. 3.10. *Subjective surface based segmentation of a 3D sphere with four holes.*

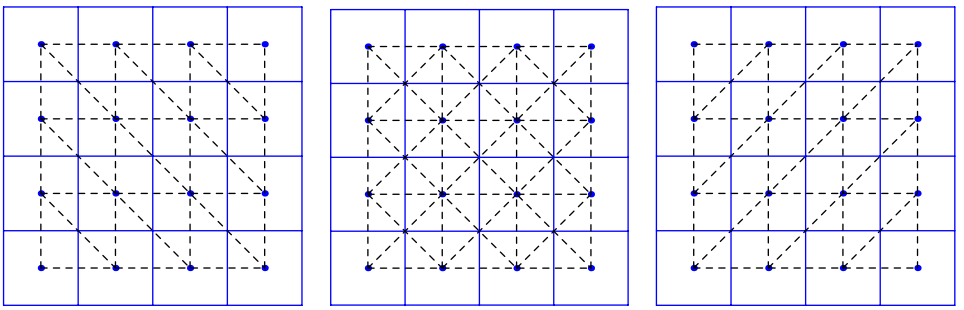


FIG. 3.11. *By dashed lines we plot the "left oriented" triangulation (left), "right oriented" triangulation (right), and "symmetric" triangulation corresponding to our method in two dimensions. The image pixels (solid lines) in all cases correspond to the covolume mesh.*

exact solution, and so it should remain stable also computationally. Using the "right oriented" triangulation with 20×20 covolumes and the method from [16], at time $T = 1$, we get the result plotted in Figure 3.12(left). We see exact solution plotted by the dashed line and its slight distortion visualizing the numerical solution. On the other hand, using the 2D analogy of the method presented in this paper, i.e., "symmetric" triangulation plotted in the middle part of Figure 3.11, we get the numerical result presented in Figure 3.12(right), where the circular shape is not distorted. Our 3D tetrahedral mesh and covolume grid construction transfer such "symmetry" of the

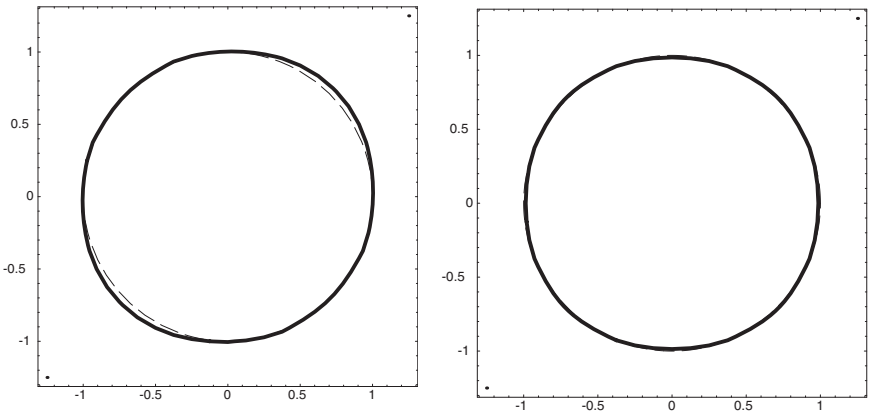


FIG. 3.12. *By the dashed line we plot the exact solution (stable unit circle). By the thick solid line we plot the level line obtained from numerical solution using the "right oriented" triangulation (left) and the "symmetric" triangulation corresponding to our method in two dimensions. In the latter case the solutions coincide in this visualization scale.*

method into the 3D case, and in this sense the previous covolume techniques are improved.

Acknowledgement. We thank Roberto Gori from CINECA for help with 3D visualizations.

REFERENCES

[1] L. ALVAREZ, F. GUICHARD, P. L. LIONS, AND J. M. MOREL, *Axioms and fundamental equations of image processing*, Arch. Rational Mech. Anal., 123 (1993), pp. 199–257.
[2] L. ALVAREZ, P.-L. LIONS, AND J.-M. MOREL, *Image selective smoothing and edge detection by nonlinear diffusion*. II, SIAM J. Numer. Anal., 29 (1992), pp. 845–866.
[3] Y. AOYAMA AND J. NAKANO, *RS/*6000 *SP: Practical MPI Programming*, IBM, http://www.redbooks.ibm.com (1999).
[4] E. BÄNSCH AND K. MIKULA, *A coarsening finite element strategy in image selective smoothing*, Comput. Vis. Sci. 1 (1997), pp. 53–61.
[5] V. CASELLES, F. CATTÉ, T. COLL, AND F. DIBOS, *A geometric model for active contours in image processing*, Numer. Math., 66 (1993), pp. 1–31.
[6] V. CASELLES, R. KIMMEL, AND G. SAPIRO, *Geodesic active contours*, Int. J. Comput. Vis., 22 (1997), pp. 61–79.
[7] V. CASELLES, R. KIMMEL, G. SAPIRO, AND C. SBERT, *Minimal surfaces: A geometric three dimensional segmentation approach*, Numer. Math., 77 (1997), pp. 423–451.
[8] F. CATTÉ, P.-L. LIONS, J.-M. MOREL, AND T. COLL, *Image selective smoothing and edge detection by nonlinear diffusion*, SIAM J. Numer. Anal., 29 (1992), pp. 182–193.
[9] S. L. CHAN AND E. O. PURISIMA, *A new tetrahedral tesselation scheme for isosurface generation*, Computers and Graphics, 22 (1998), pp. 83–90.
[10] Y.-G. CHEN, Y. GIGA, AND S. GOTO, *Uniqueness and existence of viscosity solutions of generalized mean curvature flow equation*, J. Differential Geom., 33 (1991), pp. 749–786.
[11] G. CITTI AND M. MANFREDINI, *Long time behavior of Riemannian mean curvature flow of graphs*, J. Math. Anal. Appl., 273 (2002), pp. 353–369.
[12] M. G. CRANDALL, H. ISHII, AND P. L. LIONS, *User's guide to viscosity solutions of second order partial differential equations*, Bull. Amer. Math. Soc. (N.S.), 27 (1992), pp. 1–67.
[13] K. DECKELNICK AND G. DZIUK, *Error estimates for a semi implicit fully discrete finite element scheme for the mean curvature flow of graphs*, Interfaces Free Bound., 2 (2000), pp. 341–359.
[14] K. DECKELNICK AND G. DZIUK, *Numerical approximation of mean curvature flow of graphs and level sets*, in Mathematical Aspects of Evolving Interfaces, L. Ambrosio, K. Deckelnick, G. Dziuk, M. Mimura, V. A. Solonnikov, and H. M. Soner, eds., Springer, Berlin, Heidelberg, New York, 2003, pp. 53–87.

[15] L. C. Evans and J. Spruck, *Motion of level sets by mean curvature* I, J. Differential Geom., 33 (1991), pp. 635–681.
[16] A. Handlovičová, K. Mikula, and F. Sgallari, *Semi–implicit complementary volume scheme for solving level set like equations in image processing and curve evolution*, Numer. Math., 93 (2003), pp. 675–695.
[17] J. Kačur and K. Mikula, *Solution of nonlinear diffusion appearing in image smoothing and edge detection*, Appl. Numer. Math., 17 (1995), pp. 47–59.
[18] G. Kanizsa, *Organization in Vision*, Praeger, New York, 1979.
[19] M. Kass, A. Witkin, D. and Terzopulos, *Snakes: Active contour models*, Int. J. Comput. Vis., 1 (1987), pp. 321–331.
[20] S. Kichenassamy, A. Kumar, P. Olver, A. Tannenbaum, and A. Yezzi, *Conformal curvature flows: From phase transitions to active vision*, Arch. Rational Mech. Anal., 134 (1996), pp. 275–301.
[21] R. LeVeque, *Finite Volume Methods for Hyperbolic Problems*, Cambridge Texts in Applied Mathematics, Cambridge University Press, Cambridge, UK, 2002.
[22] R. Malladi, J. A. Sethian, and B. Vemuri, *Shape modeling with front propagation: A level set approach*, IEEE Trans. Pattern Anal. Machine Intell., 17 (1995), pp. 158–174.
[23] K. Mikula and N. Ramarosy, *Semi–implicit finite volume scheme for solving nonlinear diffusion equations in image processing*, Numer. Math., 89 (2001), pp. 561–590.
[24] K. Mikula and D. Ševčovič, *Computational and qualitative aspects of evolution of curves driven by curvature and external force*. Comput. Vis. Sci., 6 (2004), pp. 211–225.
[25] D. Mumford and J. Shah, *Optimal approximation by piecewise smooth functions and associated variational problems*, Comm. Pure Appl. Math., 42 (1989), pp. 577–685.
[26] A. M. Oberman, *A convergent monotone difference scheme for motion of level sets by mean curvature*, Numer. Math., 99 (2004), pp. 365–379.
[27] S. Osher and R. Fedkiw, *Level Set Methods and Dynamic Implicit Surfaces*, Springer-Verlag, New York, 2003.
[28] S. Osher and J. A. Sethian, *Front propagating with curvature dependent speed: Algorithms based on the Hamilton–Jacobi formulation*, J. Comput. Phys., 79 (1988), pp. 12–49.
[29] P. Perona and J. Malik, *Scale-space and edge detection using anisotropic diffusion*, IEEE Trans. Pattern Anal. Machine Intell., 12 (1990), pp. 629–639.
[30] Y. Saad, *Iterative Methods for Sparse Linear Systems*, PWS, Boston, MA, 1996; reprinted, SIAM, Philadelphia, 2003.
[31] A. Sarti, R. Malladi, and J. A. Sethian, *Subjective surfaces: A method for completing missing boundaries*, Proc. Natl. Acad. Sci. USA, 97 (2000), pp. 6258–6263.
[32] A. Sarti and G. Citti, *Subjective surfaces and Riemannian mean curvature flow graphs*, Acta Math. Univ. Comenian., (N.S.), 70 (2000), pp. 85–103.
[33] A. Sarti, R. Malladi, and J. A. Sethian, *Subjective surfaces: A geometric model for boundary completion*, Int. J. Comput. Vis., 46 (2002), pp. 201–221.
[34] J. A. Sethian, *Level Set Methods and Fast Marching Methods, Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science*, Cambridge University Press, Cambridge, UK, 1999.
[35] N. J. Walkington, *Algorithms for computing motion by mean curvature*, SIAM J. Numer. Anal., 33 (1996), pp. 2215–2238.

SIAM J. SCI. COMPUT.
Vol. 28, No. 6, pp. 2266–2293
© 2006 Society for Industrial and Applied Mathematics

A PARALLEL MULTISTAGE ILU FACTORIZATION BASED ON A HIERARCHICAL GRAPH DECOMPOSITION*

PASCAL HÉNON† AND YOUSEF SAAD‡

Abstract. PHIDAL (parallel hierarchical interface decomposition algorithm) is a parallel incomplete factorization method which exploits a hierarchical interface decomposition of the adjacency graph of the coefficient matrix. The idea of the decomposition is similar to that of the well-known wirebasket techniques used in domain decomposition. However, the method is devised for general, irregularly structured, sparse linear systems. This paper describes a few algorithms for obtaining good quality hierarchical graph decompositions and discusses the parallel implementation of the factorization procedure. Numerical experiments are reported to illustrate the scalability of the algorithm and its effectiveness as a general purpose parallel linear system solver.

Key words. parallel incomplete LU factorization, ILU, sparse Gaussian elimination, wirebasket decomposition, interface decomposition, preconditioning, ILU with threshold, iterative methods, sparse linear systems

AMS subject classifications. 65F10, 65N06

DOI. 10.1137/040608258

1. Introduction.

A number of incomplete LU (ILU) factorization techniques have been developed in recent years which attempt to combine some of the general-purpose features of standard ILU preconditioners with the good scalability attributes of multi-level methods. Examples include the ILU with multielimination (ILUM) [13], the multilevel ILU (MLILU) [3] and matrix renumbering ILU (MRILU) algorithms [4], and the algebraic recursive multilevel solver (ARMS) [16] and its parallel version pARMS [9]. An ingredient that is common to all these methods is to reorder the linear system to enable a parallel (incomplete) elimination of groups of unknowns. Solution algorithms which implement similar strategies within domain decomposition have also been developed and can be found, for example, in the pARMS [9] package. This paper presents a new technique in this category. The main attraction of the new method is that it exploits a static "hierarchical" decomposition of the graph which yields natural parallelism in the factorization process. This is in contrast with the methodology used in pARMS in which the independent sets and levels are defined dynamically. The underlying ordering on which the method is based is reminiscent of the "wirebasket" techniques of domain decomposition methods [17, 18], in which cross points in a domain partitioning play an important role. Cross points have played a special role in many domain decomposition techniques; see [5], for example. One can also view the methods from the angle of an ILU factorization combined with a form of nested dissection ordering in which cross points in the separators play a special role. In this regard, the method bears some resemblance to the FETI method [6]. The ideas discussed in this paper also bear resemblance to those sketched by Magolu monga Made and van der Vorst [12, 10, 11] for matrices arising from finite difference

*Received by the editors May 14, 2004; accepted for publication (in revised form) June 14, 2006; published electronically December 18, 2006. This work was supported in part by the NSF under grants NSF/ACI-0305120 and NSF/INT-0003274, and by the Minnesota Supercomputer Institute.
http://www.siam.org/journals/sisc/28-6/60825.html

†INRIA Futurs, Scalapplix project–LaBRI, 351 cours de la Libération, 33405 Talence, France (henon@labri.fr).

‡University of Minnesota, 200 Union Street S.E., Minneapolis, MN 55455 (saad@cs.umn.edu).

discretizations of 2-dimensional partial differential equations.

The algorithm described in this paper is based on defining a "hierarchical interface structure." The hierarchical decomposition starts with a partitioning of the graph, with one layer of overlap. Then, "stages" or "levels" are defined from this partitioning, with each level consisting of a set of groups or connectors (small connected subgraphs). The levels obey an important property which is that each connector of a given level is a separator for connectors of a lower level. Thus, for a simple 5-point graph the connectors at level 1 are the sets of interior nodes for each subdomain obtained from the partitioning. Level 2 includes the interfaces between domains at the exclusion of cross points, and level 3 consists of all the cross points. The vertices are globally ordered by level, in increasing order of level number. The incomplete factorization process proceeds by level from lowest to highest. Because of the separation property of the connectors at different levels, it turns out that this process can be made highly parallel by tuning the dropping strategy to the levels. More precisely, the dropping strategies will attempt to maintain the independent set structure extracted from the interface decomposition.

As was already mentioned, a major difference between this method and existing methods of this type, such as pARMS, is that the preordering used for parallelism is performed at the outset instead of dynamically. This has both advantages and disadvantages. Because the ordering is static, the factorization is rather fast. On the other hand, the ordering may face difficulties in the case when, for example, zero or small pivots are encountered. In pARMS, some form of diagonal pivoting is performed to bypass this difficulty.

The main ingredient of the ILU factorization is the hierarchical interface decomposition of the graph. It is fairly easy to find definitions which will achieve the desired goals of yielding a hierarchy of subgraphs which separate each other. For example, just assigning to each node a label which is defined as the number of subdomains to which the node belongs yields, under certain conditions, a decomposition into levels of independent sets, given by the labels. This is the case for simple graphs such as the one of a 5-point finite difference matrix. Thus, in the illustration of Figure 2.1, this strategy would yield 3 different labels. Nodes labeled "1" are interior nodes, nodes labeled "2" are interface nodes (excluding cross points), and nodes labeled "4" are the cross points. Cross points are separators for interfaces which are separators for the sets of interior nodes for each domain. This is just the desired decomposition.

This simple strategy does not always work. It will fail, for example, if a 9-point discretization is used. In such situations, there are usually ways to modify the labels to yield the desired hierarchical decomposition. However, decompositions obtained in this way are often suboptimal for "rich" graphs, in that they may yield a large number of levels. This is an important issue to which a substantial part of the paper is devoted.

The next section discusses the hierarchical decomposition of a graph. Section 3 presents the details of the parallel elimination process. Section 6 reports on some experiments. The paper ends with some concluding remarks and tentative ideas for future work.

2. Hierarchical interface decomposition. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be the adjacency graph of a sparse matrix A which has a symmetric pattern so that $\mathcal{G}$ is undirected. Assume that $\mathcal{G}$ is partitioned into p subgraphs. An edge-based partitioning is used, in the sense that edges are primarily assigned to subdomains, and therefore vertices at the interfaces overlap two or more domains. This is the common situation which arises,

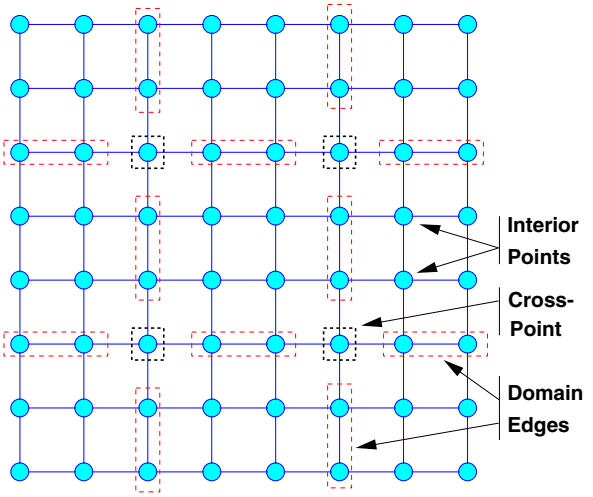


FIG. 2.1. *Partition of an* 8×8 5*-point mesh into* 9 *subdomains and the corresponding hierarchical interface structure.*

for example, when elements in finite element methods are assigned to processors.

We begin by illustrating the hierarchical decomposition process on a matrix A which arises from a simple 5-point finite difference discretization of a Laplacean in 2-dimensional space. The vertex set $\mathcal{V}$ consists of three types of vertices representing the common "interior vertices," "domain-edges," and "cross points" of the partition; see Figure 2.1 for an illustration. These three types of points are grouped in sets which will be referred to as "connectors." Thus, the interior points of a given domain form one connector, and the domain-edges, i.e., the set of interface points linking two given subdomains (excluding cross points), also form a connector. In this case, levels of connectors are also naturally defined. Level-1 connectors are the sets of interior points, level-2 connectors are the domain-edges, and level-3 connectors are the cross points.

This decomposition of the vertices has the interesting property that different connectors at the same level are not coupled. Another interesting property is that the higher-level connectors separate the lower-level connectors: Interior domains (level-1 connectors) are separated by domain-edges and cross points (level-2 and level-3 connectors) and domain-edges are themselves separated by cross points. These properties can be exploited in a parallel context by taking advantage of the block structure of the matrix A obtained by reordering the unknowns according to this decomposition. If the unknowns are reordered according to their level numbers, from the lowest to highest, the block structure of the reordered matrix would be as shown in Figure 2.2. In this figure we can see that for a 5-point mesh, this kind of ordering will naturally yield a diagonal block structure for each level of connectors. Notice that the block structure leads to natural parallelism if an ILU decomposition is performed, provided that fill-in is avoided where it would hamper parallel processing of the factorization. Our goal is to generalize this ordering to irregularly structured graphs in order to obtain a similar matrix structure which can be exploited in a parallel solver. It is worth mentioning that the regularity of the block structure in Figure 2.2 (all connectors of a level have the same size) is related to the regularity of the mesh and the particular partitioning used. Similarly, the tridiagonal structure of the diagonal blocks is also

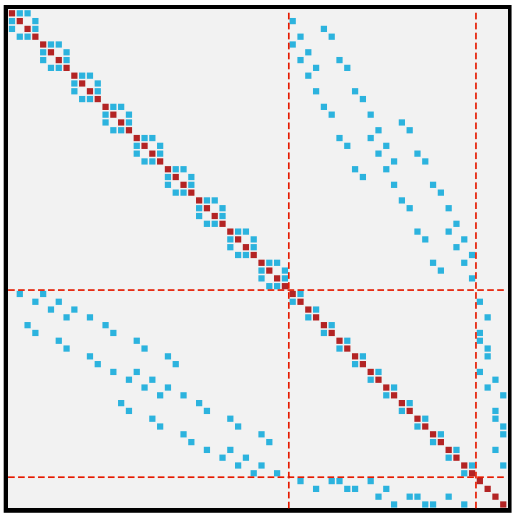


FIG. 2.2. *Matrix associated with an* 8×8 *5-point mesh reordered according to the hierarchical interface decomposition.*

due to the nature of the problem. It is not the goal of the ordering to achieve such a regular structure. Based on the hierarchical graph decomposition, the block structure of the matrix A is obtained by using an ordering that

- labels unknowns in the same connector by consecutive numbers,
- labels unknowns in a connector of level l with smaller numbers than those in connectors of level $m > l$.

The remainder of this section will discuss extensions of the "wirebasket" structure (which was just seen for the 5-point mesh example) to irregularly structured graphs. We will refer to this structure as a "hierarchical interface decomposition" of the graph. Given the general definition, we will then propose in the next section some heuristics to compute the interface decomposition.

We begin with a formal definition of a hierarchical decomposition of a graph. We will call a *connector* a connected subgraph of $\mathcal{G}$. A *connector decomposition*, or *interface decomposition*, is a grouping of connectors into *levels*. So, a *level* is simply a set of connectors. Levels are labeled from 1 to p. Two connectors are *adjacent* (or "coupled") if there is at least one edge linking a vertex from one connector to a vertex of the other. A *hierarchical interface decomposition* is defined as follows.

DEFINITION 2.1. *A set of levels labeled* $1, \ldots, p$ *is a hierarchical decomposition of* $\mathcal{G}$ *if it satisfies the following conditions:*

(1) *The connectors are disjoint and together form a partition of the graph.*
(2) *Connectors of the same level are not adjacent.*
(3) *Connectors of a given level* $l > 1$ *are separators for connectors at level* $l - 1$. *This means that the removal of a connector of level* l *will disconnect (in the original graph) at least two connectors of level* $l - 1$.

Condition (2) by itself implies that two connectors C_1, C_2 which are not adjacent in the original graph and which belong to the same level l must be separated by some connector C_3 from a different level. Condition (3) simply states that the level number

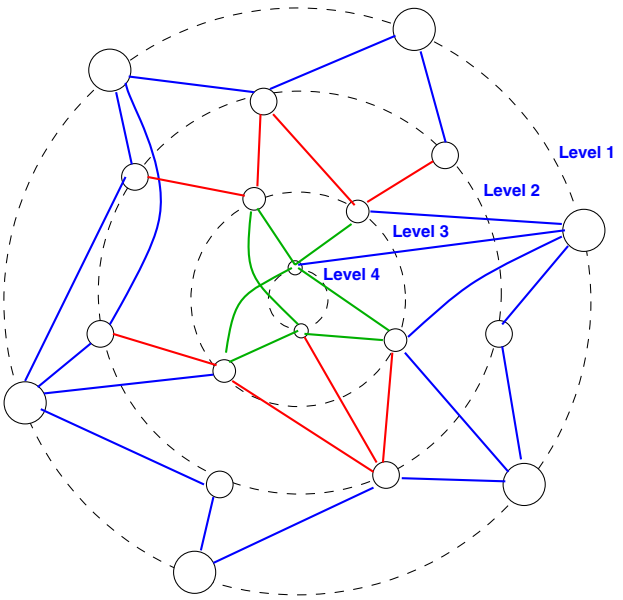


FIG. 2.3. *An example of a hierarchical decomposition of a graph. The circles are connectors, and the links show the adjacencies between them. The connectors are grouped into* 4 *levels indicated by the dashed concentric circles.*

of C_3 must be $l+1$. An illustration of the definition is given in Figure 2.3.

The issue now is to obtain such hierarchical decompositions. We may attempt to develop algorithms specifically for this task. In other words, we could seek algorithms that would take a graph and produce a partition into connectors satisfying the above conditions. In addition, the algorithm would also attempt to yield a good load balancing. This is not attempted in this paper. Instead, we will use as a starting point a partitioned graph, where the partitioning results from a standard graph partitioner such as METIS [8].

A few additional clarifications regarding the above definition may be helpful. The term "interface" is used only because of the nature of the algorithms that will produce the decomposition. The term "hierarchical" refers to the levelization of the connector sets with the property that the connectors of a given level l separate those connectors of lower levels $m < l$. For the 5-point example a good levelization was achieved by defining the level number of a connector from the number of adjacent domains. For general irregular domains and partitions, the situation is more complex, and this process, when it works, will yield a poor levelization in the sense that the resulting levelization may produce levels which contain few connectors, resulting in a high number of levels. Figure 2.4 shows an irregular domain partition (the figure shows only the connectors, not the graph vertices and edges). In this case, there are 5 levels. Clearly, a better level-structure of the connectors is achieved by grouping all the connectors represented as dots together, resulting in only 3 sets of independent connectors. In subsection 2.3, we discuss an algorithm that is able to extract a good level partitioning.

Before continuing, it may be insightful to consider alternatives for obtaining hierarchical interface decompositions. A first viewpoint can be realized by looking at the matrix in Figure 2.2, which is very similar to one that would be obtained from

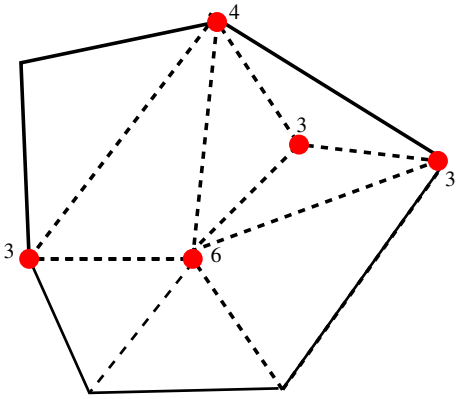


FIG. 2.4. *Connector levels resulting from an irregular domain partition. The solid lines represent the domain boundaries. The dashed lines represent level-2 connectors. The big dots are the level-i connectors with $i > 2$. The level labels for these connectors are shown next to them.*

multicoloring. In fact it can be viewed as a "block" multicoloring of the graph, in the same way that the orderings considered in ARMS [16] can be viewed as block versions of independent set orderings. While this is correct, the main issue is to define connectors first from the partitioned graph. Once these are obtained a multicoloring process can be called to define the levels. The point is that the difficulty resides in getting good connectors, not in getting the levels.

2.1. Algorithms for building hierarchical decompositions of a graph. We begin with some definitions and notation that will be used in the remainder of the paper. We are given an initial partitioning of the graph with overlap. The algorithm to define the hierarchical decomposition for general graphs will exploit the freedom to reassign the interface nodes to the neighboring subdomains so that certain properties of the modified partitioning are satisfied. In order to do this in a systematic manner, we associate with each vertex u a unique key $Key(u)$,

$$Key(u) = \{i_1, \ldots, i_m\},$$

which contains a list of all subdomains i_k to which u belongs in the graph partitioning. It is to be understood here that the assignment of u to subdomains will be altered dynamically by the procedure which defines the decomposition and thus the key will be changing in the course of the procedure. We also denote by $C_{i_1,\ldots,i_m}$ the set of vertices which are overlapped by subdomains $i_1, i_2, \ldots, i_m$. In particular note that

$$Key(u) = \{i_1, \ldots, i_m\} \Leftrightarrow u \in C_{i_1,\ldots,i_m}.$$

The subsets $C_{i_1,\ldots,i_m}$ are termed connectors. The notation $Key(C) = \{i_1, \ldots, i_m\}$ will also be used for the connector $C = C_{i_1,\ldots,i_m}$. We will also denote by $deg(C)$ the cardinality of the key of C, i.e.,

$$deg(C) = |Key(C)|,$$

which is the number of subdomains that include C. Note that C is an intersection of subdomains, so there is no ambiguity in this definition. Similarly, we can also define the *domain degree* of a node by $deg(u) = |Key(u)|$, which is again the number of

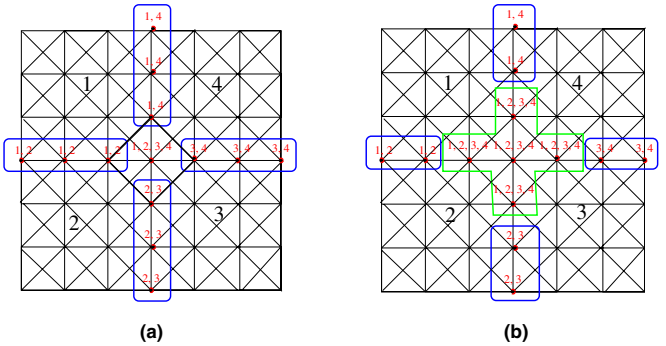


FIG. 2.5. (a) *Partition of a* 9*-point grid into* 4 *subdomains. Connectors induced by the domain partitioning are not independent.* (b) *One solution for recovering independent sets of connectors.*

domains with which u is associated. By abuse of language we will sometimes call this the degree of u, which is to be distinguished from the actual degree of u in the graph.

The purpose of the keys is to help build good connectors. Given an arbitrary set of domain labels $\{i_1, i_2, \ldots, i_k\}$, one can formally define any connector $C_{i_1,i_2,\ldots,i_k}$ as the set of all nodes belonging to subdomains $i_1, \ldots, i_k$ (most connectors defined this way will be empty). For any particular partitioning, $Key(u) = \{i_1, \ldots, i_k\}$ if and only if u belongs to $C_{i_1,i_2,\ldots,i_k}$. The connectors depend on the particular partitioning. The algorithms to be described in this section will slightly alter the partitioning, i.e., the keys, or equivalently, the connectors, to achieve certain properties. In the proposed algorithms, these slight modifications will affect only the overlapping regions of the subdomains.

In the example of the grid, the partition of the interface into domain-edges and cross points defines a partition of connectors: Interior domains are connectors of degree 1, domain-edges are connectors of degree 2, and cross points are connectors of degree 4. If C is a domain-edge between domains 1 and 2, then $Key(C) = \{1, 2\}$. As was mentioned above, in more complex cases, it will be necessary to redefine keys in order for the connectors to be independent. Though section 2.2 is devoted to this issue we provide an example in Figure 2.5 to illustrate the idea.

Figure 2.5(a) is an original partitioning of a 9-point grid into 4 subdomains. The middle lines overlap, and the center point is assigned to all 4 subdomains. The initial keys associated with this mapping are shown above each node. This example shows that level-2 connectors defined directly from the domain degree $|Key(u)|$ may be adjacent. There are several possible remedies to change the keys so that the new connectors become independent. Figure 2.5(b) shows one possible and natural solution which consists of expanding $Key(u)$ so that it contains the list of all the domains of each member of $adj(u)$, i.e., the set of all subdomains of the nearest neighbors of u. As will be seen in section 2.2, this is not necessarily the best solution.

The original decomposition of a graph sometimes yields a hierarchical decomposition such that the connectors of the same level are uncoupled: This was the case for the 5-point mesh example seen earlier. By this we mean that in such situations, sets of connectors of the same degree will constitute a level for the hierarchical decomposition. Unfortunately, this situation is special. For an arbitrary irregular graph, the connectors obtained in this manner will not in general form a hierarchical interface decomposition. Indeed, connectors of the same degree can be adjacent. In

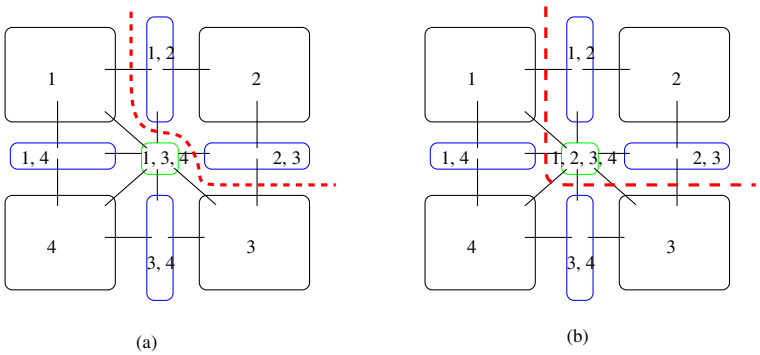


FIG. 2.6. (a) *Nonconsistent interface decomposition.* (b) *Consistent interface decomposition.*

the remainder of this section, we define first what properties we want to obtain from the interface decomposition and then we discuss algorithms to modify any interface decomposition in order to achieve these properties.

We need a property of a connector decomposition which will be called *consistency.* It is given next.

DEFINITION 2.2. *An interface decomposition is said to be consistent if any pair of connectors* C_1, C_2 *verifies*

- C_1 *is adjacent to* $C_2 => Key(C_1) \subsetneq Key(C_2)$ *or* $Key(C_2) \subsetneq Key(C_1)$.

Another way to state the definition is that when two connectors are adjacent in a consistent interface decomposition, then one connector key must be a strict superset of the other. In particular, two adjacent connectors must have different degrees, and the key of the one with higher degree contains (strictly) the key of the other. Figure 2.6 shows an example of a consistent and a nonconsistent interface decomposition. In case (a) the connector $C_{1,2}$ is adjacent to $C_{1,2,3}$ and yet neither of the keys $\{1, 2\}$ and $\{1, 3, 4\}$ is included in the other. Similarly, $C_{2,3}$ is adjacent to $C_{1,3,4}$, yet $\{2, 3\} \not\subseteq \{1, 3, 4\}$ and $\{1, 3, 4\} \not\subseteq \{2, 3\}$. Modifying the key of a connector is equivalent to modifying the overlap between subdomains. Thus, the dashed line in Figure 2.6 shows the boundary of subdomain 2 (overlap included). Note that subdomain k is the union of all connectors whose key contains the index k. In case (a) the connector $C_{1,3,4}$ is not a part of subdomain 2, whereas in case (b) the modified connector $C_{1,2,3,4}$ becomes a part of subdomain (2) as shown. The modification consisted of adding label (2) to the key of the connector $C_{1,3,4}$.

An interface decomposition which satisfies the consistency requirement obeys two important properties which can be helpful in a parallel block ILU factorization.

LEMMA 2.1. *In a consistent interface decomposition, two connectors of the same degree cannot be adjacent.*

The proof of Lemma 2.1 was essentially stated above, namely, two adjacent connectors have different degrees.

This property allows an independent elimination of all connectors of the same degree in a block ILU(0) factorization. Assume that a Gaussian elimination process is undertaken whereby the unknowns are ordered by blocks corresponding to the connector partition of the graph. The connectors, i.e., the blocks of unknowns, are ordered by increasing degree. Recall that in an ILU factorization based on the level-of-fill [14] there exists a fill-in entry of level k if and only if there exists a fill-path of length $k + 1$ in the adjacency graph of the matrix. A fill-path is a path between

The cases a) and b) show all possible fill-paths of length ≤ 3 in the connector quotient graph. In these cases, the connectors in the fill path belong to at least a domain listed in the "minimum" connector key (the intersection of all keys of the connectors) of the path.
Thus, in case a) we have $Key(C_2) \subsetneq Key(C_1)$ and $Key(C_2) \subsetneq Key(C_3)$. In this situation $Key(C_1) \bigcap Key(C_2) \bigcap Key(C_3) = Key(C_2)$. In the case b), we also have $Key(C_1) \bigcap Key(C_2) \bigcap Key(C_3) \bigcap Key(C_4) = Key(C_3)$. This is true for the scenarios at the left and right of the case figure.
Case c) illustrates the fact that for a longer path we cannot say anything since $\bigcap_{i=1}^{5} Key(C_i) = Key(C_2) \bigcap Key(C_4)$, which may be empty. This is a situation where there is no common intersection (minimum) to all connectors: there are two local minima.

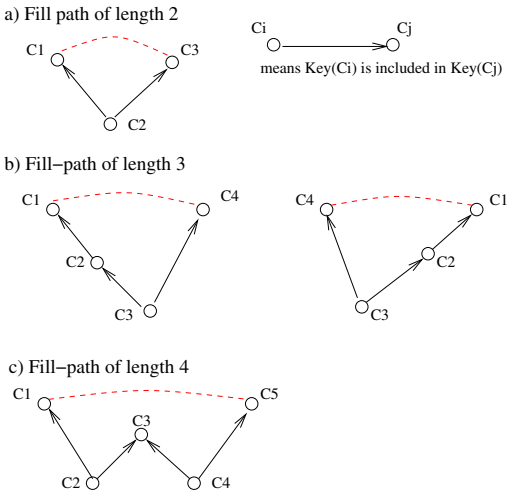


FIG. 2.7. *Proof of Lemma* 2.2.

two vertices v_i and v_j such that all intermediate vertices in the path have a lower label than those of v_i and v_j. Consider now a *block* ILU(k) factorization in which the blocks correspond to a partitioning of the unknown according to the interface decomposition. In the block factorization, fill-in is allowed in a given block by just using the level-of-fill pattern obtained from the quotient graph corresponding to the connectors. In this situation, blocks that are initially zero in the matrix A, and where fill-in will occur in the block ILU(k) factorization, correspond to fill-paths of length $\leq k+1$ in this quotient graph, which we call the connector graph. Note that in this graph, an edge between two connectors represents the adjacency between these connectors. A fill-path of length k is then a path of connectors $C_1, \ldots, C_k$ in the connector graph $\mathcal{C}$ such that

$$\forall i,\ 1 < i < k,\ deg(C_i) < deg(C_1)\ \ and\ \ deg(C_i) < deg(C_k)$$

since all connectors are numbered by ascending degree.

LEMMA 2.2. *Consider the connector graph $\mathcal{C}$ associated with a consistent interface decomposition. Then for a fill-path of length ≤ 3 there is always at least one domain number that is listed in all the keys of the connectors of the path.*

The proof is given in Figure 2.7 with some illustrative examples. A corollary of Lemma 2.2 is that the fill-in induced by a block ILU(k) factorization with $k \leq 2$ of a matrix reordered and partitioned using a consistent interface decomposition is always contained in the local matrix of a subdomain. Section 3 will reconsider this in detail.

2.2. Algorithms for obtaining consistent decompositions. In the remainder of this section, we will discuss first an algorithm for obtaining a consistent decomposition of the interface into connectors. Then an algorithm to obtain levels of independent connectors will be discussed.

Figure 2.5(a) shows that level-2 connectors defined directly from the domain degree $|Key(u)|$ may be adjacent. One possible remedy in this case is to define $Key(u)$

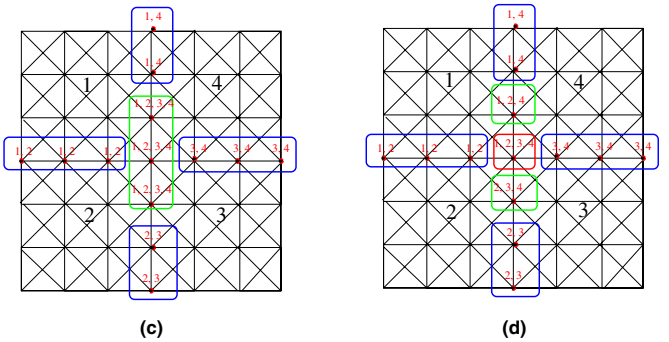


FIG. 2.8. *Two additional possible solutions to get independent connectors in each level by modifying some vertex keys.*

differently. For example, we could define $Key(u)$ as containing the list of all the domains of each member of $adj(u)$, i.e., the set of all subdomains of the nearest neighbors of u. This fixes the problem but leads to interfaces that are unnecessarily "thick" in some cases. Thus, in the example of the 9-point matrix of Figure 2.5(a), the domain interfaces will be 3 lines wide instead of just 1 line wide. In this case, there is a remedy, which is to define $Key(u)$ in this manner only for noninterior nodes. This will give the solution shown in Figure 2.5(b). A more general solution to get a consistent connector set is to start with a connector decomposition based on the initial overlap given by the partition and then refine it by *modifying, in a certain way, the keys* $Key(u)$ *of some overlapped nodes at the interface of the subdomains.* Thus, the solution in Figure 2.5(b) consists of modifying 4 keys in the manner just described. Figure 2.8 shows two additional solutions to get independent connectors in a level, based on modifying keys.

Several possibilities arise in redefining keys to comply with the requirements of a hierarchical interface decomposition, and some solutions appear to be better than others. In Figure 2.8 the solution (c) produces 3 levels (including the first level of interior node) in the overlap of the interface, whereas solution (d) produces 4 levels. In Figure 2.5(b) 4 keys are modified, resulting in 3 levels (degrees 1, 2, and 4), while in Figure 2.8(c) 2 keys are modified, resulting again in 3 levels, and in Figure 2.8(d) 2 keys are modified, resulting in 4 levels. Note that case (d) is not a consistent connector decomposition (Key $\{3,4\}$ is not included in key $\{1,2,4\}$, but the corresponding connectors are adjacent). This example illustrates the fact that some postprocessings of the keys are more desirable than others.

In the previous examples, we discussed two important criteria to use in an irregular graph to find a good interface partitioning. The first is to limit the number of vertices whose key must be expanded to obtain a consistent connector set. The second is to obtain as small a number of levels as possible. Thus, *the ideal interface decomposition should yield a small number of levels by modifying as few keys as possible.* In the general case, the problem of finding the best hierarchical decomposition of an interface is NP-complete because the problem of finding the minimum vertex set to be modified in order to get independent level connectors is similar to a minimun set cover problem [1]. Therefore, to define the connectors and levels as described before, we use two heuristics. The first algorithm aims at modifying the minimum number of vertex keys to enforce the consistency requirement. The second algorithm aims at building larger level sets in order to minimize the number of levels.

The principle of the first algorithm, Algorithm 2.1, is to "expand" iteratively some vertex keys of the same length, in order to uncouple connectors with the same degree. Note that expanding a vertex key means that this vertex will be overlapping more subdomains than those given by the initial partitioning. In this algorithm we use the following notation:

- p is the number of subdomains;
- L^l is the set of vertices whose keys have a cardinality equal to l (i.e., vertices of domain-degree l);
- $V^l(i)$ is the set of vertices belonging to L^l and adjacent to vertex i;
- $Kdeg_u$ is the number of neighbors in $V^l(u)$ that have a key different from $Key(u)$.

ALGORITHM 2.1. Computation of Independent Connectors.

1. **Initialization:**
2. For each vertex $u \in \mathcal{V}$
3. $Key(u) :=$ list of subdomains containing vertex u.
4. end
5. For each vertex $u \in \mathcal{V}$
6. $Kdeg_u := |\{\ v \in V^l(u)\ /\ Key(v) \neq Key(u)\}|$.
7. end
8. For $l = 1$ to p Do:
9. $L^l = \{u \in \mathcal{V}\ /\ |Key(u)| == l\}$
10. end
11. **Main loop:**
12. For $l = 2$ to p Do:
13. For each vertex $u \in L^l$ do
14. $Key(u) := Key(u) \cup \bigcup_{k=1,\ldots,l-1}(\bigcup_{v \in V^k(u)} Key(v))$
15. end
16. While all vertices in L^l have not been processed
17. Get u the vertex in L^l such that $Kdeg_u$ is maximum.
18. $Key(u) := Key(u) \cup \bigcup_{v \in V^l(u)} Key(v)$
19. $m = |Key(u)|$
20. if $m > l$, then
21. $L^l := L^l \setminus \{u\}$
22. $L^m := L^m \cup \{u\}$
23. For each $v \in V^l(u)$
24. $Kdeg_v := Kdeg_v - 1$
25. end
26. end
27. end
28. end

At the initialization of Algorithm 2.1, the key of each vertex u is set to the list of subdomains that contain vertex u. In the main loop, the keys of each vertex u of degree 2 that is connected to another vertex of degree 2 with a different key are expanded by the union of all the keys from vertices in $V^l(u)$. Then the vertex u is set in a higher degree connector. This is continued iteratively over all the sets of connectors that have the same degree. Lines 13–15 of Algorithm 2.1 ensure the property of consistency (Definition 2.2). In this algorithm, the traversal order of the vertices of same degree is important since two different traversals can give very

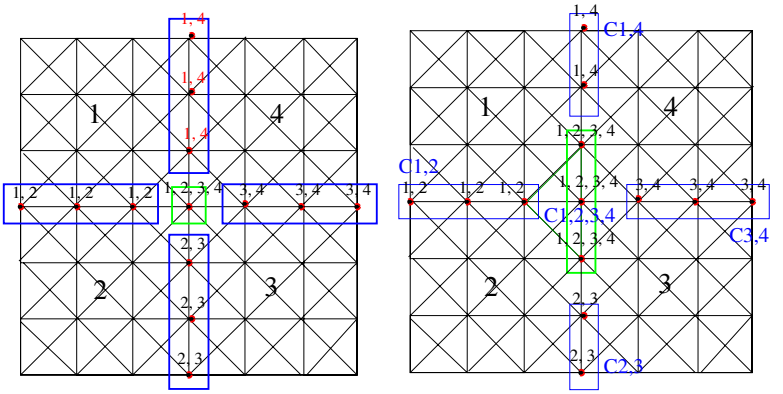


FIG. 2.9. *Partition of a* 9*-point grid into* 4 *subdomains. The connectors and values of the keys are shown at the beginning and at completion of the algorithm.*

different results with a different number of vertices moved to an upper level. This heuristic finds a good traversal of the vertices of the same degree in order to modify as few vertex keys as possible to enforce the consistency condition. The principle is to define $Kdeg(u)$ as the number of neighbors in $V^l(u)$ which have a key different from $Key(u)$ and then choose the next vertex u in the traversal as the one that has the maximum $Kdeg(u)$ (line 17). For this we need to update on the fly the $Kdeg$ of the neighbors when a vertex expands its key (lines 23, 24, 25). An implementation strategy is to sort the vertices by increasing order of $Kdeg$ in a max heap; the cost to obtain the maximum vertex is then constant, and the cost to update the rank of a vertex neighbor is bounded by $O(log(n_i))$, where n_i is the number of vertices in the interfaces. If we denote by d_{max} the maximum degree of a vertex in the interface subgraph, then a pessimistic bound on the complexity of the main loop of Algorithm 2.1 is in $O(n+d_{max}.n_i.log(n_i))$. The initialization requires a loop over all the vertices to decide which ones are on an interface; thus it has a linear complexity with respect to the total number of vertices.

At the end of the algorithm, the key of a vertex u indicates the connector with which it has been associated, i.e., $C_{i_1,\ldots,i_m} = \{u \in \mathcal{V} \mid Key(u) = \{i_1,\ldots,i_m\}\}$. Figure 2.9 illustrates the results of this algorithm on the 9-point grid example.

2.3. The levelization algorithm. We have seen earlier that defining the level label of a connector as equal to its degree may yield a poor interface decomposition because an irregular domain partition may produce a large number of levels. A better definition of the levels from the keys is required to alleviate this problem. This definition requires an order relation $\overset{c}{>}$ for pairs of adjacent connectors.

DEFINITION 2.3. *For two adjacent connectors* C_1 *and* C_2*,*

$$C_1 \overset{c}{>} C_2 \quad \longleftrightarrow \quad Key(C_2) \subsetneq Key(C_1).$$

Note that for consistent decompositions, two adjacent connectors are always comparable with the relation $\overset{c}{>}$. The idea of the level computation algorithm is to compute iteratively the level as the set of local minimum for $\overset{c}{>}$ in the set of all unleveled connectors.

On irregular graphs, the connector decomposition can produce connectors that are of no benefit to the main goal of separating the graph into levels of independent

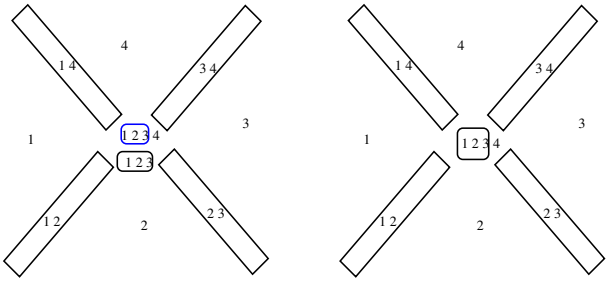


FIG. 2.10. *On the left, connector $C_{1,2,3,4}$ is singular. On the right, the singular connector has been merged.*

sets. For example, Figure 2.10 shows a decomposition which produces two connectors $C_{1,2,3}$ and $C_{1,2,3,4}$ which should clearly be merged into one. Indeed, it is meaningless to differentiate between these two connectors because connector $C_{1,2,3,4}$ does not separate two connectors of the level immediately above it. A connector such as $C_{1,2,3,4}$ in Figure 2.10 will be termed a *singular connector*. A singular connector is always merged with a connector of a level immediately below it. The merge operation takes place just after a new level has been computed. The principle, described in Algorithm 2.3, is to merge any neighbor of the current level connectors such that the new merged connector is still a local minimum for $\overset{c}{>}$ in the set of all unleveled connectors.

In Algorithms 2.2 and 2.3, $\mathcal{C}_{unleveled.}$ is the set of all unleveled connectors, and $\mathcal{C}^i$ is the set of level-i connectors.

ALGORITHM 2.2. Level Computation Algorithm.

1. $i := 0$;
2. While $\mathcal{C}_{unleveled.} \neq \emptyset$
3. $\quad i := i + 1$
4. $\quad \mathcal{C}^i :=$ the set of all minimum connectors in $\mathcal{C}_{unleveled.}$ for $\overset{c}{>}$.
5. $\quad \mathcal{C}_{unleveled.} := \mathcal{C}_{unleveled.} \setminus \mathcal{C}^i$
6. $\quad$ Merge all the singular connectors for level i.
7. end

ALGORITHM 2.3. Merge singular connectors for level i.

1. For each connector $C \in \mathcal{C}^i$ Do:
2. $\quad$ For each connector $V \in \mathcal{V}(C) \bigcap \mathcal{C}_{unleveled.}$
3. $\quad\quad$ if $C \cup V$ is a minimum for $\overset{c}{>}$ in $\mathcal{C}_{unleveled.} \cup \mathcal{C}^i$, then
4. $\quad\quad\quad C = C \cup V$
5. $\quad\quad\quad Key(C) = Key(C) \cup Key(V)$
6. $\quad\quad\quad \mathcal{C}_{unleveled.} = \mathcal{C}_{unleveled.} \setminus V$
7. $\quad\quad$ End
8. $\quad$ End
9. End

One can notice that Algorithm 2.2 will always produce a number of levels that is less than or equal to the number of different degrees in the connector set. If we denote by D_{max} the maximum degree of a connector in the connector graph, by l the number of levels, and by c the number of connectors, then a pessimistic bound of the complexity is in $O(l.c.D_{max})$.

3. Parallel incomplete factorizations based on hierarchical interface decompositions. The previous section defined what we termed a "hierarchical interface decomposition" related to a partitioning of the graph $\mathcal{G}$. This hierarchical interface decomposition consists of several levels of connectors which have the property of forming an independent set at each level. If the global matrix is (implicitly) reordered by level, starting with the nodes of level 1, followed by those of level 2, and so on, the independent sets will lead naturally to parallel ILU-type factorizations if careful dropping is used. This technique bears some similarity to the ARMS discussed in [16, 9] with a major difference being that in ARMS the independent sets are determined dynamically, whereas in the new algorithm, the ordering is set in advance. In addition, dropping is especially adapted to the partitioning, as is explained in the next section.

3.1. Consistent dropping strategies. The interface decomposition and the keys associated with each node will be intimately related to the ILU factorization process. In particular, this decomposition and the keys will help define a suitable fill-pattern. The goal is to retain as much parallelism as possible in the ILU factorization.

The first principle of the elimination is to proceed by levels: Nodes in the connectors of level 0 are eliminated first, followed by those in connectors of level 1, etc. All nodes of the zeroth level can be eliminated independently since there is no fill-in between nodes i and j of two different connectors of level 0 (interior of the subdomain). In contrast, fill-ins may appear between connectors from level 1 down to the last level of the interface connectors. These fill-ins will cause some uncoupled connectors to become coupled, and so they will no longer form an independent set. This will result in reduced parallelism.

Recall that in the standard static ILU factorization a nonzero pattern is selected in advance based on the sparsity pattern of A. For example, the nonzero pattern for the ILU(0) factorization is simply the set of all pairs (i, j) such that $a_{ij} \neq 0$; i.e., it is equal to the nonzero pattern of the original matrix. For details and extensions, see a standard text on iterative methods, e.g., [14, 2]. As was mentioned above, certain nonzero patterns will allow more parallelism than others during the factorization. Certain choices of nonzero patterns may also lead to simpler implementations.

We consider two possible options. The first is not to allow any fill-ins between two uncoupled connectors of the same level. In this fashion, the elimination of all connectors at a certain level can always proceed in parallel. The second option, which is less restrictive, is to allow fill-ins but to restrict them to occur only when i, j are associated to a common subdomain. In this case, elimination should be done in a certain order, and parallel execution can be maintained by exploiting independent sets. Figure 3.2 draws the block structure of the global matrix reordered according to the hierarchical interface decomposition (HID) of the domain illustrated in Figure 3.1. Figure 3.2 illustrates the two dropping strategies.

3.2. Strictly consistent nonzero patterns. A nonzero pattern P is said to be strictly consistent with the HID if it does not allow any fill-in between any two connectors of the same level. This implies that an entry (i, j) will be in the nonzero pattern P only if vertices i and j belong to two adjacent connectors. In regard to the consistency requirement, this means that

$$P \subset \{(i,j) \mid Key(i) \subset Key(j) \text{ or } Key(i) \subset Key(j)\} .$$

In a block representation of the matrix, where blocking is done according to the levels, this means that this approach would discard all fill-ins between blocks, thus preserving

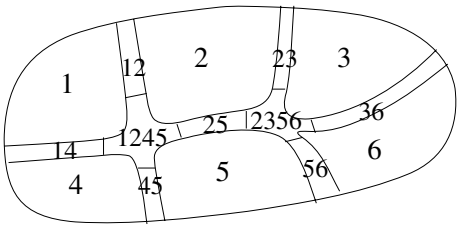


FIG. 3.1. *Domain partition into* 6 *subdomains. Connectors and their keys are represented.*

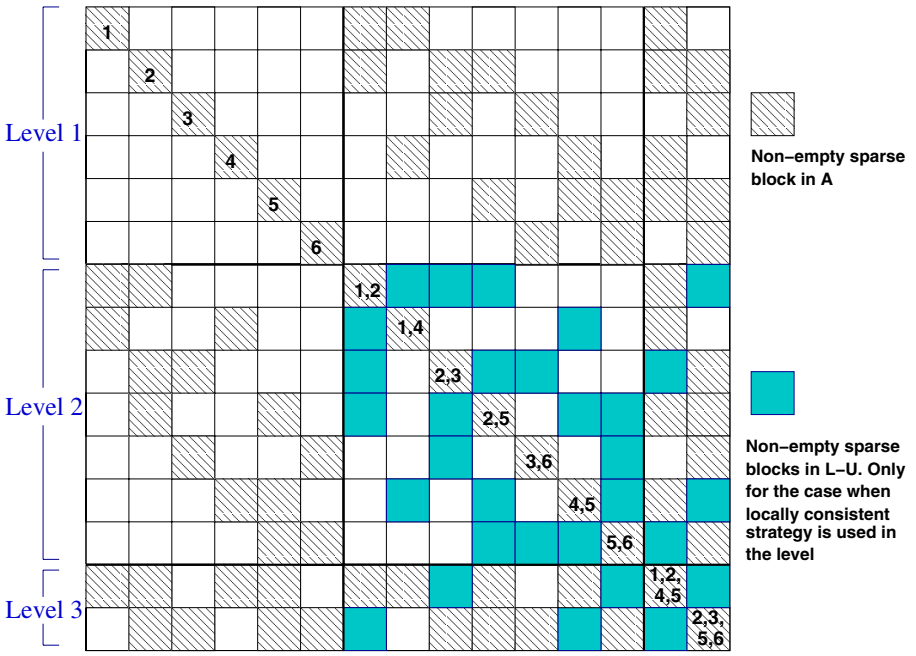


FIG. 3.2. *Block decomposition of the matrix according to the HID reordering. For convenience, all blocks have been drawn at the same size; using a real scale, the blocks become significantly smaller in the high levels.*

the initial global block structure. A typical way to define strictly consistent nonzero patterns is to take a common dropping strategy and restrict the dropping further by imposing the above rule.

An important consequence of the restriction embodied in the above definition is that all connectors of the same level will form an independent set throughout the elimination process. Indeed, any fill-in in the diagonal block submatrices (corresponding to connectors that have the same level label) is discarded, and thus the diagonal block structure is kept. In particular all eliminations will proceed in parallel at all stages. There remains only to consider how to handle the rows resulting from the elimination in different processors when these rows share two or more subdomains. Since these nodes are represented in more than one subdomain, they will undergo different modifications in each subdomain because the elimination performed in each subdomain is incomplete. In order to complete the process, we have two options. If i is the vertex index, we can have each processor in $Key(i)$ perform the same operations.

Communication is required locally so that all processors will get the modifying rows. The second option is to select one processor in $Key(i)$ to perform all modifications. Once the modifications are completed, this processor will then send the final row to its neighbors. In the algorithms to be described later, this option is retained.

3.3. Locally consistent nonzero pattern. This type of nonzero pattern is less restrictive and allows *fill-ins* only between connectors belonging to the same subdomain. In other words, a fill-in is allowed in location (i, j) if $Key(i) \cap Key(j) \neq \emptyset$, as stated in the following definition.

DEFINITION 3.1. *The nonzero pattern P is said to be locally consistent with the hierarchical interface decomposition if*

$$P \subset \{(i, j) \mid Key(i) \cap Key(j) \neq \emptyset\} .$$

Thus, an entry in location (i, j) of L or U is permitted only if $Key(i)$ and $Key(j)$ have at least one common index, i.e., only when i and j share one common subdomain. This permits an easy characterization of fill-ins based on the interface decomposition. Again, one can select a fill-in pattern such as ILU(k) or a dynamic one such as ILU with threshold (ILUT)—with an additional constraint that no fill-in is allowed in locations not satisfying the compatibility property defined above. For the special case of ILU(k) with $k \leq 2$, it is interesting to note that no additional dropping would be exercised for the locally consistent dropping pattern for any number of domains. This property is a direct consequence of Lemma 2.2, since a scalar ILU(k) nonzero pattern is always contained in the block ILU(k) nonzero pattern.

In this situation, elimination can proceed simultaneously for nodes of two connectors of the same level whose keys are disjoint. As mentioned above, Gaussian elimination will eliminate the unknowns of the connectors level by level. A good way to exploit parallelism is to partition the set of connectors in the level into maximum weighted independents sets. Here, the independent sets are for a graph whose vertices are connectors (identified by a key K) and whose edges represent the binary relation $Key_1 \cap Key_2 \neq \emptyset$.

DEFINITION 3.2. *A set of connectors is "independent" if any two connectors of the set have disjoint keys. Such a set is "maximal" if it cannot be augmented by other connectors.*

Note that a maximal independent set is not one which has a maximum cardinality among all possible independent sets. The term "block independent sets" or "group independent sets" was used in [16, 9] for sets of subsets of vertices which are uncoupled. The idea of using independent sets of nodes (or subsets of nodes) in parallel Gaussian elimination is not new. Multicoloring was employed, for example, to define parallel ILU factorizations [7, 14]. Essentially, a global order—called "schedule" in [7]—in which to process the connectors is required. This global order can be defined by processing all those connectors which will not be coupled by any fill-in.

Figure 3.3 illustrates the elimination order that can be obtained in our incomplete factorization when a locally consistent strategy is used in all levels, on a simple example. With a strictly consistent strategy there is no need to find any special elimination connector ordering in a level since connectors can be eliminated independently.

3.4. Ordering of the unknowns inside the connector subsets. In addition to the global ordering of the connectors, a local ordering can be applied within each connector. In our codes, we apply a reordering of the interior domain unknowns (i.e., nodes within each level-1 connector are reordered) to minimize fill-in for direct

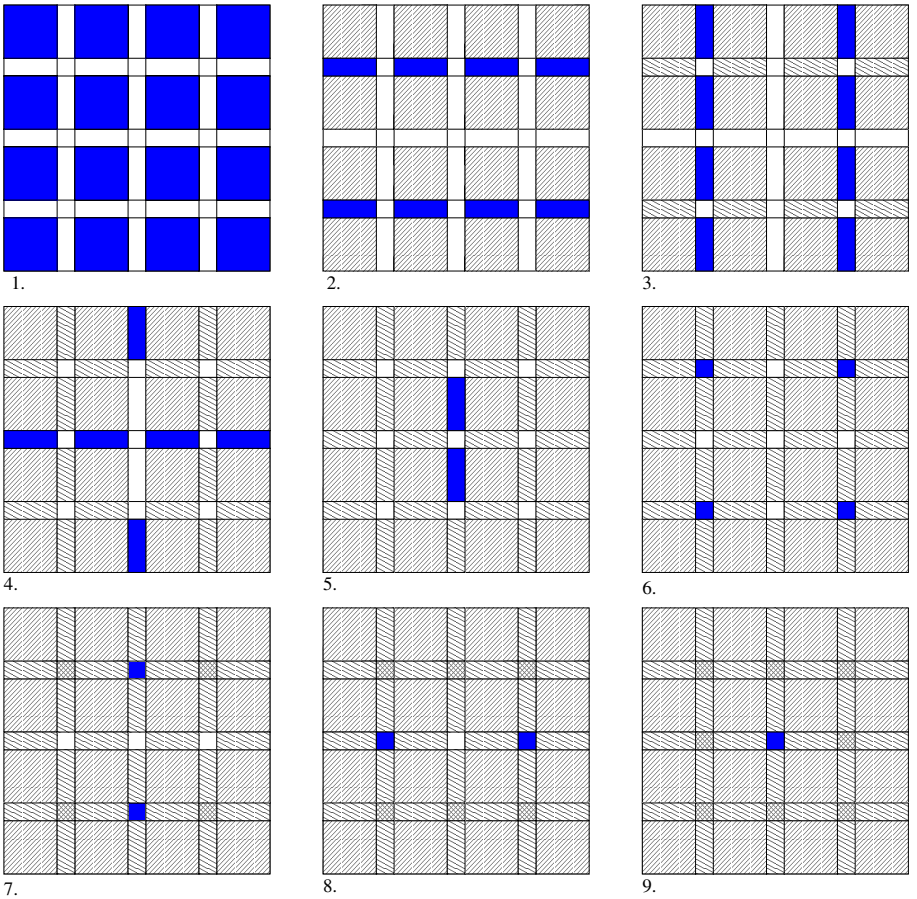


FIG. 3.3. *Elimination of the connectors. The dense connectors are the maximum independent set selected for elimination. The dashed connectors are those already eliminated.*

factorization. Such reorderings are provided by METIS. No special reordering is applied to the higher-level connectors.

3.5. Interpretation and generalizations. An interesting observation regarding the consistent dropping strategies just defined is that they can be viewed from the angle of quotient graphs. Define the quotient graph of connectors as a graph whose vertices represent the connectors and the edges the adjacency relationship between them. The dropping strategy can be viewed as a 2-level strategy which considers dropping within each connector independently from the dropping strategy between connectors. The former uses the original graph, whereas the latter uses both the original graph and the connector graph.

In particular, notice that because the partitioning considered is edge-based (i.e., an edge never straddles two different subdomains), an edge in the connector graph between two connectors C_1 and C_2 implies that the two connectors are in the same subdomain. Consider now a zero fill strategy *on the connector graph*. In this strategy no edge will be created during the elimination process. Of course the ordering of the connector graph is consistent with that of the reordered matrix. An illustration is shown in Figure 3.4. In the case of strictly consistent dropping strategy, the block fill-

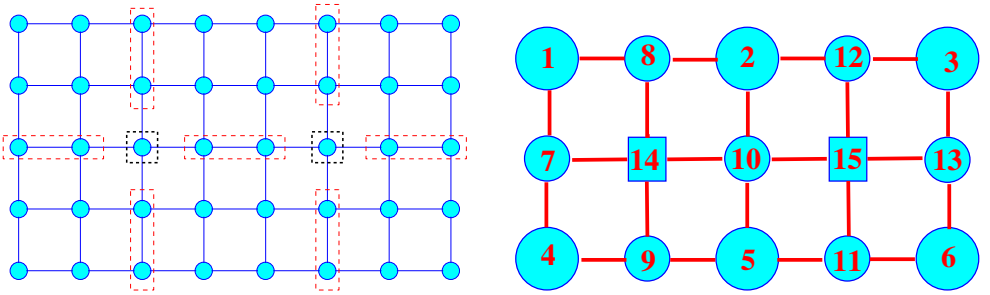


FIG. 3.4. *Hierarchical decomposition and corresponding quotient graph. Large circles show level-*1 *sets, small circles level-*2 *sets, and squares level-*3 *sets.*

in pattern is the same as that implied by an ILU(0) pattern on the connector quotient graph. Fill-in is obviously allowed within each connector. Because connectors that are adjacent in the quotient graph belong to the same domain, the second strategy defined earlier corresponds to allowing fill-ins of level ≤ 2 in the quotient graph (see Lemma 2.2). Generalizations can now be made. On can imagine applying a block ILU(k) with $k > 2$ on the connector quotient graph in the same way that this is defined in ILU(k) factorizations. Of course such dropping rules will entail more communication and a more complicated processing; for instance, fill-ins may appear between connectors which do not share a common subdomain.

4. Parallel factorization. The partition of the unknowns into connectors induces a block structure of the matrix which enables an easy expression of the parallel algorithm. Figure 4.1 shows one such structure. The figure zooms in on the block matrix associated with subdomain 2. The diagonal blocks correspond to all the connectors which are part of domain 2, and the off-diagonal blocks correspond to the coupling between these connectors. If each subdomain k is assigned to processor k, then the label on each block is a list of all the processors which own a copy of the block. For example, the subblock in position (4,6) of the block matrix is labeled with 2 and 5, and therefore it is located in both processors 2 and 5. The label of a diagonal block is the key of the corresponding connector as defined previously. The label of an off-diagonal block (i, j) in the local block matrix is the intersection of the labels of the blocks (i, i) and (j, j). For convenience, in the rest of this paper we will denote by $Key(i, j)$ the label of the block (i, j). Therefore, using previous notation,

$$Key(i, j) = Key(i) \cap Key(j).$$

The parallel factorization algorithm is given below (Algorithm 4.1). The following symbols are used in the algorithm.

- E represents the local matrix associated with processor p.
- C represents a local matrix where a block $C_{i,j}$ is a temporary location to add up all the updates to the block $E_{i,j}$. This matrix is empty at the start of the algorithm.
- `leader`$(p_1, \ldots, p_m)$ is a function which selects a processor from a processor subset (see below for details).
- P is the nonzero block pattern of the matrix.

The function "leader" is used to choose a unique processor to process a block matrix operation when several choices of processor are possible. In other words, this function

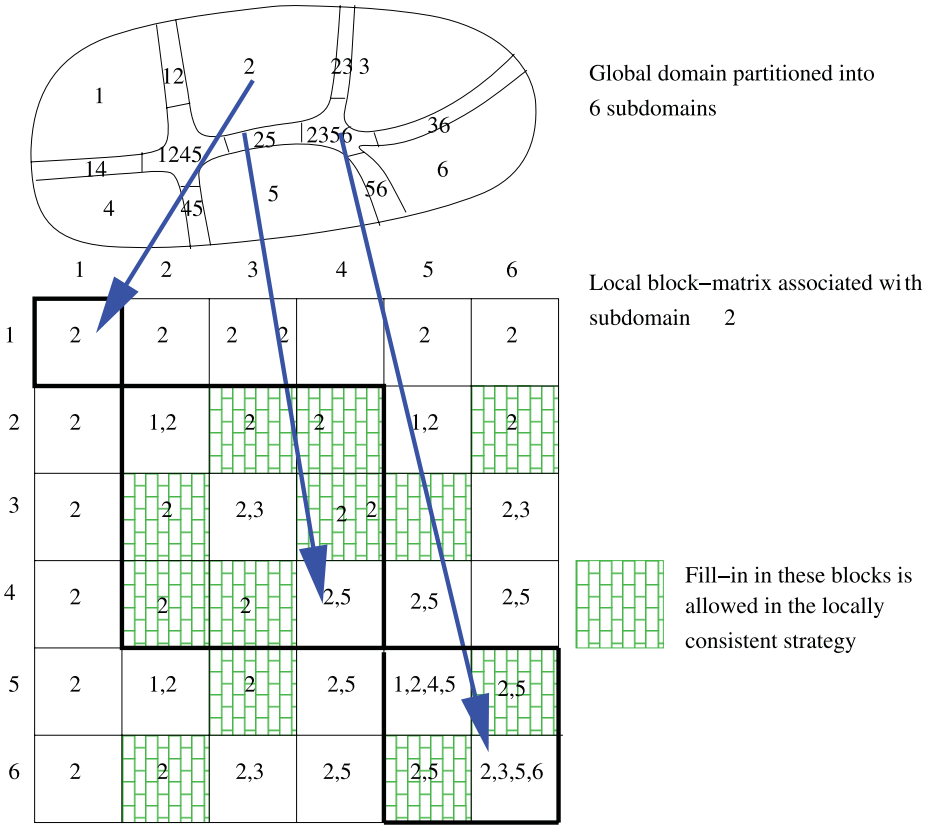


FIG. 4.1. *Block partition of a local matrix.*

is used to balance the workload and storage related to blocks overlapped by several local matrices on different processors.

Note that because of the ordering into levels, there is no explicit distinction between the levels in the algorithm. This distinction is inherent to the algorithm and is implicitly defined by the keys.

ALGORITHM 4.1. Parallel Elimination Algorithm.

0. Let N = block dimension of local matrix, and p = label of this processor.
1. For $i := 1$ to N
2. If $p = \text{leader}(Key(i,i)$
3. Receive and add in $E_{i,i}$ all $C_{i,i}$ from processors $\in Key(i,i)$
4. Factorize $E_{i,i}$ into $L_i.U_i$
5. Send $\{L_i, U_i\}$ to processors $\in Key(i,i) \setminus p$
6. Else
7. Receive L_i, U_i from leader$(Key(i,j))$
8. EndIf
9. For $j := i+1$ to N, such that $(i,j) \in P$ and $p = \text{leader}(Key(i,j))$
10. Receive and add in $E_{i,j}, E_{j,i}$ all $C_{i,j}, C_{j,i}$ from processors $\in Key(i,j)$
11. $E_{i,j} := L_i^{-1}.E_{i,j}$
12. $E_{j,i} := E_{j,i}.U_i^{-1}$
13. Send $\{E_{i,j}, E_{j,i}\}$ to processors $\in Key(i,j) \setminus p$

```
14.       EndFor
15.       For j := i + 1 to N, such that (i, j) ∈ P and p ≠ leader(Key(i, j))
16.            Receive {E_{i,j}, E_{j,i}} from leader(Key(i, j))
17.       EndFor
18.       For j := i + 1 to N, such that (j, i) ∈ P
19.            For k := j to N, such that (i, k) ∈ P
20.                 If p = leader(Key(j, i) ∩ Key(i, k))
21.                      C_{j,k} = C_{j,k} − E_{j,i}.E_{i,k}
22.                      If j ≠ k
23.                           C_{k,j} = C_{k,j} − E_{k,i}.E_{i,j}
24.                      EndIf
25.                      If all local contributions have been added up in C_{j,k}
26.                           Send {C_{j,k}, C_{k,j}} to processors ∈ Key(j, k)\p
27.                      EndIf
28.                 EndIf
29.            EndFor
30.       EndFor
31.  EndFor
```

A few additional details on the implementation will now be given. First, it is important to note that the forward and backward solves are also parallel at each level. The outer (level) loop is sequential. This is due to the structure of the diagonal blocks associated with the levels. Second, no pivoting is performed, but the standard forms of pivoting in the context of LU can be applied. For example, "static pivoting" can be performed whereby diagonal entries are increased in an attempt to prevent the need for pivoting. Alternatively, we can also perform local block pivoting whereby the pivot row is confined to stay within the dense block.

5. The preconditioning operation. The solution process consists in using the incomplete factorization as a right preconditioner combined with an iterative accelerator such as GMRES. The current version of PHIDAL includes two ways of using the incomplete factorization. If we partition the matrix A according to an ordering associated with listing interior points first, followed by interface points, then we can define a block incomplete LU factorization as follows:

$$A = \begin{pmatrix} B & F \\ E & C \end{pmatrix} \approx \begin{pmatrix} L & \\ EU^{-1} & L_S \end{pmatrix} \times \begin{pmatrix} U & L^{-1}F \\ & U_S \end{pmatrix} \equiv M.$$

Here we assume that $B \approx LU$ is an incomplete factorization of B and that $L_S U_S$ is some incomplete factorization of an approximation to the Schur complement matrix $S = C - (EU^{-1})(L^{-1}F)$. In practice, with the above factorization we need only store the factors L, U, L_S, U_S and the original blocks E, F. Given a right-hand side $y = \left(\begin{smallmatrix} y_B \\ y_C \end{smallmatrix}\right)$, the solution $x = \left(\begin{smallmatrix} x_B \\ x_C \end{smallmatrix}\right)$ to the system $Mx = y$ can be computed as follows:

$$\begin{cases} x_C = U_S^{-1} L_S^{-1} (y_C - EU^{-1} L^{-1} y_B), \\ x_B = U^{-1}(L^{-1} y_B - L^{-1} F x_C). \end{cases} \tag{5.1}$$

An alternative is to explicitly compute and store some approximations G and W to the matrices $L^{-1}F$ and EU^{-1}, respectively. In this case the solution can be obtained from

$$\begin{cases} x_C = U_S^{-1} L_S^{-1} (y_C - WL^{-1} y_B), \\ x_B = U^{-1}(L^{-1} y_B - G x_C). \end{cases} \tag{5.2}$$

We refer to this as the M_{GW} preconditioner. One advantage of using the first variant (without the approximations G, W) is that it does not require us to store the submatrices G and W. These matrices are actually available during the factorization process as they are needed to obtain S; thus no additional computation is incurred in this approach. In this version the G, W matrices are discarded after they are used and no longer needed. Another advantage is that this approach will tend to be more accurate than one which uses the approximations G, W. Therefore, the preconditioner M is usually more efficient than M_{GW}. On the other hand, the use of the G, W approximation will be slightly more expensive in general. Indeed, the number of floating point operations required to apply the preconditioner M is

$$ope_M \approx 2\,(\mathrm{nnz}(L) + \mathrm{nnz}(U)) + \mathrm{nnz}(L_S) + \mathrm{nnz}(U_S) + \mathrm{nnz}(E) + \mathrm{nnz}(F).$$

In contrast, applying the preconditioner M_{GW}, which uses the G, W approximations, costs roughly

$$ope_{M_{GW}} \approx \mathrm{nnz}(L) + \mathrm{nnz}(U) + \mathrm{nnz}(L_S) + \mathrm{nnz}(U_S) + \mathrm{nnz}(G) + \mathrm{nnz}(W).$$

In the situation when the B block is large, the M preconditioner will cost more to apply in general, depending on the sparseness of the matrices W, G. Note that similar choices and issues were also discussed in the context of the ARMS preconditioner [16]. In the numerical tests presented in the next section, the option $KeepGW = 0$ refers to the preconditioner M, and $KeepGW = 1$ refers to the preconditioner M_{GW}.

6. Numerical tests. Algorithm 4.1 allows many possible variations depending on, for example, the version of ILU that is used, whether consistent or strictly consistent dropping is applied, and the selection of the various threshold parameters for dropping small terms. The current version of PHIDAL is based on an ILUT factorization (a numerical threshold is used to drop small entries). One can use two threshold parameters in the ILUT factorization: $thresh_L$ and $thresh_U$ corresponding to the damping criterion for the lower triangular part and the upper triangular part of the matrix. In all our tests we set the same numerical threshold $thresh$ for L and U. The accelerator was GMRES with a restart parameter of 60. In section 5, we mentioned two variants for using the incomplete factorization as a preconditioner ($KeepGW = 1$ and $KeepGW = 0$). In all the tables the fill ratio is defined as the number of nonzeros stored in the preconditioner divided by the number of nonzeros in the original coefficient matrix A. Thus, the fill ratio is smaller with the option $KeepGW = 0$ than with $KeepGW = 1$.

The preconditioning strategies used in the following tests utilize a combination of strictly and locally consistent dropping. Indeed, it appears appealing to use locally consistent strategies in the first few levels only and the less accurate and costly strictly consistent strategy at other levels. In the experiments, the number "locally consistent levels" refers to the number of the first few levels (in the interface) in which dropping is performed in a locally consistent way. Strictly consistent dropping is performed thereafter.

We used the graph partitioner METIS to get the initial domain partition and to reorder the interior domain unknowns in order to reduce fill-in. Similar results can be obtained by other graph partitioners and reordering algorithms to minimize fill-in.

6.1. Discretized 3-dimensional Poisson equation. The problem solved is the elliptic partial equation $-\Delta.u = f$ on a square region with Dirichlet boundary conditions discretized with a seven-point centered finite difference scheme on an $n_x \times$

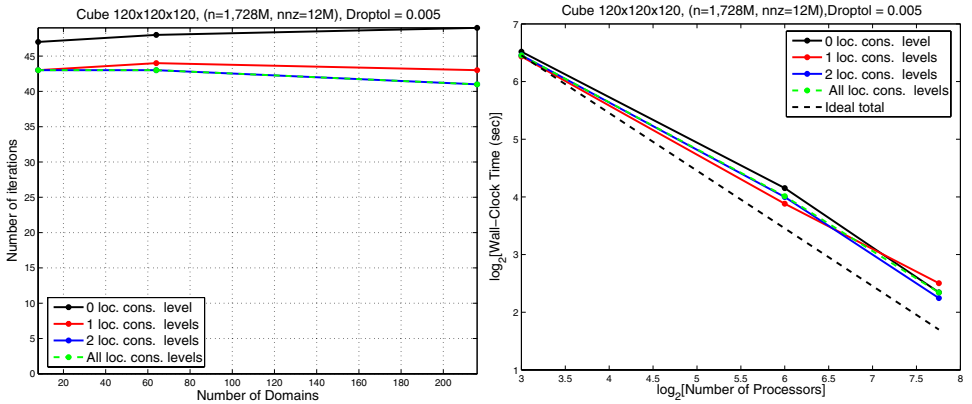


FIG. 6.1. *Performance of GMRES accelerated with PHIDAL for a* $120 \times 120 \times 120$ *Poisson equation as the number of processors varies.*

$n_y \times n_z$ (3-dimensional) grid, excluding boundary points. The mesh is mapped to a virtual 3-dimensional grid of $p_x \times p_y \times p_z$ processors. Therefore, subcubes of $r_x = n_x/p_x$, $r_y = n_y/p_y$, and $r_z = n_z/p_z$ points are mapped into each processor in the x, y, and z directions, respectively.

The numerical dropping thresholds in the factorization were set to 0.005. With this drop tolerance the fill factor, i.e., the ratio of the memory required by the factorization over that for the original matrix, was about 5. The relative residual precision of the solution is 10^{-7}. The machine used to run the tests was the IBM SP3 (named seaborg) of the NERSC, a computer equipped with NightHawks nodes clocked at 375Mhz. Each node has 16 processors which share a common pool of between 16 and 64 GB of shared memory.

In this case, the decomposition in the interface is naturally defined by the partition (see Figure 2.1 in section 2). The left part of Figure 6.1 suggests that the number of iterations required for convergence is almost independent of the number of subdomains. The various cases shown correspond to the number of levels in the interface on which locally consistent dropping is performed. In the case of a cube there are 4 levels of connectors (interior domain points, faces, edges, and cross points), and we can choose a locally consistent strategy for only three connector levels in the interface (faces, edges, and cross points). Thus the curves for *all locally consistent levels* correspond to the case of 3 *locally consistent levels*. The right curves show that the time to precondition and solve the problem scales very well with the number of processors; this shows that for this model problem and for a good initial partition of the graph the parallel factorization and the triangular solves are almost perfectly scalable in time.

We also ran an isoscalability test on the same problem, where each processor was assigned a subdomain of size $40 \times 40 \times 40$. This means, for example, that on a virtual $3 \times 3 \times 3$ processor grid, the whole discretized problem is a $120 \times 120 \times 120$ mesh ($n \approx 1.7 * 10^6, nnz \approx 12 * 10^6$). Similarly, on a virtual $6 \times 6 \times 6$ processor grid the discretized problem corresponds to a $240 \times 240 \times 240$ mesh ($n = 13.8 * 10^6, nnz = 96.4 * 10^6$). Table 6.1 shows the iteration counts and the execution times for different numbers of processors. The numerical dropping thresholds in the factorization were set to 0.01 for these tests; with this threshold the fill rate in the factors was about 4.

TABLE 6.1
Scaled performance experiment: 3-dimensional grids $40\times40\times40$ *per processor, locally consistent strategy in all levels, numerical threshold set to* 0.01*, and* $KeepGW = 1$.

$p_x \times p_y \times p_z$	Facto time in sec	Solve time in sec	Iter nbr	Fill ratio
3x3x3	6.10	28.85	52	3.98
4x4x4	6.58	42.05	63	3.98
5x5x5	6.80	57.60	78	3.99
6x6x6	6.81	66.53	84	3.99

These results were obtained with preconditioner M_{GW} (option $KeepGW = 1$).

6.2. Tests on a magnetohydrodynamic problem. The matrices used in the following tests arise from a magnetohydrodynamic flow problem that couples Maxwell's equation

$$\frac{\partial \mathbf{B}}{\partial t} - \nabla \times (\mathbf{u} \times \mathbf{B}) + \eta \, \nabla \times (\nabla \times \mathbf{B}) + \nabla \mathbf{q} = 0,$$
$$\nabla \cdot \mathbf{B} = 0,$$

with the Navier–Stokes equation. Coupling is through the body force $\mathbf{f} = \frac{1}{\mu}(\nabla \times \mathbf{B}) \times \mathbf{B}$. In the following test, u is imposed, and Maxwell's equation is solved for B. The finite element mesh is composed of mixed tetrahedra and hexahedra elements. Two matrices, MHD1 and MHD2, are used. The matrix MHD1 is of size $n = 485,597$ and has $nnz = 24,233,141$ nonzeros. MHD2, which uses a different Reynolds number, is of size $n = 470,596$ and has $nnz = 23,784,208$ nonzero entries. For additional details on this problem, see [15, 19]. The machine used to run these tests was a cluster of IBM power5 SMP nodes, where each node is composed of 16 processors sharing a common pool of 32 GB of memory.

Levelization algorithm. Figure 6.2 shows the algorithm of levelization in action for both MHD1 and MHD2 when the number of processors used is 256. The plots show the various levels obtained with the straightforward use of Algorithm 2.1 and then with the improved strategy based on Algorithm 2.2. For each of these levels the $\log_2$ of the number of nodes at that level is plotted. In the ideal situation we would have a small number of levels or, equivalently, we should obtain very few levels that have a small number of nodes as this hampers parallel execution. As can be seen, this is the case for both matrices. We should point out that for MHD2, the basic algorithm created 4 empty levels, i.e., levels with zero nodes, out of the 23 levels. These are the levels 18, 19, 20, and 22. Also, as expected, the effect of the levelization is not nearly as significant for fewer processors.

Table 6.2 gives the times (in seconds) to perform the hierarchical interface decomposition ordering for different numbers of domains on the MHD1 matrix. The first line indicates the percentage of nodes in the interface from the total number of nodes for the domain decomposition given in input. The second line gives the number of levels obtained before and after the levelization algorithm. The last line gives the time needed to perform the hierarchical interface decomposition ordering: It includes the times of Algorithm 2.1 (independent connector computation) and the times of Algorithms 2.2 and 2.3 (levelization algorithms).

We omit the details on the various times spent in each of these algorithms (we report only the total) because Algorithms 2.2 and 2.3 cost very little relative to Algorithm 2.1. The fact that the time of the ordering for 32 domains is slightly

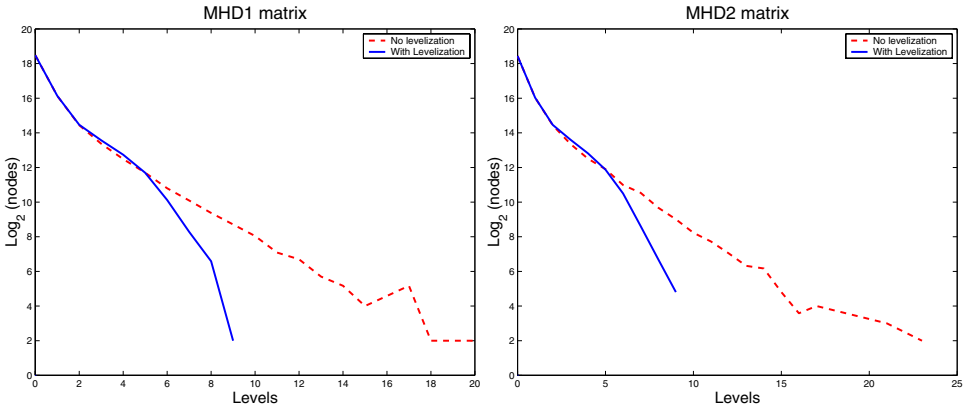


FIG. 6.2. *Effect of the levelization algorithm. The plot shows the* $\log_2$ *of the number of nodes at each level obtained for a* 256 *domain partition. As desired, the levelization results in fewer levels (*9 *instead of* 20*) and, indirectly, in far fewer small-size levels.*

TABLE 6.2

*Hierarchical decomposition for MHD*1 $n = 485597$. *The first line gives the number of domains. The second line gives the percentage of nodes in the interface between domains. The third line gives the number of connector levels obtained before/after the levelization algorithm. The fourth line gives the total time to compute the hierarchical interface decomposition.*

Number of domains	16	32	64	128	256
Nodes in interface	9.1 %	12.3 %	15.4 %	20.1 %	24.4 %
Number of levels	6/5	10/6	11/8	15/9	20/9
Time (sec)	4.97	4.79	5.11	7.01	11.28

smaller than for 16 processors is not significant and is certainly due to the accuracy of the timing.

The ordering time does not vary very much until the number of domains reaches 64. This is explained by the fact that in these cases a major part of the time is spent initializing Algorithm 2.1, consuming a time that is linear in the total number of nodes (interior domain and interface). When the number of domains increases, the number of nodes in the interface grows and the time spent in the main loop of Algorithm 2.1 grows, but the whole ordering time is still acceptable considering that the domain decomposition is usually done only once in a realistic physical simulation.

Effect of the type of dropping. The next experiments will show the effect of the dropping strategy on the quality of the preconditioner. Recall that we can define a strictly consistent scheme whereby no fill-in is allowed between connectors and locally consistent schemes whereby fill-ins are allowed between connectors at a given level when these connectors are located in the same processor. We can also define an arbitrary number of levels that remain strictly consistent. Strictly consistent dropping allows us to save memory but, as can be expected, sacrifices a little on the quality of the preconditioner. Clearly, the first level is always treated as a strictly consistent level. Beyond level 1, our code can decide to process as many levels in "locally" consistent mode as desired. It is important to process the last levels in strictly consistent mode since these will tend to generate a large amount of fill-in. Thus, a parameter is used to define how many levels from the second level down are processed in locally consistent mode. The experiments reported in Tables 6.3–6.6 show the executions obtained for

TABLE 6.3
Left-hand table is for MHD1, and right-hand table is for MHD2. Number of locally consistent levels = 1, thresh = 0.002, KeepGW = 1.

Proc	Facto (sec.)	Solve (sec.)	Total (sec)	Iter	Fill ratio
2	25.79	38.42	64.21	68	1.66
4	14.31	21.95	36.26	69	1.67
8	9.01	11.27	20.28	70	1.67
16	5.20	7.68	12.88	73	1.69
32	3.49	5.23	8.72	75	1.69
64	2.21	5.10	7.31	122	1.69
128	1.61	3.05	4.66	127	1.68

Proc	Facto (sec.)	Solve (sec.)	Total (sec)	Iter	Fill ratio
2	25.85	27.59	53.44	49	1.70
4	14.21	15.78	29.99	49	1.71
8	8.79	8.00	16.79	50	1.71
16	5.39	5.13	10.52	51	1.73
32	3.46	3.65	7.11	53	1.73
64	2.29	2.31	4.6	54	1.73
128	1.62	1.32	2.94	59	1.72

"zero" locally consistent levels (meaning that all levels are treated in strictly consistent mode), one locally consistent level (meaning that the second level is treated in locally consistent mode), and all locally consistent levels. Table 6.3 reports results obtained with the option $KeepGW = 1$ while Tables 6.4–6.6 were obtained with the option $KeepGW = 0$. Table 6.3 is given as a comparison to Table 6.5 to show the effect of the option $KeepGW$. In the following experiments the threshold was always set to 0.002. The restart parameter in GMRES was again 60, and the stopping criterion was a reduction of the residual norm by a factor of 10^{-7}.

The main observation from the experiments is the remarkable behavior of the iteration number for the "all" consistent levels and, to a lesser extent, for the one consistent level case. This iteration number remains about the same. On the other hand, this comes at a higher factorization cost, especially for the "all" locally consistent case.

The compromise of using one locally consistent level seems a good one for this case. Note that this experiment suggests that using one or a few locally consistent levels would improve robustness without sacrificing performance too much.

On these problems, with higher numbers of locally consistent levels, the iteration counts remain small while the overall solution times as well as memory costs are higher. This trend is all the more important when the number of processors increases. It is difficult to set a general rule on the number of locally consistent levels to use, which may vary as a function of the desired precision, the matrix condition number, and the connectivity of the adjacency graph of the matrix. Nevertheless, one locally consistent level should be a good choice in general since the bulk of the unknowns are usually interior points and points of the first level of connectors in the interface, whereas the higher connector levels contain few unknowns but require much more communication and synchronizations between processors when they are treated with a locally consistent strategy.

Comparison with pARMS. We compared the performance of PHIDAL with that of pARMS on the MHD matrices discussed in earlier sections. These comparisons are not easy to set up because pARMS uses different orderings which depend on numerical properties (see [9] for details) in the interior subdomain as well as different rules for dropping small terms in the ILUT factorization. We used a set of parameters which seem to offer an acceptable comparison in terms of fill-in ratio with PHIDAL. Because an iterative solver will usually require fewer iterations and less time if more fill-in is allowed in the preconditioner, an ideal comparison of quality would consist of testing both algorithms under the same (or close) memory requirements. Therefore, the parameters were set to achieve a fill-factor which is close for both methods. While

TABLE 6.4
Left-hand table is for MHD1, and right-hand table is for MHD2. Number of locally consistent levels = 0, thresh = 0.002, KeepGW = 0.

Proc	Facto (sec.)	Solve (sec.)	Total (sec.)	Iter	Fill ratio
2	25.46	57.62	83.08	68	1.58
4	13.98	25.94	39.92	65	1.50
8	8.22	13.02	21.24	64	1.42
16	4.81	8.06	12.87	67	1.32
32	2.74	7.22	9.96	111	1.21
64	1.73	3.93	5.66	113	1.08
128	1.02	2.54	3.56	130	0.95

Proc	Facto (sec.)	Solve (sec.)	Total (sec.)	Iter	Fill ratio
2	25.90	39.68	65.58	47	1.62
4	14.14	19.41	33.55	45	1.53
8	8.56	9.32	17.88	46	1.47
16	4.83	5.12	9.95	45	1.34
32	2.79	3.19	5.98	51	1.22
64	1.76	1.78	3.54	54	1.11
128	1.01	1.14	2.15	64	0.96

TABLE 6.5
Left-hand table is for MHD1, and right-hand table is for MHD2. Number of locally consistent levels = 1, thresh = 0.002, KeepGW = 0.

Proc	Facto (sec.)	Solve (sec.)	Total (sec)	Iter	Fill ratio
2	25.52	58.39	83.91	68	1.58
4	14.31	29.85	44.16	65	1.51
8	9.01	12.97	21.98	62	1.43
16	5.31	8.38	13.69	66	1.33
32	3.53	7.92	11.45	107	1.23
64	2.26	4.31	6.57	106	1.12
128	1.63	2.66	4.29	117	1.02

Proc	Facto (sec.)	Solve (sec.)	Total (sec)	Iter	Fill ratio
2	25.90	39.58	65.48	47	1.62
4	14.25	19.80	34.05	44	1.53
8	8.82	9.32	18.14	45	1.47
16	5.24	5.21	10.45	44	1.35
32	3.51	3.37	6.88	47	1.24
64	2.33	1.98	4.31	50	1.16
128	1.63	1.31	2.94	57	1.04

TABLE 6.6
Left-hand table is for MHD1, and right-hand table is for MHD2. All levels treated in locally consistent mode. Thresh = 0.002, KeepGW = 0.

Proc	Facto (sec.)	Solve (sec.)	Total (sec)	Iter	Fill ratio
2	25.54	57.68	83.22	68	1.58
4	14.34	29.30	43.64	65	1.51
8	9.23	12.89	22.12	61	1.43
16	5.60	8.66	14.26	66	1.34
32	5.18	8.55	13.73	106	1.24
64	4.68	5.59	10.27	100	1.14
128	4.95	3.90	8.85	107	1.06

Proc	Facto (sec.)	Solve (sec.)	Total (sec)	Iter	Fill ratio
2	25.90	39.64	65.54	47	1.62
4	14.25	19.51	33.76	44	1.53
8	9.20	9.83	19.03	44	1.47
16	5.59	5.37	10.96	43	1.36
32	4.98	3.68	8.66	45	1.25
64	5.27	2.55	7.82	46	1.17
128	5.34	1.88	7.22	50	1.08

pARMS allows the option of using an inner iteration for solving the Schur complement system of the interface, this feature is not yet available in PHIDAL. We use 5 inner iterations for pARMS; that is, each time the preconditioner is applied, 5 inner GMRES iterations are invoked to solve the last Schur complement system, which is a much smaller system. Another test which did not use inner iterations showed little difference in performance and is not reported here. It required roughly the same execution times but sightly more outer iterations to converge.

The *thresh* parameter in ILUT was set to 0.002 for PHIDAL and pARMS. pARMS also uses a fill parameter that was set to 50. We used a relative residual tolerance of 10^{-6} for these tests. With this tolerance pARMS gives good results on both MHD1 and MHD2 with these parameters. For a tolerance of 10^{-7} pARMS does not converge in fewer than 200 outer iterations for the MHD1 test case, though some other parameters would have allowed us to solve this problem. The results of PHIDAL are reported for the option $KeepGW = 0$, and all levels are treated in strictly consistent mode.

TABLE 6.7
Test on MHD1. Left-hand table is for PHIDAL, and right-hand table is for pARMS.

Proc	Facto (sec.)	Solve (sec.)	Total (sec)	Iter	Fill ratio
2	25.51	46.48	71.99	55	1.58
4	13.92	21.36	35.28	53	1.50
8	8.20	10.49	18.69	52	1.42
16	4.80	6.08	10.88	51	1.32
32	2.75	3.25	6.00	53	1.21
64	1.68	2.05	3.73	58	1.08

Proc	Facto (sec.)	Solve (sec.)	Total (sec)	Iter	Fill ratio
2	47.62	42.45	90.07	72	1.78
4	24.19	25.65	49.84	77	1.76
8	12.21	16.31	28.52	89	1.76
16	6.21	11.20	17.41	89	1.76
32	3.07	5.85	8.92	88	1.6
64	1.41	3.19	4.6	91	1.28

TABLE 6.8
Test on MHD2. Left-hand table is for PHIDAL, and right-hand table is for pARMS.

Proc	Facto (sec.)	Solve (sec.)	Total (sec)	Iter	Fill ratio
2	26.13	33.20	59.33	40	1.62
4	14.05	15.40	29.45	38	1.53
8	8.42	7.84	16.26	39	1.47
16	4.60	4.41	9.01	39	1.34
32	2.79	2.63	5.42	44	1.22
64	1.74	1.49	3.23	46	1.11

Proc	Facto (sec.)	Solve (sec.)	Total (sec)	Iter	Fill ratio
2	44.83	33.21	78.04	57	1.78
4	23.95	19.52	43.47	61	1.76
8	11.90	11.05	22.95	63	1.76
16	5.61	7.55	13.16	63	1.6
32	2.81	4.39	7.2	64	1.6
64	1.28	2.13	3.41	63	1.92

The slight differences of factorization time in Tables 6.7, 6.8, and 6.4 are due only to the fact that these tests were run a second time for this comparison (the relative residual precision of the solution was different).

The comparison was limited to 64 processors because the pARMS test program which uses METIS as a graph partitioner requires that the whole matrix graph be loaded on one processor. The cluster on which the tests were performed does not offer sufficient memory per processor (when we ask for 128 processors) to run METIS on the whole matrix graph on one processor. Note that this is not a limitation of the pARMS library but a limitation due to the test program (provided in pARMS) which uses METIS as a graph partitioner. It is possible to use PARMETIS to circumvent the problem, but this was not attempted.

Comparing the PHIDAL and pARMS total execution times shows a slight superiority of PHIDAL over pARMS for a small number of processors, but the times become quite close when the number of processors increases. In both cases, the number of iterations is generally lower for PHIDAL than for pARMS.

Since this comparison is for only two sample problems which are of the same type, it is not necessarily representative of a general situation. What can be said is that, overall, the performance is rather comparable for this problem and that more testing is required, specifically with a larger number of processors and larger problems, to reach a conclusion. Such an exhaustive comparison is beyond the scope of this paper.

7. Concluding remarks. We have presented a Schur complement-type preconditioner which is designed to work for general sparse linear systems. The preconditioner is a multistage or multilevel technique which exploits a graph-based interface structure. Experiments show that the method scales quite well for model Laplacean problems, both in terms of time and iterations. Scalability was also shown to be quite good for general sparse systems arising from harder problems.

A number of improvements are still possible and are currently being investigated. The first concerns the inclusion of pivoting. Adding some form of local pivoting to PHIDAL is straightforward. A second issue is related to the graph partitioning.

One may ask if it is possible to develop partitioning strategies that are especially tailored to enhance the overall performance of PHIDAL. In fact, partitionings that yield balanced graphs (with overlap) are still difficult to obtain in spite of the existence of excellent graph partitioners such as METIS. Finally, we plan to explore the impact of including an inner accelerator for solving the systems which arise at each level. This is a natural extension given the local matrix structure provided by the hierarchical interface decomposition.

Acknowledgments. The authors benefited from insightful discussions with John Wallis on the topic of hierarchical interface decomposition. In particular, the levelization algorithm was inspired from suggestions he made on reducing the number of levels and connectors. The authors also acknowledge Zhongze Li's extensive help in setting up the comparison with pARMS. Finally, we wish to thank two anonymous referees for their very careful reviews of the paper.

REFERENCES

[1] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccameta, and M. Protasi, *Complexity and Approximation*, Springer-Verlag, Berlin, 2003.
[2] O. Axelsson, *Iterative Solution Methods*, Cambridge University Press, Cambridge, UK, 1994.
[3] R. E. Bank and C. Wagner, *Multilevel ILU decomposition*, Numer. Math., 82 (1999), pp. 543–576.
[4] E. F. F. Botta and F. W. Wubs, *Matrix renumbering ILU: An effective algebraic multilevel ILU preconditioner for sparse matrices*, SIAM J. Matrix Anal. Appl., 20 (1999), pp. 1007–1026.
[5] J. H. Bramble, J. E. Pasciak, and A. H. Schatz, *The construction of preconditioners for elliptic problems by substructuring.* I, Math. Comp., 47 (1986), pp. 103–134.
[6] C. Farhat and F. X. Roux, *Implicit parallel processing in structural mechanics*, Comput. Mech. Adv., 2 (1994), pp. 1–124.
[7] D. Hysom and A. Pothen, *A scalable parallel algorithm for incomplete factor preconditioning*, SIAM J. Sci. Comput., 22 (2001), pp. 2194–2215.
[8] G. Karypis and V. Kumar, *MeTis: Unstructured Graph Partitioning and Sparse Matrix Ordering System, Version* 4.0, September 1998.
[9] Z. Li, Y. Saad, and M. Sosonkina, *pARMS: A parallel version of the algebraic recursive multilevel solver*, Numer. Linear Algebra Appl., 10 (2003), pp. 485–509.
[10] M. Magolu monga Made and H. A. van der Vorst, *A generalized domain decomposition paradigm for parallel incomplete LU factorization preconditionings*, Future Generation Computer Systems, 17 (2001), pp. 925–932.
[11] M. Magolu monga Made and H. A. van der Vorst, *Parallel incomplete factorizations with pseudo-overlapped subdomains*, Parallel Comput., 27 (2001), pp. 989–1008.
[12] M. Magolu monga Made and H. A. van der Vorst, *Spectral analysis of parallel incomplete factorizations with implicit pseudo-overlap*, Numer. Linear Algebra Appl., 9 (2002), pp. 45–64.
[13] Y. Saad, *ILUM: A multi-elimination ILU preconditioner for general sparse matrices*, SIAM J. Sci. Comput., 17 (1996), pp. 830–847.
[14] Y. Saad, *Iterative Methods for Sparse Linear Systems*, 2nd ed., SIAM, Philadelphia, 2003.
[15] Y. Saad, A. Soulaimani, and R. Touihri, *Variations on algebraic recursive multilevel solvers (ARMS) for the solution of CFD problems*, Appl. Numer. Math., 51 (2004), pp. 305–327.
[16] Y. Saad and B. Suchomel, *ARMS: An algebraic recursive multilevel solver for general sparse linear systems*, Numer. Linear Algebra Appl., 9 (2002), pp. 359–378.
[17] B. Smith, P. Bjørstad, and W. Gropp, *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations*, Cambridge University Press, Cambridge, UK, 1996.
[18] B. F. Smith, *Domain Decomposition Algorithms for the Partial Differential Equations of Linear Elasticity*, Technical report 517, Department of Computer Science, Courant Institute of Mathematical Sciences, New York University, New York, 1990.
[19] A. Soulaimani, N. B. Salah, and Y. Saad, *Enhanced GMRES acceleration techniques for some CFD problems*, Internat. J. CFD, 16 (2002), pp. 1–20.

SIAM J. SCI. COMPUT.
Vol. 28, No. 6, pp. 2294–2312
© 2006 Society for Industrial and Applied Mathematics

2-D PARACHUTE SIMULATION BY THE IMMERSED BOUNDARY METHOD*

YONGSAM KIM† AND CHARLES S. PESKIN‡

Abstract. Parachute aerodynamics involves an interaction between the flexible, elastic, porous parachute canopy and the high speed airflow (relative to the parachute) through which the parachute falls. Computer simulation of parachute dynamics typically simplifies the problem in various ways, e.g., by considering the parachute as a rigid bluff body. Here, we avoid such simplification by using the immersed boundary (IB) method to study the full fluid-structure interaction. The IB method is generalized to handle porous immersed boundaries, and the generalized method is used to study the influence of porosity on parachute stability.

Key words. parachute, canopy, payload, porosity, oncoming velocity, control mechanism, stability, immersed boundary method

AMS subject classifications. 65-04, 65M06, 76D05, 76M20

DOI. 10.1137/S1064827501389060

1. Introduction.

The purpose of this paper is to show that the immersed boundary (IB) method can be properly applied to the parachute problem and to demonstrate how the porosity of the parachute canopy affects the stability of parachute motion. The problem of parachute aerodynamics encompasses several of the most complex phenomena in classical fluid dynamics, such as porous bluff body aerodynamics and highly deformable structures. The cloth part of the parachute, which is called the canopy, acts as a bluff body resisting a moving flow and changes its shape rapidly in response to the surrounding flow field, but the airflow that generates the aerodynamic forces depends on the shape of the parachute canopy. Thus, parachute aerodynamics is inherently a fluid-structure interaction phenomenon, and to express this requires the time-dependent position of the parachute as well as the usual variables that appear in the Navier–Stokes equation [4, 13, 19, 21].

The IB method [22] was developed to study flow patterns around heart valves and is a generally useful method for problems in which elastic materials interact with a viscous incompressible fluid. In the IB formulation, the action of the elastic canopy immersed in the air flow appears as a localized body force acting on the fluid. This body force arises from the elastic stresses in the parachute canopy. Moreover, the parachute canopy is required to move at the local fluid velocity as a consequence of the no-slip condition. This condition is modified, however, in the case of canopy porosity, as described below. The central idea of the IB method is that the Navier–Stokes solver does not need to know anything about the complicated time-dependent geometry of the elastic boundary, and that therefore we can escape from the difficulties caused by the interaction between the elastic boundary and the fluid flow. This whole approach has been applied successfully to problems of blood flow in the heart [14, 16, 17, 18, 28],

*Received by the editors May 1, 2004; accepted for publication (in revised form) June 15, 2006; published electronically December 21, 2006. This work was supported by the National Science Foundation under KDI research grant DMS-9980069.

http://www.siam.org/journals/sisc/28-6/38906.html

†Department of Mathematics, University of Texas at Austin, Austin, TX 78712 (kimy@math.utexas.edu).

‡Courant Institute of Mathematical Sciences, New York University, 251 Mercer Street, New York, NY 10012 (peskin@cims.nyu.edu).

wave propagation in the cochlea [3, 6], platelet aggregation during blood clotting [8], and several other problems [1, 9, 10, 12, 29].

An interesting and important fact about the parachute problem is that it is a porous bluff-body motion. Because the purpose of a parachute is to provide drag, parachute fluid dynamics is irrevocably associated with the airflow around and through a bluff body. This bluff body naturally causes air flowing around the parachute to separate and induces vortex shedding [2]. But since the vorticity behind the bluff canopy shape usually is not symmetric and is too complicated to predict and control, it may affect canopy motion in an asymmetric way and induce an instability of the parachute motion [19]. The parachute canopy is generally equipped with a vent at its apex, which helps to control the inflation process and to stabilize the near-wake region. Even with an extremely small opening through the apex, the reduction of drag as well as the suppression of asymmetric wake oscillations are observed [11]. Besides its central vent, a typical parachute also has a lot of gaps between ribbons or ringslots on its canopy that effectively make it a porous body. Air flow allowed to go through this porous body collides with and then suppresses large-scale vortex motion behind the canopy and helps to stabilize parachute motion. For more details concerning parachute design and the porosity effect, see [4, 13, 21].

Besides demonstrating the application of the IB method to the parachute problem, the main purpose of this paper is to give porosity to the parachute and investigate its effect on parachute movement. The real parachute has many discrete holes in the canopy that make it porous such as a vent at the apex of the canopy and gaps between ribbons and ringslots. In the computation, however, it is impractical to resolve these holes individually. To do so would require that each hole be at least a few fluid meshwidths in diameter. Otherwise the hole would be effectively closed. Thus a very fine mesh would be required. Instead of modeling each hole individually, we give the canopy as a whole a porosity which depends neither on the meshwidth of the boundary nor on that of the fluid.

The idea of how to do this comes from the fact that porosity reduces the drag force of fluid. The drag force is the most significant aerodynamic characteristic of the parachute, but a drag force produced by a porous body is smaller than when a body has no porosity. In the usual application of the IB method, the immersed boundary moves at the local fluid velocity. This is the familiar no-slip condition of a viscous fluid. In the case of a porous immersed boundary, however, we have to allow relative slip between the boundary and the surrounding fluid. This slip is only in the normal direction; the tangential no-slip condition still holds. We assume that the normal relative velocity is determined by Darcy's law [20], i.e., that the normal relative velocity is proportional to the pressure difference across the boundary. Fortunately, there is no need to evaluate this pressure difference directly, since the IB method provides the force that the immersed boundary applies to the fluid, and the normal component of this force is proportional to the pressure difference across the boundary [25]. Thus we allow a relative slip between boundary and fluid, in the normal direction only, by an amount proportional to the normal component of the boundary force. The constant of proportionality is called the porosity.

2. Equations of motion. We begin by stating the mathematical formulation of the equations of motion for a system comprised of a two-dimensional (2-D) viscous incompressible fluid containing an immersed, elastic, massless, porous boundary:

$$\rho\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u}\right) = -\nabla p + \mu\nabla^2\mathbf{u} + \mathbf{f}, \tag{1}$$

$$\nabla \cdot \mathbf{u} = 0, \tag{2}$$

$$\mathbf{f}(\mathbf{x},t) = \int \mathbf{F}(s,t)\delta(\mathbf{x} - \mathbf{X}(s,t))ds, \tag{3}$$

$$\begin{aligned}\frac{\partial \mathbf{X}}{\partial t}(s,t) &= \mathbf{u}(\mathbf{X}(s,t),t) + \lambda(\mathbf{F}(s,t)\cdot\mathbf{n}(s,t))\mathbf{n}(s,t)\\ &= \int \mathbf{u}(\mathbf{x},t)\delta(\mathbf{x} - \mathbf{X}(s,t))d\mathbf{x} + \lambda(\mathbf{F}(s,t)\cdot\mathbf{n}(s,t))\mathbf{n}(s,t),\end{aligned} \tag{4}$$

$$\mathbf{F}(s,t) = \frac{\partial}{\partial s}(T(s,t)\boldsymbol{\tau}(s,t)), \tag{5}$$

$$T(s,t) = c\Big(\Big|\frac{\partial \mathbf{X}(s,t)}{\partial s}\Big| - 1\Big), \tag{6}$$

$$\boldsymbol{\tau}(s,t) = \frac{\partial \mathbf{X}(s,t)/\partial s}{|\partial \mathbf{X}(s,t)/\partial s|}, \tag{7}$$

$$\mathbf{n}(s,t) = \boldsymbol{\tau}(s,t) \times \mathbf{e}_3. \tag{8}$$

Equations (1) and (2) are the familiar Navier–Stokes equations for a viscous incompressible fluid. The constant parameters ρ and μ are the fluid density and viscosity, respectively. The unknown functions in the fluid equations are the fluid velocity, $\mathbf{u}(\mathbf{x},t)$; the fluid pressure, $p(\mathbf{x},t)$; and the force per unit area applied by the immersed boundary to the (2-D) fluid, $\mathbf{f}(\mathbf{x},t)$, where $\mathbf{x} = (x,y)$ are fixed Cartesian coordinates and t is the time.

Equations (5)–(8) are the IB equations which are written in Lagrangian form. The unknown $\mathbf{X}(s,t)$ completely describes the motion of the immersed boundary and also its spatial configuration at any given time. Other unknown functions of (s,t) that appear in the boundary equations are the unit tangent vector to the boundary, $\boldsymbol{\tau}(s,t)$; the tension, $T(s,t)$; force density at the boundary, $\mathbf{F}(s,t)$; and the unit normal to the boundary $\mathbf{n}(s,t) = \boldsymbol{\tau}(s,t) \times \mathbf{e}_3$, where $\mathbf{e}_3$ is a constant unit vector normal to the plane of the (2-D) flow. The equation for the elastic tension T follows Hooke's law for a simple spring which resists both stretching and contracting with a linear constitutive relation but does not resist bending. But it could be easily generalized to include bending resistance or a more complicated nonlinear constitutive law.

Finally, we come to the interaction equations (3) and (4). These both involve the 2-D Dirac delta function $\delta(\mathbf{x}) = \delta(x)\delta(y)$, which expresses the local character of the interaction. Equation (3) simply expresses the relation between the two corresponding force densities $\mathbf{f}(\mathbf{x},t)d\mathbf{x}$ and $\mathbf{F}(s,t)ds$. We can see this fact by integrating each side of (3) over an arbitrary region Ω. It should be noted, however, that, in (3), since $\delta(\mathbf{x})$ is the 2-D Dirac delta function, but integration is only over a one-dimensional (1-D) boundary contour, $\mathbf{f}(\mathbf{x},t)$ is a singular function like a 1-D delta function.

Equation (4) is the equation of motion of the immersed elastic boundary. It is explained as follows. First consider the special case $\lambda = 0$. Then (4) is the familiar no-slip condition. In the first form of (4), the expression $\mathbf{u}(\mathbf{X}(s,t),t)$ is the fluid

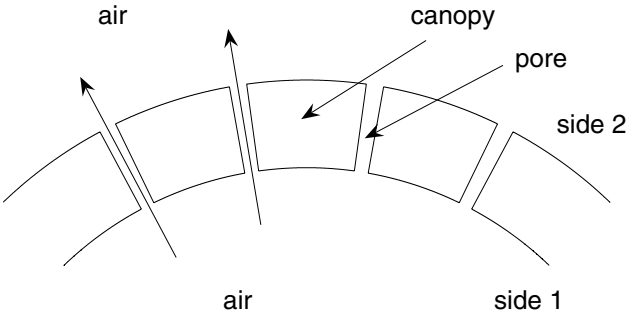


FIG. 1. *Porous boundary. The pores have small diameter in comparison to their length (the thickness of the boundary).*

velocity evaluated at the boundary. This is rewritten in terms of the Dirac delta function in the second form of (4). We do so in order to expose a certain symmetry with (3), in which the force generated by the immersed boundary is reexpressed as a body force acting on the fluid. This symmetry is important in the construction of our numerical scheme. In the following discussion of porosity, however, we shall simply use the notation $\mathbf{U}(s,t)$ for the fluid velocity evaluated at the boundary point $\mathbf{X}(s,t)$, i.e,

$$\mathbf{U}(s,t) = \mathbf{u}(\mathbf{X}(s,t),t) = \int \mathbf{u}(\mathbf{x},t)\delta(\mathbf{x}-\mathbf{X}(s,t))d\mathbf{x}. \tag{9}$$

Now consider the porosity of the immersed elastic boundary. (This discussion will be phrased in terms of the 2-D case, but of course a similar discussion could be made in three dimensions, with a few changes of units.) Consider the interval $(s, s+ds)$ of the immersed elastic boundary. Let β be the number density of pores, in the sense that βds is the number of pores in the interval $(s, s+ds)$. Let each pore have an aerodynamic conductance (reciprocal of resistance) equal to γ. This means that the flux through the pore is equal to $\gamma(p_1 - p_2)$, where p_1 and p_2 are the pressures on the two sides of the boundary; see Figure 1. Then the flux through the interval $(s, s+ds)$ of the boundary is given by $\beta\gamma(p_1 - p_2)ds$. This flux can be evaluated in another way by considering the difference between the fluid velocity at the boundary and the velocity of the boundary itself. The resulting expression for the flux is

$$\left(\mathbf{U}(s,t) - \frac{\partial \mathbf{X}}{\partial t}(s,t)\right) \cdot \mathbf{n} \left|\frac{\partial \mathbf{X}(s,t)}{\partial s}\right| ds, \tag{10}$$

where $\mathbf{n}$ is the unit normal to the boundary pointing from side 1 towards side 2. The factor $\left|\frac{\partial \mathbf{X}(s,t)}{\partial s}\right|$ appears because $\left|\frac{\partial \mathbf{X}(s,t)}{\partial s}\right| ds$ is the arc length of the segment $(s, s+ds)$. Setting these two expressions for the flux equal to each other, we get

$$\beta\gamma(p_1 - p_2)ds = \left(\mathbf{U}(s,t) - \frac{\partial \mathbf{X}}{\partial t}(s,t)\right) \cdot \mathbf{n} \left|\frac{\partial \mathbf{X}(s,t)}{\partial s}\right| ds. \tag{11}$$

But $(p_1 - p_2)$ can be related to the normal component of the boundary force $\mathbf{F}(s,t)$. The normal equilibrium of our massless boundary requires that

$$(p_1 - p_2)\left|\frac{\partial \mathbf{X}}{\partial s}(s,t)\right| ds + \mathbf{F}(s,t) \cdot \mathbf{n} ds = 0. \tag{12}$$

Combining these equations, we find

$$\left(\frac{\partial \mathbf{X}}{\partial t}(s,t) - \mathbf{U}(s,t)\right) \cdot \mathbf{n} = \frac{\beta\gamma}{|\partial \mathbf{X}(s,t)/\partial s|^2} \mathbf{F}(s,t) \cdot \mathbf{n}. \tag{13}$$

We also need a tangential component for the porous boundary condition. This is a complicated issue; see, for example, [3], where a slip boundary condition is derived at the interface of a fluid and a porous solid. In that paper, the solid has isotropic porosity. Here we assume that the parachute canopy is a thin shell of porous material *with pores oriented normal to the surface of the shell*. Moreover, we assume that the pores have small diameter in comparison to their length (the thickness of the shell). Under these conditions, the flow in each pore is normal to the surface of the canopy, so the tangential velocity of the flow in each pore is zero. Between the pores, we also have zero tangential velocity by the no-slip condition at a solid-fluid interface (see Figure 1). Combining these observations, it seems clear that the appropriate tangential boundary condition is the familiar no-slip condition, despite the porosity:

$$\left(\frac{\partial \mathbf{X}(s,t)}{\partial t} - \mathbf{U}(s,t)\right) \cdot \boldsymbol{\tau} = 0. \tag{14}$$

Then

$$\frac{\partial \mathbf{X}}{\partial t}(s,t) = \mathbf{U}(s,t) + \frac{\beta\gamma}{|\partial \mathbf{X}(s,t)/\partial s|^2} (\mathbf{F}(s,t) \cdot \mathbf{n})\mathbf{n}, \tag{15}$$

which is equivalent to (4), provided that we set

$$\lambda = \frac{\beta\gamma}{|\partial \mathbf{X}(s,t)/\partial s|^2}. \tag{16}$$

A question that still remains is whether β and γ depend on $|\partial \mathbf{X}(s,t)/\partial s|$. Recall that $|\partial \mathbf{X}(s,t)/\partial s|$ is the ratio of arc length to unstressed arc length, so it measures how stretched the material is. Intuitively, one would think that stretch would tend to increase either the number of pores or their conductance or both. Thus, one would expect $\beta\gamma$ to increase with $|\partial \mathbf{X}(s,t)/\partial s|$ but in a manner that would be hard to determine a priori. Here we make the simple assumption that λ is independent of $|\partial \mathbf{X}(s,t)/\partial s|$, i.e, that $\beta\gamma$ is proportional to $|\partial \mathbf{X}(s,t)/\partial s|^2$. More information about the material would be needed to refine this assumption.

3. Numerical method. We now describe a formally second-order IB method to solve the equations of motion [15, 18]. The word "formally" is used as a reminder that this scheme is only second-order accurate for problems with smooth solutions. Even though our solutions are not smooth (the velocity has jumps in derivative across the immersed boundary), the use of the formally second-order method results in improved accuracy; see [15].

The specific formally second-order method that we use is the one described in [18]. In this method, each time step proceeds in two substeps, which are called the preliminary and final substeps. In the preliminary substep, we get data at time level $n + \frac{1}{2}$ from data at n by a first-order accurate method. Then the final step starts

again at time level n and proceeds to time level $n+1$ by a second-order accurate method. This Runge–Kutta framework allows the second-order accuracy of the final substep to be the overall accuracy of the scheme.

We use a superscript to denote the time level. Thus $\mathbf{X}^n(s)$ is shorthand for $\mathbf{X}(s, n\Delta t)$, where Δt is the duration of the time step, and similarly for all other variables. Our goal is to compute updated $\mathbf{u}^{n+1}$ and $\mathbf{X}^{n+1}$ from given data $\mathbf{u}^n$ and $\mathbf{X}^n$.

Before describing how this is done, we have to say a few words about the spatial discretization. There are two such discretizations: one for the fluid and one for the elastic boundary. The grid on which the fluid variables are defined is a fixed uniform lattice of meshwidth $h = \Delta x_1 = \Delta x_2$. Now we define the central difference operator D_i, defined for $i = 1, 2$ as follows:

$$(D_i\phi)(\mathbf{x}) = \frac{\phi(\mathbf{x} + h\mathbf{e}^i) - \phi(\mathbf{x} - h\mathbf{e}^i)}{2h}, \tag{17}$$

where $\mathbf{e}^i$ is the unit vector in the ith coordinate direction. As the notation suggests, the difference operator in ith direction D_i corresponds to the ith component of the differential operator ∇. Thus $\mathbf{D}p$ will be the discrete gradient of p, and $\mathbf{D} \cdot \mathbf{u}$ will be the discrete divergence of $\mathbf{u}$.

We shall also make use the central difference Laplacian L.

$$(L\phi)(\mathbf{x}) = \sum_{i=1}^{2} \frac{\phi(\mathbf{x} + h\mathbf{e}^i) + \phi(\mathbf{x} - h\mathbf{e}^i) - 2\phi(\mathbf{x})}{h^2}. \tag{18}$$

The IB variables are defined as functions of s with meshwidth Δs. For any function $\phi(s)$, define D_s:

$$(D_s\phi)(s) = \frac{\phi(s + \Delta s/2) - \phi(s - \Delta s/2)}{\Delta s}. \tag{19}$$

The fluid mesh and the elastic boundary mesh defined above are connected by a smoothed approximation to the Dirac delta function. It is denoted δ_h and is of the following form:

$$\delta_h(\mathbf{x}) = h^{-2}\phi(\frac{x_1}{h})\phi(\frac{x_2}{h}), \tag{20}$$

where $\mathbf{x} = (x_1, x_2)$, and the function ϕ is given by

$$\phi(r) = \begin{cases} \frac{3-2|r|+\sqrt{1+4|r|-4r^2}}{8} & , \text{ if } |r| \le 1 \\ \frac{5-2|r|-\sqrt{-7+12|r|-4r^2}}{8} & , \text{ if } 1 \le |r| \le 2 \\ 0 & , \text{ if } 2 \le |r|. \end{cases} \tag{21}$$

The motivation and derivation for this particular choice is discussed in [23, 24].

We are now ready to describe a typical time step of the numerical scheme. The preliminary substep, which goes from time level n to $n + \frac{1}{2}$, proceeds as follows.

First, update the position of the massless boundary $\mathbf{X}^{n+\frac{1}{2}}(s)$:

$$\frac{\mathbf{X}^{n+\frac{1}{2}} - \mathbf{X}^n}{\Delta t/2} = \sum_{\mathbf{x}} \mathbf{u}^n(\mathbf{x})\delta_h(\mathbf{x} - \mathbf{X}^n(s))h^2 + \lambda(\mathbf{F}^n \cdot \mathbf{n}^n)\mathbf{n}^n, \tag{22}$$

where $\mathbf{n}^n$ is the unit normal to the boundary $\mathbf{X}^n$ and $\mathbf{F}^n$ can be calculated from $\mathbf{X}^n$ (see below). In general $\mathbf{n} = \boldsymbol{\tau} \times \mathbf{e}_3$ in (22), but $\boldsymbol{\tau}$ is not defined at each boundary point. To overcome this, we use

$$\boldsymbol{\tau}^n(s) = \frac{\mathbf{X}^n(s+\Delta s) - \mathbf{X}^n(s-\Delta s)}{|\mathbf{X}^n(s+\Delta s) - \mathbf{X}^n(s-\Delta s)|} \tag{23}$$

and then get $\mathbf{n}^n(s) = \mathbf{e}_3 \times \boldsymbol{\tau}^n(s)$.

Next, we calculate the force density $\mathbf{F}^{n+\frac{1}{2}}$ from the deformation of elastic boundary $\mathbf{X}^{n+\frac{1}{2}}$. The force density $\mathbf{F}^{n+\frac{1}{2}}$ can be obtained by the discretization of (5)–(7):

$$\boldsymbol{\tau}^{n+\frac{1}{2}} = \frac{D_s \mathbf{X}^{n+\frac{1}{2}}}{|(D_s \mathbf{X}^{n+\frac{1}{2}})|}, \quad T^{n+\frac{1}{2}} = c(|D_s \mathbf{X}^{n+\frac{1}{2}}| - 1), \quad \mathbf{F}^{n+\frac{1}{2}} = D_s(T^{n+\frac{1}{2}} \boldsymbol{\tau}^{n+\frac{1}{2}}). \tag{24}$$

Note that, since $\boldsymbol{\tau}^{n+\frac{1}{2}}$ and $T^{n+\frac{1}{2}}$ are defined at values of s halfway between those at which $\mathbf{X}^{n+\frac{1}{2}}$ is defined, $\mathbf{F}^{n+\frac{1}{2}}$ are defined at the same values of s as $\mathbf{X}^{n+\frac{1}{2}}$.

Now we have to change this elastic force defined on Lagrangian grid points into the force at Eulerian spatial grid points to be applied in the Navier–Stokes equations:

$$\mathbf{f}^{n+\frac{1}{2}}(\mathbf{x}) = \sum_s \mathbf{F}^{n+\frac{1}{2}}(s) \delta_h(\mathbf{x} - \mathbf{X}^{n+\frac{1}{2}}(s)) \Delta s, \tag{25}$$

where $\mathbf{x}$ is any point of the fluid mesh. With $\mathbf{f}^{n+\frac{1}{2}}$ in hand, we can turn to solving the Navier–Stokes equations

$$\rho \left(\frac{u_i^{n+\frac{1}{2}} - u_i^n}{\Delta t/2} + \frac{1}{2}(\mathbf{u} \cdot \mathbf{D} u_i + \mathbf{D} \cdot (\mathbf{u} u_i))^n \right) + D_i p^{n+\frac{1}{2}} = \mu L u_i^{n+\frac{1}{2}} + f_i^{n+\frac{1}{2}} \tag{26}$$

for $i = 1, 2$, together with

$$\mathbf{D} \cdot \mathbf{u}^{n+\frac{1}{2}} = 0. \tag{27}$$

Note that the unknowns in (26) and (27) are $u_i^{n+\frac{1}{2}}$ and $p^{n+\frac{1}{2}}$ and that they enter into these equations linearly. Since all the coefficients of these equations are constants, the system of equations (26)–(27) can be solved by the fast Fourier transform with the periodic boundary condition [23, 24].

The final step is the correction of $\mathbf{u}^{n+\frac{1}{2}}$ and $\mathbf{X}^{n+\frac{1}{2}}$ obtained in the preliminary step.

First, using the fluid velocity $\mathbf{u}^{n+\frac{1}{2}}$, we can find the boundary configuration $\mathbf{X}^{n+1}$:

$$\frac{\mathbf{X}^{n+1} - \mathbf{X}^n}{\Delta t} = \sum_{\mathbf{x}} \mathbf{u}^{n+\frac{1}{2}}(\mathbf{x}) \delta_h(\mathbf{x} - \mathbf{X}^{n+\frac{1}{2}}(s)) h^2 + \lambda(\mathbf{F}^{n+\frac{1}{2}} \cdot \mathbf{n}^{n+\frac{1}{2}}) \mathbf{n}^{n+\frac{1}{2}}, \tag{28}$$

where $\mathbf{n}^{n+\frac{1}{2}}$ is the unit normal to the boundary $\mathbf{X}^{n+\frac{1}{2}}$ and can be obtained in the same manner as $\mathbf{n}^n$ in the preliminary step.

The last thing that we have to do is to update the fluid velocity data by solving the following system of equations for the unknowns $\mathbf{u}^{n+1}$ and $p^{n+1/2}$:

$$\rho \left(\frac{u_i^{n+1} - u_i^n}{\Delta t} + \frac{1}{2}(\mathbf{u} \cdot \mathbf{D} u_i + \mathbf{D} \cdot (\mathbf{u} u_i))^{n+\frac{1}{2}} \right) + D_i p^{n+\frac{1}{2}} = \frac{1}{2} \mu L (u_i^{n+1} + u_i^n) + f_i^{n+\frac{1}{2}} \tag{29}$$

for $i = 1, 2$, together with

$$\mathbf{D} \cdot \mathbf{u}^{n+1} = 0. \tag{30}$$

Note that $p^{n+1/2}$ here is not the same as $p^{n+1/2}$ in (26). They are different approximations to the pressure at time level $n + 1/2$. The method of solving (29)–(30) is the same as that of solving (26)–(27), and indeed the same subroutines can be reused for this purpose. Since we have now computed $\mathbf{u}^{n+1}$ and $\mathbf{X}^{n+1}$, the time step is complete.

4. Two-dimensional parachute model. In this section we introduce a 2-D computational model of a parachute. We present the initial configuration of our model and display the physical and computational parameters which are used in the numerical experiments.

Consider the incompressible viscous fluid in a square box $(0, 4\text{m}) \times (0, 4\text{m})$ with periodic boundary conditions (but see below) which contains an immersed elastic parachute. Figure 2 shows the initial constructed configuration of a 2-D parachute and a part of the computational domain. The canopy has as its initial shape an upper half-ellipse centered at $(0, 0)$ which is expressed $\{(a\cos(s), b\sin(s)) : 0 \leq s \leq \pi\}$ as a parameterized curve. Linking the two end points of the canopy to the payload position completes the basic symmetric parachute which is then rotated about the

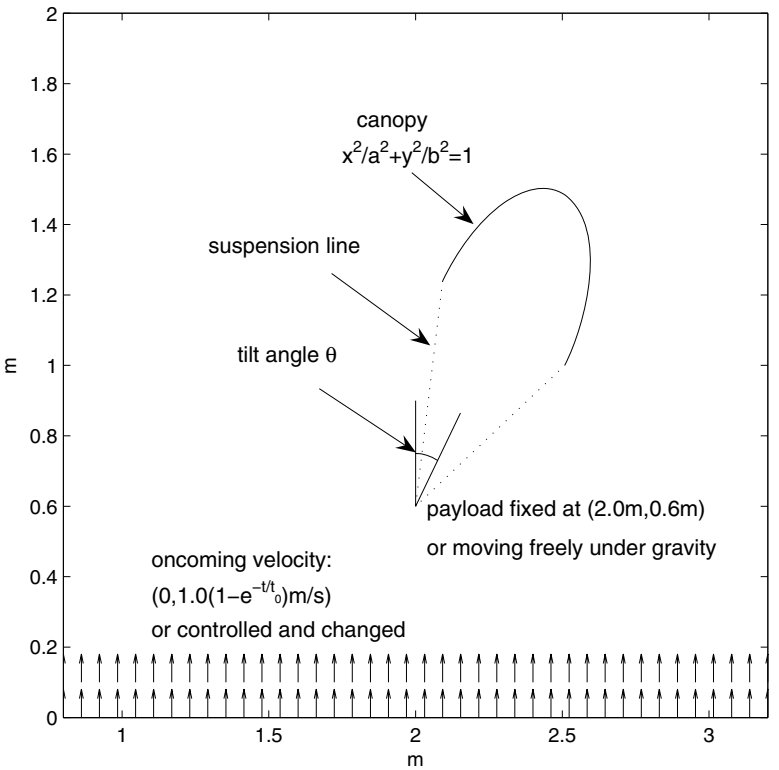


FIG. 2. *The initial configuration of the parachute. The tilt angle from the y-axis is the initial attack angle θ with respect to the oncoming velocity $\mathbf{u}_0 = (0, V_0(t))$. The payload is either fixed or moving freely. While the oncoming velocity is prescribed in the fixed-payload case, it changes determined by a control mechanism in the free-payload case.*

payload point by an arbitrary angle θ. The first simulation has its payload position fixed at the point (2.0m, 0.6m) which does not move and, in each time step, is just attached to two end points of the updated canopy position with two straight lines. These two lines, which are called suspension lines, are pure force generators. They do not interact with the fluid along their length, but simply act as linear springs which apply force to the two end points of the canopy.

The most natural way to model a parachute would be to let it fall, under the influence of gravity acting on its payload, through air which would be at rest at a large distance from the parachute. Here, however, we drive the air upwards and keep the parachute within the computational domain either by attaching the payload to a fixed point or by letting the payload fall freely through the rising air under the influence of gravity and adjusting the velocity of the rising air so that the parachute stays near a fixed level. In either case, we need a way to specify the "oncoming velocity" of the air that is coming up from below the parachute. This is done by applying an external force per unit volume equal to

$$\mathbf{f}_0(\mathbf{x},t) = \begin{cases} \alpha_0(\mathbf{u}_0(t) - \mathbf{u}(\mathbf{x},t)), & \mathbf{x} \in \Omega \\ 0 & \text{otherwise,} \end{cases} \tag{31}$$

where Ω is a rectangular strip along the bottom of the domain in which we want to specify the oncoming velocity. In practice, we choose Ω two meshwidths wide on the coarsest grid that we employ for these computations. Then, since Ω represents a fixed physical region, the number of horizontal meshlines within Ω must increase inversely with the meshwidth as the meshwidth is refined.

For the fixed payload case, for example, the oncoming velocity $\mathbf{u}_0(t)$ is given by $(0, 1.0(1 - \exp(-\frac{t}{t_0}))\text{m/s})$. Here, in order to avoid the abrupt change of velocity field and then parachute configuration, we set the oncoming velocity as a function of time which is initially $(0, 0)$ but increases gradually up to a constant value (0,1.0m/s).

Although this method of specifying the oncoming velocity is crude, it is not quite as crude as one might think. We address here two concerns that may occur to the reader. First, does the velocity field on Ω really match the oncoming velocity $\mathbf{u}_0(t)$? When we look closely at the velocity field in this region, we can see that the real velocity in this region quickly catch up with the oncoming velocity and becomes very close to it. Second, since the pressure is computed by solving a periodic problem, doesn't this result in unwanted interaction between the top and bottom of the domain? To investigate this, we have compared the pressures above and below the lines on which the oncoming velocity is specified. After an initial transient, these pressures are actually *anti*correlated, thus showing that the specified oncoming velocity has effectively broken the periodicity, as one would expect by considering, say, the effect of a thin porous plug in a circular pipe.

The second simulation is done with the same parachute as before but we remove the tethered point and allow the payload to move. The point payload of our model has no direct effect on the fluid such as vortex shedding, and its movement is independent of the local fluid velocity around the payload. But it has a point mass M and therefore feels a gravitational force $M\mathbf{g}$, where $\mathbf{g}$ is the downward acceleration $(0, -g)$. Since the payload is also loaded by the stresses of two suspension lines, let $T_i(t)(i = 1, 2)$ be the tension of each suspension line and $\boldsymbol{\tau}_i(t), (i = 1, 2)$ be the unit vectors pointing from the payload to the two end points of the canopy, respectively. We then have the

total force $\mathbf{F}_p(t)$ acting on the payload,

$$\mathbf{F}_p(t) = M\mathbf{g} + \sum_{i=1}^{2} T_i(t)\boldsymbol{\tau}_i(t). \tag{32}$$

Let the velocity of the payload be $\mathbf{U}_p(t)$ and the position be $\mathbf{X}_p(t)$; then we have the equations of motion:

$$\frac{d\mathbf{U}_p(t)}{dt} = \frac{\mathbf{F}_p(t)}{M}, \tag{33}$$

$$\frac{d\mathbf{X}_p(t)}{dt} = \mathbf{U}_p(t). \tag{34}$$

In the fixed-payload case, the oncoming velocity was arbitrary, but with a free payload if we specify the oncoming velocity arbitrarily we shall find that it is either too small, in which case the parachute will fall out the bottom of the domain, or too large, in which case the parachute will rise out the top of the computational domain. To keep the parachute within the domain, and away from the meshlines on which the oncoming velocity is specified, we use a control mechanism to adjust the oncoming velocity in such a manner that the y-coordinate of the payload settles to a predetermined value. The equation governing this control mechanism is as follows:

$$\frac{dV_0(t)}{dt} = k(y_{target} - Y_p(t)) - \sigma V_p(t). \tag{35}$$

In (35), $Y_p(t)$ and $V_p(t)$ are obtained by taking the y-components of $\mathbf{X}_p(t)$ and $\mathbf{U}_p(t)$ from (33) and (34), respectively. The velocity $(0, V_0(t)) = \mathbf{u}_0$ is the oncoming velocity at time t, y_{target} is the fixed value at which the y-coordinate of the payload is supposed to have its equilibrium, and k and σ are constant coefficients. The equation says that if, at some time, the height of the payload $Y_p(t)$ is lower than the target position of the payload y_{target}, the oncoming velocity increases, and if $Y_p(t)$ is greater than y_{target}, the oncoming velocity decreases. But the change of the oncoming velocity is damped according to $V_p(t)$ in order to avoid large oscillations of the oncoming velocity. The coefficients k and σ are chosen so that the y-coordinate of the payload is stable around the target position y_{target}. Note, however, that we allow the parachute to move out the side of the domain, in which case we should handle the data outside the domain by duplicating them into the domain in a periodic way.

The readers may wonder in the free-payload case, why we need an oncoming velocity at all. Why not use the periodicity to let the parachute fall out the bottom of the domain and reappear as it does so at the top? Aside from the interaction of the parachute with its own wake that would then occur, there is a more fundamental problem. Since the periodic domain contains only a finite mass of fluid to which a constant force (the weight of the payload) is applied, the total downward momentum of the system will increase linearly with time, and no terminal velocity will exist. We avoid this difficulty through the use of the oncoming velocity.

The overall performance of a parachute can be summarized by the relationship between the drag force it generates and the speed at which it is falling (relative to the air at a large distance from the parachute). The two types of computer experiments introduced above assess this relationship in different ways. When the payload is fixed in place and the oncoming velocity is arbitrary, then the speed of the parachute

TABLE 1
Physical parameters.

Physical parameters	Symbol	Magnitude	Unit
Density	ρ	1.2	kg/m^3
Viscosity	μ	0.002	$kg/(m \cdot s)$
Gravitational acceleration	g	9.8	m/s^2
Mass of payload	M	0.08	kg
Computational domain		4×4	$m \times m$
Canopy length		1.0561	m
Suspension line length		0.6462	m
Opening length	$2a$	$0.1 - 0.48$	m
Porosity	λ	$0.0 - 0.2$	$m^2/(N \cdot s)$
Initial tilt angle	θ	$0.0 - \pi/6$	

(relative to the distant air) is the independent variable, and the drag force is computed. Indeed, the drag force can be determined simply by examining the tensions and angles of the suspension lines. When the payload is free and the oncoming velocity is adjusted to keep the parachute from falling or rising, then the drag force is the independent variable, since it has to be equal to the specified weight of the payload. In this case, the speed corresponding to the given drag force is just the equilibrium value of the oncoming velocity, as set by the control mechanism. Although we would expect to get the same steady-state relationship between speed and drag force from either type of computational experiment, the dynamics of the two cases could certainly be different. This is because the free-payload case has two additional degrees of freedom and one additional parameter (the payload mass), which might well be expected to influence the dynamics.

Two important parameters of our computational experiments are the initial tilt angle θ (defined in Figure 2) and the porosity λ (defined in (16) and discussed above). An initial tilt angle is needed to break the left-right symmetry of the problem in order to explore the possibility that the symmetric configuration of the parachute may be unstable, and that the parachute may oscillate from side to side. If the symmetric configuration is linearly unstable, then any nonzero initial tilt angle will lead to such oscillations. But if it is only nonlinearly unstable, then there may be a threshold value of the initial tilt angle below which the parachute settles into a steady symmetrical configuration and above which it settles into a sustained side-to-side oscillation. The porosity λ is introduced in order to study its influence on the stability of the steady, symmetrical parachute configuration. To avoid numerical difficulties at the ends of the parachute canopy, we make λ a function of s that is constant at its maximum value near the center of the parachute canopy and then tapers smoothly to zero at the ends of the canopy. When we report a numerical value of λ, that refers to the maximum value.

Table 1 shows other physical parameters in addition to θ and λ. Our model of parachute is chosen to have a small dimension compared to that of real parachutes. The density is the same as that of air, but the dynamic viscosity is 0.002, which is 100 times bigger than that of air. The reason for the small size of parachute and the large viscosity is to reduce the Reynolds number. We believe that, with computational resolution affordable so far, we can properly compute up to a Reynolds number of several hundreds. With those values, we have the Reynolds number as large as 400 based on the oncoming velocity and the diameter of the canopy. The initial canopy

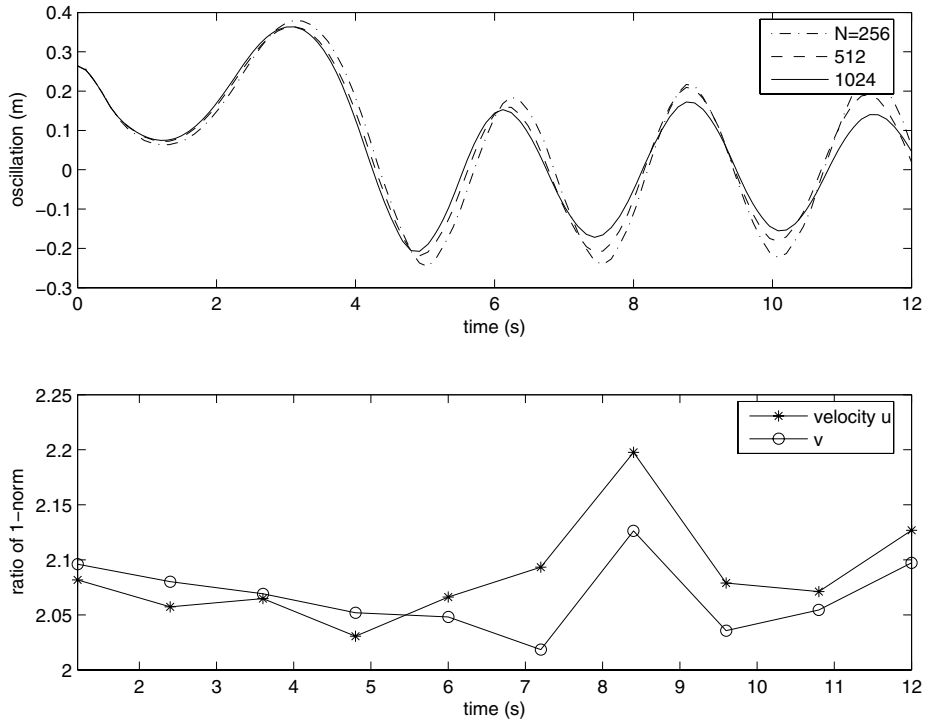


FIG. 3. *The top graph plots the difference between the x-coordinate of the midpoint of the canopy and the payload of the parachute (in meters) as a function of time (in seconds), which represents the side-to-side oscillations. The three curves show results obtained with three different resolutions: $N = 256$, 512, and 1024. The lower panel plots the convergence ratios of the velocity field (u, v) in the 1-norm. The convergence ratio 2 implies that the scheme is first-order accurate.*

configuration is a half-ellipse with semiaxes a and b. In our numerical experiments, the data that we vary are the porosity λ, the initial tilt angle θ, and the initial opening length $2a$. The ranges of these parameters are given in Table 1.

5. Results and discussion. We first verify that the IB method as applied to a porous boundary is robust and consistent. To do that, we choose a parachute with porosity coefficient $\lambda = 0.09$ m^2/(N·s), tilt angle $\theta = \pi/12$, and the initial opening length $2a = 0.48$m. The parachute is connected to a payload fixed at (2.0m, 0.6m) by two suspension lines. For $N = 256$, 512, and 1024, we change the time step $\Delta t = 0.0256/N$, the space meshwidth $h = 4/N$, and the boundary meshwidth $\Delta s = (128L_0)/(70N)$, where L_0 is the canopy length. That is, when we refine the meshwidths h and Δs by a factor of 2, the time step Δt is also reduced by the same factor.

The top of Figure 3 shows the x-coordinate of the midpoint of the parachute canopy relative to that of the fixed point as a function of time. This indicates the deviation of the parachute from its symmetrical falling configuration and shows whether the parachute is falling straight down or oscillating from side to side. The oscillations computed on the three different meshwidths track each other quite well through the four cycles of oscillation shown. To get a more quantitative measure of convergence, we compare the velocity fields computed on the three different meshwidths. The con-

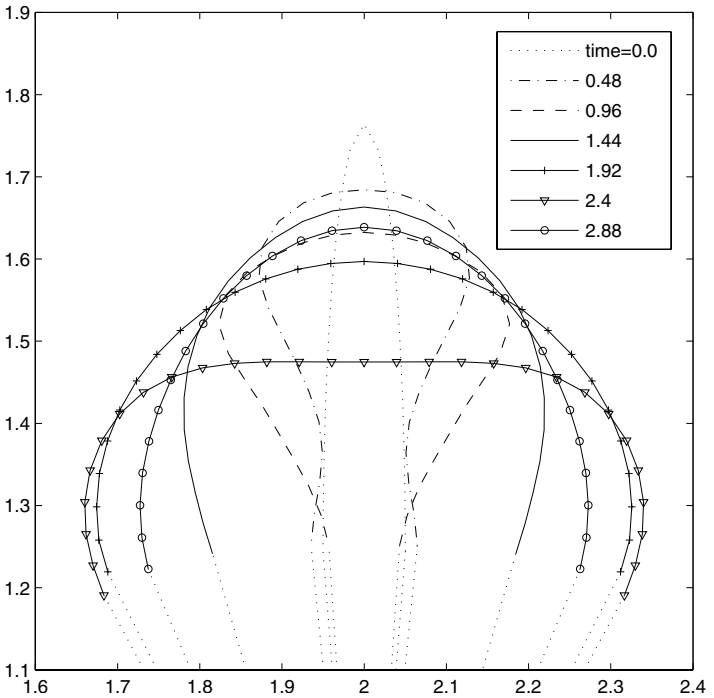


FIG. 4. *Inflation of parachute at early times. Note that inflation occurs first in the upper part of the canopy. In fact the suspension lines actually move slightly closer together at first but then move apart as inflation spreads to the entire canopy.*

vergence ratio is roughly equal to 2 in the L_1 norm. Let (u_N, v_N) be the velocity field, and let $||\cdot||_1$ be L_1 norm. Then the bottom of Figure 3 shows the convergence ratios $||u_{256} - u_{512}||_1/||u_{512} - u_{1024}||_1$ and $||v_{256} - v_{512}||_1/||v_{512} - v_{1024}||_1$ as functions of time. The convergence ratio 2 implies that the scheme has first-order accuracy which is typical of the IB method as applied to problems with thin elastic boundaries (for second-order convergence of the IB method in the case of an immersed elastic structure of finite thickness, see [7]). Throught the rest of this paper, we use $N = 512$, which is the intermediate resolution of the convergence study depicted in Figure 3.

The first simulation that we consider involves the process of parachute inflation, starting from a nearly closed configuration and studying the changes in shape of the parachute at early times; see Figure 4. Of all the results that we consider, it is this one that shows most clearly the need for a method that can handle the unknown changes in shape of the parachute canopy. In the IB method, this is done without any regridding, since the canopy is represented in the fluid dynamics computation by a force field defined on a uniform grid. For alternative approaches, see [5, 26, 27].

Figure 4 shows the inflation of a parachute which has the payload tethered at $(2.0\text{m}, 0.6\text{m})$ and has no initial tilt angle, and for which the porosity coefficient $\lambda = 0\ \text{m}^2/(N{\cdot}\text{s})$ and the initial opening length $2a = 0.1$ m. At the beginning of parachute inflation, the upper part of canopy is expanding more than the lower part. But air captured by the canopy finally applies enough pressure to inflate the parachute completely. After time proceeds beyond the last time shown in Figure 4, the parachute moves stably without further change in its configuration.

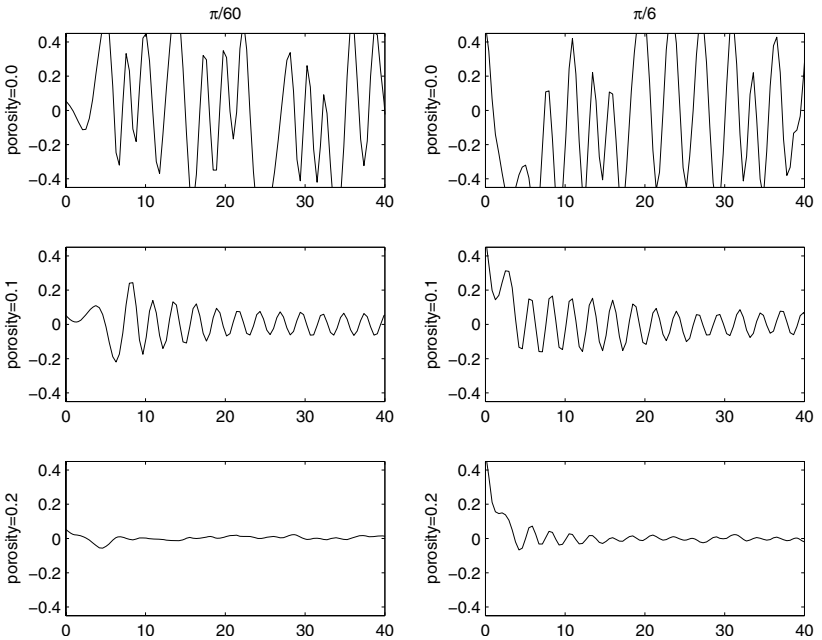


FIG. 5. *These graphs compare the stability of parachutes which have* $\pi/60$ *(left column) and* $\pi/6$ *(right column) as the initial tilt angle, and different coefficients* λ *of porosity equal to* 0 *(top row),* 0.1 $m^2/(N{\cdot}s)$ *(middle row), and* 0.2 $m^2/(N{\cdot}s)$ *(bottom row). Each graph plots the difference between the* x*-coordinates of the midpoint of the parachute canopy and the fixed payload (in meters) as a function of time (in seconds). Three qualitatively distinct behaviors are seen: unstable oscillations with large variable amplitude (top), oscillating steady state with damped amplitude (middle), or a rapid approach to the symmetrical steady state (upon which only small movements, possibly related to asymmetric vortex shedding, are superimposed).*

An important theme of this paper is the relationship between porosity and stability. The effect of porosity on stability is investigated in Figures 5 and 6. Each panel of Figure 5 is a graph showing the oscillation of the parachute in terms of the difference between the x-coordinates of the midpoint of the parachute canopy and the fixed position of the payload as a function of time. In the left-hand column, the initial tilt angle is $\theta = \pi/60$, which we regard as a small perturbation from symmetry about a line x=2. In the right-hand column, a larger initial tilt angle ($\theta = \pi/6$) is used. In each row of Figure 5, the parachute canopy has a different porosity, beginning with $\lambda = 0.0$ (no porosity) at the top and increasing to 0.1 in the middle and 0.2 $\mathrm{m}^2/(\mathrm{N{\cdot}s})$ at the bottom.

The top row shows that the oscillating steady state (limit cycle) is unstable for a parachute with no porosity, at least under the conditions of our simulation. For both initial conditions tried, the parachute does not settle into an oscillating steady state in which it rocks back and forth in a regular way. Instead its oscillatory motion looks somewhat chaotic. The parachute of the middle row has the intermediate porosity $\lambda = 0.1$ $\mathrm{m}^2/(\mathrm{N{\cdot}s})$. With this porosity, we can see the oscillation damped, and the parachute seems to be settling into the oscillating steady state (limit cycle). Finally, the bottom row suggests that the symmetrical steady state is globally stable for sufficiently high porosity, as almost no oscillations are seen in this case for either of the two initial conditions tried.

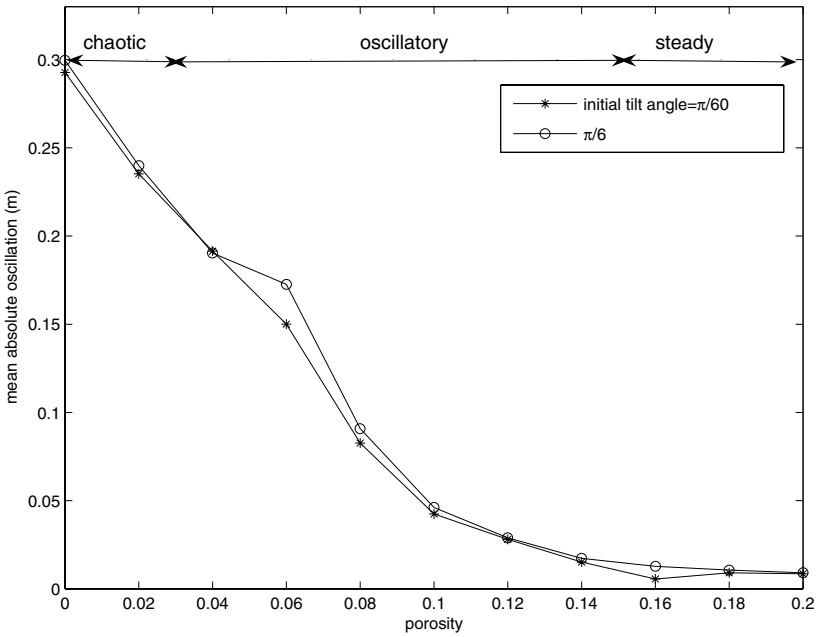


FIG. 6. *The mean absolute oscillation which is obtained by averaging the absolute oscillation over the time interval from* 25*s to* 42*s is plotted in terms of porosity. The mean oscillation of the parachute deceases almost linearly with the increasing porosity independent of the initial perturbation.*

We have found that the transition point from the chaotic motion to the oscillating steady state is when the porosity is around 0.025 $\mathrm{m}^2/(\mathrm{N}\cdot\mathrm{s})$. As the porosity increases beyond that point, the amplitude of the oscillation decreases almost linearly. The transition from the oscillating steady state to the symmetrical steady state happens at about a porosity of 0.15 $\mathrm{m}^2/(\mathrm{N}\cdot\mathrm{s})$. The amplitude of parachute motion with the same porosity is independent of the initial perturbation. The only difference is that the parachute with a larger initial tilt angle takes more time to settle down to its eventual chaotic, oscillatory, or steady state. Figure 6 shows the mean absolute oscillation for both cases. For each porosity, we plot the mean oscillation averaged over the time interval from 25s to 42s, thus avoiding the initial transient. Note that the results in Figure 6 are essentially the same for the two different initial conditions. The mean oscillation decreases linearly with porosity until a porosity of about 0.15 $\mathrm{m}^2/(\mathrm{N}\cdot\mathrm{s})$, above which there is essentially no oscillation and the parachute falls symmetrically in a steady manner.

The next result concerns the parachute with a moving payload. As discussed above, to prevent the parachute from leaving the computational domain, we use the oncoming velocity controlled by (35) with $k = 0.25\ \mathrm{s}^{-2}$ and $\sigma = 0.5\ \mathrm{s}^{-1}$. These values are chosen through numerical experiments to make the parachute with payload stay in the domain.

Figure 7 shows that, with these values of the parameters, the control mechanism of (35) behaves in a stable manner. In the upper graph, as time goes on, while the y-coordinate of the payload $Y_p(t)$ is converging to the target position $y_{target} = 1.2$ m for the parachute with porosity coefficient λ=0.1 $\mathrm{m}^2/(\mathrm{N}\cdot\mathrm{s})$, $Y_p(t)$ oscillates around the target position for the parachute without porosity. Similarly, the oncoming velocity

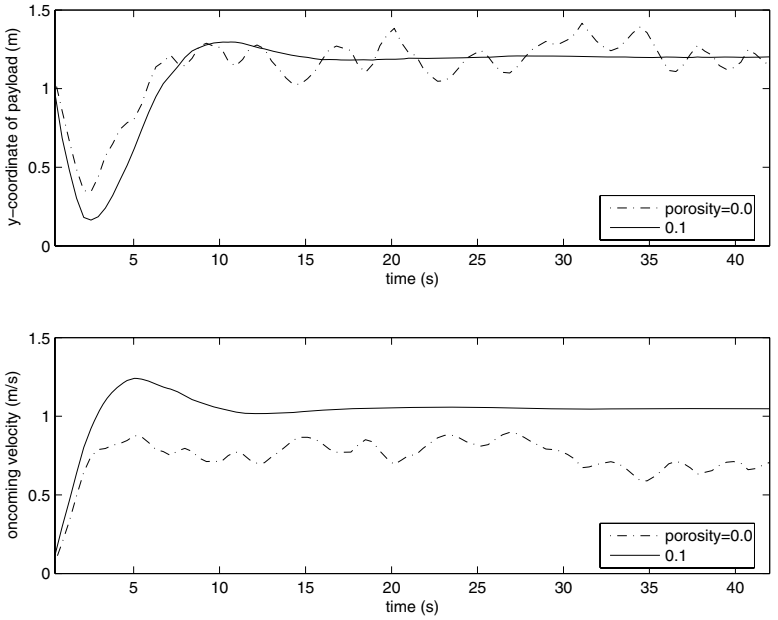


FIG. 7. *Behavior of the control mechanism that regulates the oncoming velocity in the free-payload case. Upper graph shows the y-coordinate of the payload $Y_p(t)$ as a function of time. $Y_p(t) \to 1.2m = y_{target}$ as $t \to \infty$ (with porosity) or $Y_p(t)$ fluctuates around y_{target} (without porosity). Lower graph shows the oncoming velocity $V_0(t)$ as a function of time. $V_0(0) = 0$ m/s, and $V_0(t) \to 1.05$ m/s as $t \to \infty$ for the parachute with porosity $\lambda = 0.1$ $m^2/(N{\cdot}s)$, but $V_0(t)$ changes around 0.76 m/s for the parachute with no porosity.*

$V_0(t)$ stays near one value after a short time with porosity 0.1 (lower graph), but the result in the case of no porosity goes through an unstable oscillation around another value. These fluctuations around the expected equilibrium values for the parachute with no porosity are related to its chaotic oscillatory motion; see below. Note from the lower graph that the parachute with porosity needs a larger oncoming velocity than the parachute without porosity to support the payload. Because porosity reduces the drag force of the fluid and causes the parachute with porosity to fall faster, a larger oncoming velocity is needed to generate a drag equal to the weight of the payload.

In this relatively stable situation, which we can regard as y-directional stability, we now investigate the relation between the porosity and the x-directional stability of the parachute. Figure 8 compares the x-directional stabilities of two parachutes which have the same initial tilt angle $\pi/12$, the initial opening length 0.48 m and two different porosities $\lambda = 0$ and 0.1 $\mathrm{m}^2/(\mathrm{N}{\cdot}\mathrm{s})$. From the top panel, which compares the x-coordinates of the payloads of the two parachutes, we can see that the parachute with porosity settles quickly to a fixed lateral position, but the parachute without porosity wanders considerably from side to side. The bottom panel shows the oscillation (difference between the x-coordinates of the midpoint of canopy and the payload) of the two parachutes. The parachute with porosity $\lambda = 0.1$ $\mathrm{m}^2/(\mathrm{N}{\cdot}\mathrm{s})$ moves more steadily than the parachute without porosity.

From Figures 5–8, we can see that porosity helps to stabilize the parachute against both side-to-side motion about the payload and x-directional motion as a whole (free payload only). As stated in the introduction, asymmetry of vortex shedding is the

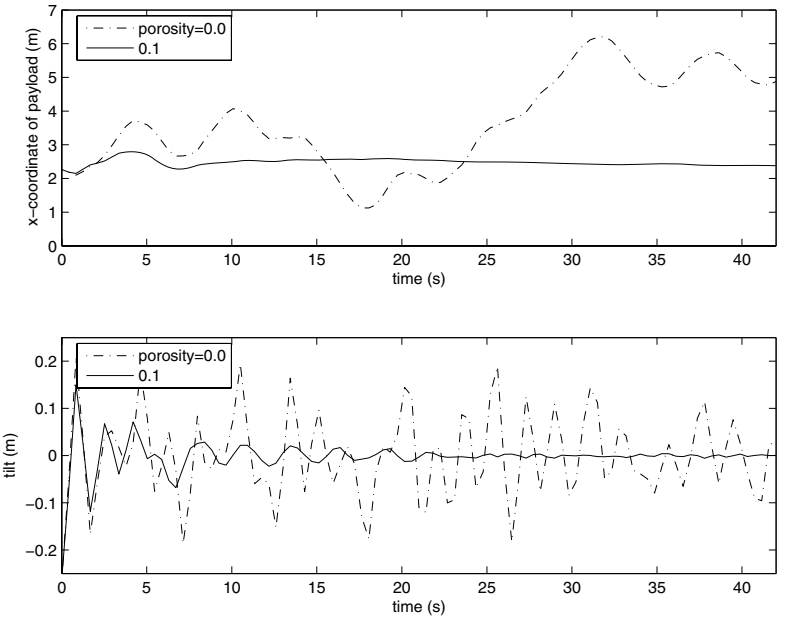


FIG. 8. *The graph compares the lateral stability of parachutes which have the same initial tilt angle $\pi/10$ but different coefficient λ of porosity. For each parachute, the x-coordinate of the payload is plotted in the upper panel, and the "tilt," i.e., the difference between the x-coordinate of the midpoint of the canopy and that of the payload, is plotted in the lower panel.*

probable cause of the parachute instability, but porosity can reduce the asymmetry of the vortex wake in the neighborhood of the canopy. To see that this idea is plausible, Figure 9 compares the vorticity contours of three cases of parachutes which have the same initial tilt angle $\theta = \pi/60$ but different porosity. The parachutes have fixed payloads. The parachute in the first column has no porosity, the one in the second column has porosity $\lambda = 0.1$ m^2/(N·s), and the one in the third column has $\lambda = 0.2$ m^2/(N·s). Each row represents a certain fixed time. At time 0, all parachutes have the same configuration as the initial configuration shown in Figure 2, and there is neither wind velocity nor vorticity. Figure 9 shows that the porous parachute settles into the oscillating steady state (second column) or symmetrical steady state (third column), but the parachute without porosity continues to oscillate in an unstable, chaotic way. We can also observe that the no-porosity case has very large and asymmetric vorticity. The parachute with porosity 0.1 shows a more regular version of alternate vortex shedding, and the parachute with porosity 0.2 has a nearly symmetrical distribution of vorticity in its wake.

6. Summary and conclusions. We have presented numerical experiments concerning the parachute problem in the 2-D case. Two basic configurations have been studied: one with a fixed payload in a prescribed updraft and the other with a free payload in a controlled updraft, the controller being designed to adjust the updraft so that the parachute stays within the computational domain. The coupled equations of motion of the air and the flexible parachute canopy have been solved by the IB method. We have used this methodology to simulate the details of parachute inflation and to study the influence of canopy porosity on the lateral stability of the parachute.

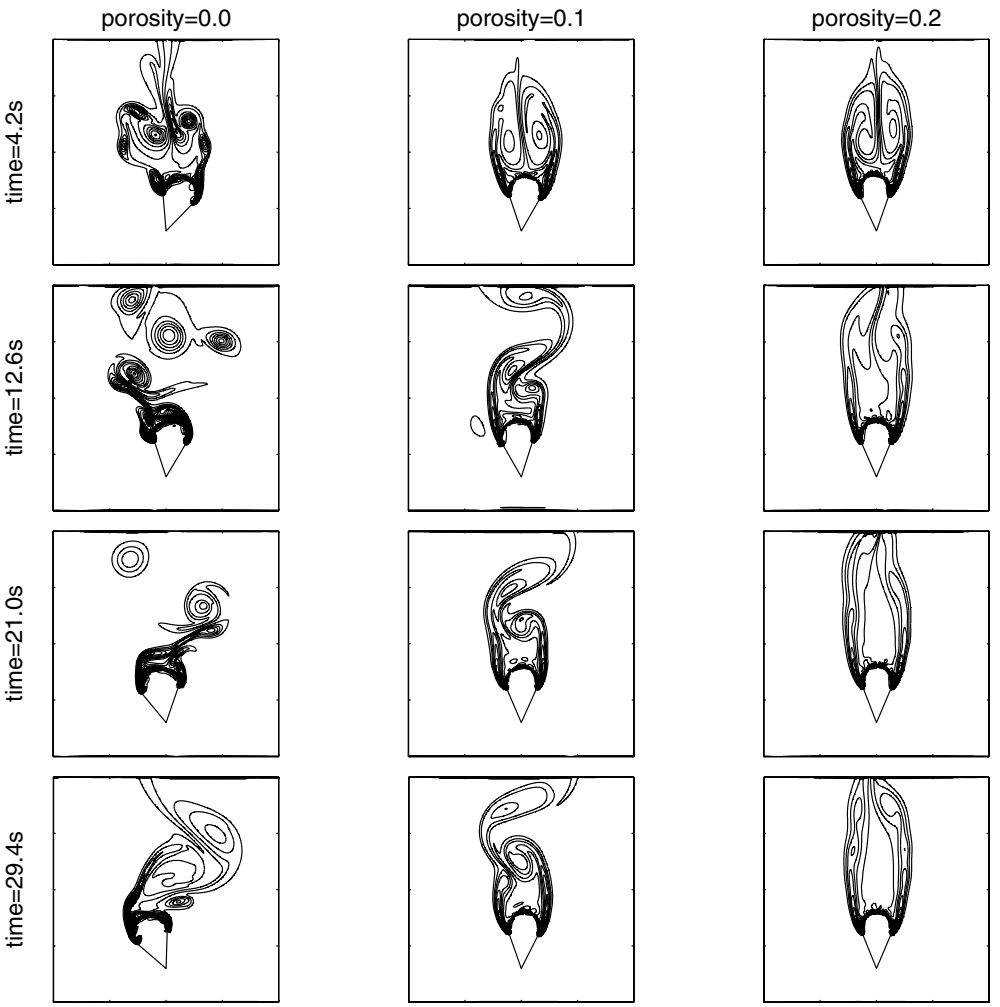


FIG. 9. *Vorticity contours in the wake of parachutes with fixed payloads. The parachute in the first column has zero porosity, the parachute in the second column porosity* $\lambda = 0.1$*, and the parachute in the third column* $\lambda = 0.2$ $m^2/(N{\cdot}s)$*. Each row shows a different time. A side-to-side oscillation and a very asymmetric vortex wake are observed in the case of the parachute with zero porosity. The parachute with porosity* 0.1 *still shows alternate vortex shedding but in a more regular manner, and the parachute with porosity* 0.2 *has an essentially symmetrical voticity distribution in its wake.*

Future work will include the generalization to the three-dimensional case, and studies concerning the influence of wind shear on parachute dynamics.

REFERENCES

[1] K. M. ARTHURS, L. C. MOORE, C. S. PESKIN, E. B. PITMAN, AND H. E. LAYTON, *Modeling arteriolar flow and mass transport using the immersed boundary method*, J. Comput. Phys., 147 (1998), pp. 402–440.

[2] P. W. BEARMAN AND M. TAKAMOTO, *Vortex shedding behind rings and disc*, Fluid Dynam. Res., 3 (1988), pp. 214–218.

[3] R. P. BEYER, *A computational model of the cochlea using the immersed boundary method*, J. Comput. Phys., 98 (1992), pp. 145–162.

[4] D. J. Cockrell, *The Aerodynamics of Parachutes*, AGARDograph 295, North Atlantic Treaty Organization, Advisory Group for Aerospace Research & Development, 1987.
[5] I. V. Dneprov, *Computation of aeroelastic characteristics and stress-strained state of parachutes*, in Proceedings of the 12th AIAA Aerodynamic Decelerator Systems Technology Conference and Seminar, 1993, AIAA Paper 93-1237.
[6] E. Givelberg, *Modeling Elastic Shells Immersed in Fluid,* Ph.D. thesis, Department of Mathematics, New York University, New York, 1997.
[7] B. E. Griffith and C. S. Peskin, *On the order of accuracy of the immersed boundary method: Higher order convergence rates for sufficiently smooth problems*, J. Comput. Phys., 208 (2005), pp. 75–105.
[8] A. L. Fogelson, *A mathematical model and numerical method for studying platelet adhesion and aggregation during blood clotting*, J. Comput. Phys., 56 (1984), pp. 111–134.
[9] A. L. Fogelson and C. S. Peskin, *A fast numerical method for solving the three-dimensional Stokes equations in the presence of suspended particles*, J. Comput. Phys., 79 (1988), pp. 50–69.
[10] L. J. Fauci and C. S. Peskin, *A computational model of aquatic animal locomotion*, J. Comput. Phys., 77 (1988), pp. 85–108.
[11] H. Higuchi and F. Takahashi, *Flow past two-dimensional ribbon parachute models*, J. Aircraft, 26 (1989), pp. 641–649.
[12] E. Jung and C. S. Peskin, *Two-dimensional simulations of valveless pumping using the immersed boundary method*, SIAM J. Sci. Comput., 23 (2001), pp. 19–45.
[13] T. W. Knacke, *Parachute Recovery Systems Design Manual*, NWC TP 6575, Naval Weapons Center, China Lake, CA, 1987.
[14] S. J. Kovacs, D. M. McQueen, and C. S. Peskin, *Modeling cardiac fluid dynamics and diastolic function*, Philos. Trans. R. Soc. Lond. Ser. A, 359 (2001), pp. 1299–1314.
[15] M. C. Lai and C. S. Peskin, *An immersed boundary method with formal second-order accuracy and reduced numerical viscosity*, J. Comput. Phys., 160 (2000), pp. 705–719.
[16] D. M. McQueen and C. S. Peskin, *Shared-memory parallel vector implementation of the immersed boundary method for the computation of blood flow in the beating mammalian heart*, J. Supercomputing, 11 (1997), pp. 213–236.
[17] D. M. McQueen and C. S. Peskin, *A three-dimensional computer model of the human heart for studying cardiac fluid dynamics*, Comput. Graphics, 34 (2000), pp. 56–60.
[18] D. M. McQueen and C. S. Peskin, *Heart simulation by an immersed boundary method with formal second-order accuracy and reduced numerical viscosity*, in Mechanics for a New Millennium, Proceedings of the International Conference on Theoretical and Applied Mechanics (ICTAM) 2000, H. Aref and J. W. Phillips, eds., Kluwer Academic Publishers, Norwell, MA, 2001.
[19] R. C. Maydew and C. W. Peterson, *Design and Testing of High-Performance Parachutes*, AGARD-AG-319, NTIS, Springfield, VA, 1991
[20] D. A. Nield and A. Bejan, *Convection in Porous Media*, Springer-Verlag, NY, 1991.
[21] C. W. Peterson, J. H. Strickland, and H. Higuchi, *The fluid dynamics of parachute inflation*, Annu. Rev. Fluid. Mech., 28 (1996), pp. 361–387.
[22] C. S. Peskin, *The immersed boundary method*, Acta Numer., 11 (2002), pp. 479–517.
[23] C. S. Peskin and D. M. McQueen, *Three dimensional computational method for flow in the heart: Immersed elastic fibers in a viscous incompressible fluid*, J. Comput. Phys., 81 (1989), pp. 372–405.
[24] C. S. Peskin and D. M. McQueen, *Fluid dynamics of the heart and its valves*, in Case Studies in Mathematical Modeling: Ecology, Physiology, and Cell Biology, Prentice–Hall, Englewood Cliffs, NJ, 1996, pp. 309–337.
[25] C. S. Peskin and B. F. Printz, *Improved volume conservation in the computation of flows with immersed elastic boundaries*, J. Comput. Phys., 105 (1993), pp. 33–46.
[26] J. Sahu, G. Cooper, and R. Benney, 3-*D parachute descent analysis using coupled CFD and structural codes*, in Proceedings of the 13th AIAA Aerodynamic Decelerator Systems Technology Conference, 1995, AIAA Paper 95-1580.
[27] K. R. Stein and R. J. Benney, *Parachute Inflation: A Problem in Aeroelasticity*, US Army Tech. Report NATICK/TR-94/015, Natick Research Development and Engineering Center, Natick, MA, 1994.
[28] E. J. Vigmond, D. M. McQueen, and C. S. Peskin, *The cable as a basis for a mechanoelectric whole heart model*, Internat. J. Bioelectromagnetism, 5 (2003), pp. 183–184.
[29] L. Zhu and C. S. Peskin, *Simulation of a flapping flexible filament in a flowing soap film by the immersed boundary method*, J. Comput. Phys., 179 (2002), pp. 452–468.

SIAM J. SCI. COMPUT.
Vol. 28, No. 6, pp. 2313–2336
© 2006 Society for Industrial and Applied Mathematics

DISCRETE MAXIMUM PRINCIPLE AND ADEQUATE DISCRETIZATIONS OF LINEAR PARABOLIC PROBLEMS*

ISTVÁN FARAGÓ† AND RÓBERT HORVÁTH‡

Abstract. In this paper, we analyze the connections between the different qualitative properties of numerical solutions of linear parabolic problems with Dirichlet-type boundary condition. First we formulate the qualitative properties for the differential equations and shed light on their relations. Then we show how the well-known discretization schemes can be written in the form of a one-step iterative process. We give necessary and sufficient conditions of the main qualitative properties of these iterations. We apply the results to the finite difference and Galerkin finite element solutions of linear parabolic problems. In our main result we show that the nonnegativity preservation property is equivalent to the maximum-minimum principle and they imply the maximum norm contractivity. In one, two, and three dimensions, we list sufficient a priori conditions that ensure the required qualitative properties. Finally, we demonstrate the above results on numerical examples.

Key words. parabolic problems, heat conduction, numerical solution, qualitative properties, discrete maximum principle, nonnegativity, contractivity in maximum norm

AMS subject classifications. 65M06, 65M50, 65M60, 65F10

DOI. 10.1137/050627241

1. Introduction.

Let us consider the partial differential equation $\mathcal{L}v = f$, where $\mathcal{L}$ denotes some linear partial differential operator, f is a given function, and v is the unknown function to be determined. A function g is also given, which prescribes the values of v on certain parts of the boundary of the solution domain.

Several problems for partial differential equations possess some characteristic qualitative properties, which are typical of the phenomenon the partial differential equation describes. The three most important of them are the *maximum-minimum principle* (MP), the *nonnegativity preservation* (NP), and the *maximum norm contractivity* (MNC). The MP states that the solution of a differential equation can be estimated from above and below by means of the given functions f and g. For example, when $f = 0$, the solution takes its extremal values on the boundary of the solution domain as well. The NP property means that the nonnegativity of f and g implies the nonnegativity of the solution v. The MNC property, which makes sense for time-dependent problems only, says that for two arbitrary initial functions the maximum norm of the difference of the solutions at every time level is not greater than the maximum norm of the difference of the initial functions.

In most cases, a partial differential equation cannot be solved in a useful form. In some particular cases, the solution can be written in the form of an infinite Fourier series, but this is also useless from the practical point of view. Thus, the use of numerical methods is a necessary step to obtain an approximate solution. It is a

*Received by the editors March 21, 2005; accepted for publication (in revised form) July 27, 2006; published electronically December 21, 2006.

http://www.siam.org/journals/sisc/28-6/62724.html

†Department of Applied Analysis, Eötvös Loránd University, Pf. 120, H-1518, Budapest, Hungary (faragois@cs.elte.hu). This author was supported by the National Scientific Research Fund (OTKA) N. T043765.

‡Corresponding author, Institute of Mathematics and Statistics, University of West-Hungary, Erzsébet u. 9, H-9400, Sopron, Hungary (rhorvath@ktk.nyme.hu). This author was supported by the National Scientific Research Fund (OTKA) N. T043765 and K61800.

natural requirement of an adequate numerical method that it has to possess the discrete equivalents of the qualitative properties the continuous problem satisfies.

The discrete version of the MP is the so-called *discrete maximum-minimum principle* (DMP). The topic of validity of various DMPs arose 30 years ago and was first investigated for elliptic problems (see, e.g., [4, 5, 18, 23]). The DMP was generally guaranteed by some geometrical conditions for the meshes. The DMP for parabolic problems was discussed in [8, 10, 11, 20]. In [11], based on the acuteness of the tetrahedral meshes, a sufficient condition of the DMP was obtained for the Galerkin finite element solution of certain parabolic problems. In the paper [8], a necessary and sufficient condition of the DMP was derived for Galerkin finite element methods and sufficient conditions were given for hybrid meshes. About the DMP, a comprehensive survey can be found in papers [2] and [3]; [17] investigates nonlinear problems.

The conditions of the *discrete nonnegativity preservation property* (DNP) were discussed in [7, 12] for linear finite elements in one, two, and three dimensions, and in [6] in one dimension with the combination of the finite difference and finite element methods. The DNP is investigated for nonlinear problems in [24].

The *discrete maximum norm contractivity property* (DMNC) was analyzed for one-dimensional (1D) parabolic problems in [13, 19]. In both references the necessary and sufficient conditions were given. In the first one, the dependence on the spatial discretization was also discussed.

For 1D problems, we can deduce some other remarkable qualitative properties such as the preservation of the shape and the monotonicity of the initial function, and the sign-stability (see, e.g., [9, 14, 15, 16, 22]).

Although some sufficient conditions (and in certain cases the necessary ones as well) are known for the above-listed three main qualitative properties, the relations and the implications between them have not been revealed yet. The main goal of this paper is to establish the connections between the MP, the NP, and the MNC both for linear parabolic problems and their numerical solutions. In this paper, we will show the implications

$$(D)NP \Longleftrightarrow (D)MP \Longrightarrow (D)MNC;$$

that is, the MP is equivalent to the NP property and both imply the MNC. We give necessary and sufficient conditions for the adequate (qualitatively correct) discrete models. Furthermore, we also give some useful sufficient conditions.

The paper is organized as follows. In section 2, we discuss the qualitative properties of continuous problems. In section 3, we give the linear algebraic form of the finite difference and finite element discretization of the investigated problem. We prove the existence and the uniqueness of the numerical solution. In section 4, we construct a one-step iterative process, we define its qualitative properties, and we formulate necessary and sufficient conditions of their preservation. In section 5, we apply the results of section 4 to the discretizations given in section 3. In section 6, some sufficient conditions are listed in order to guarantee the numerical qualitative properties a priori. In the last section, we demonstrate our results on numerical examples.

For simplicity, we denote zero matrices and zero vectors by the symbol $\mathbf{0}$, whose size is always chosen according to the context. The ordering relation for vectors and matrices is always meant elementwise.

2. Qualitative properties of linear parabolic problems. Let Ω and $\partial\Omega$ denote, respectively, a bounded domain in $\mathbb{R}^d$ ($d \in \mathbb{N}^+$) and its boundary and we

introduce the sets

$$Q_\tau = \Omega \times (0,\tau), \qquad Q_{\bar\tau} = \Omega \times (0,\tau], \qquad \Gamma_\tau = (\partial\Omega \times [0,\tau]) \cup (\Omega \times \{0\})$$

for any arbitrary positive number τ. For some fixed number $T > 0$, we consider the linear partial differential operator

$$\mathcal{L} \equiv \frac{\partial}{\partial t} - \sum_{0<|\varsigma|\le\delta} a_\varsigma \frac{\partial^{|\varsigma|}}{\partial^{\varsigma_1} x_1 \dots \partial^{\varsigma_d} x_d} \equiv \frac{\partial}{\partial t} - \sum_{0<|\varsigma|\le\delta} a_\varsigma D^\varsigma, \tag{2.1}$$

where $\delta \ge 1, \varsigma_1, \dots, \varsigma_d$ denote nonnegative integers, $|\varsigma|$ is defined as $|\varsigma| = \varsigma_1 + \dots + \varsigma_d$ for the multi-index $\varsigma = (\varsigma_1, \dots, \varsigma_d)$, and the coefficient functions $a_\varsigma : Q_T \to \mathbb{R}$ are bounded on the set Q_T. We define the domain of the operator $\mathcal{L}$, denoted by $\operatorname{dom}\mathcal{L}$, as the space of functions $v \in C(Q_T \cup \Gamma_T)$, for which the derivatives $D^\varsigma v$ $(0 < |\varsigma| \le \delta)$ and $\partial v/\partial t$ exist in Q_T and they are bounded. It can be seen easily that $\mathcal{L}v$ is bounded on $Q_{\bar t_1}$ for each $v \in \operatorname{dom}\mathcal{L}$ and $0 < t_1 < T$. Thus $\inf_{Q_{\bar t_1}} \mathcal{L}v$ and $\sup_{Q_{\bar t_1}} \mathcal{L}v$ are finite.

Definition 2.1. *We say that the operator $\mathcal{L}$ satisfies the MP if for any function $v \in \operatorname{dom}\mathcal{L}$ the inequality*

$$\min_{\Gamma_{t_1}} v + t_1 \cdot \min\{0, \inf_{Q_{\bar t_1}} \mathcal{L}v\} \le v(\mathbf{x}, t_1) \le \max_{\Gamma_{t_1}} v + t_1 \cdot \max\{0, \sup_{Q_{\bar t_1}} \mathcal{L}v\}$$

is satisfied for all $\mathbf{x} \in \Omega$, $0 < t_1 < T$.

The following statement shows that, in case of the MP, the functions from $\operatorname{dom}\mathcal{L}$ are uniquely determined by their values on the boundary Γ_T.

Theorem 2.2. *Let $t_1 \in (0,T)$ be any arbitrary fixed number. If $\mathcal{L}$ satisfies the MP, then for any two functions $v^\star, v^{\star\star} \in \operatorname{dom}\mathcal{L}$ the relations $\mathcal{L}v^\star|_{Q_{t_1}} = \mathcal{L}v^{\star\star}|_{Q_{t_1}}$ and $v^\star|_{\Gamma_{t_1}} = v^{\star\star}|_{\Gamma_{t_1}}$ imply $v^\star|_{Q_{t_1}} = v^{\star\star}|_{Q_{t_1}}$.*

Proof. Let us choose an arbitrary number satisfying the condition $0 < t_0 < t_1$, and we consider the function $\bar v = v^\star - v^{\star\star}$. For this function the relations $\mathcal{L}\bar v|_{Q_{\bar t_0}} = 0$ and $\bar v|_{\Gamma_{t_0}} = 0$ are valid. Thus, based on the MP we have $\bar v(\mathbf{x}, t_0) = 0$ for all $\mathbf{x} \in \Omega$ and $0 < t_0 < t_1$. This completes the proof. ∎

Definition 2.3. *The operator $\mathcal{L}$ is called nonnegativity preserving when the relations $\min_{\Gamma_{t_1}} v \ge 0$ and $\mathcal{L}v|_{Q_{\bar t_1}} \ge 0$ imply $v|_{Q_{\bar t_1}} \ge 0$ for all $0 < t_1 < T$.*

Definition 2.4. *The operator $\mathcal{L}$ is called contractive in maximum norm when for all arbitrary two functions $v^\star$, $v^{\star\star} \in \operatorname{dom}\mathcal{L}$ with $\mathcal{L}v^\star|_{Q_{\bar t_1}} = \mathcal{L}v^{\star\star}|_{Q_{\bar t_1}}$ and $v^\star|_{\partial\Omega\times[0,t_1]} = v^{\star\star}|_{\partial\Omega\times[0,t_1]}$ the relation*

$$\max_{\mathbf{x}\in\bar\Omega} |v^\star(\mathbf{x}, t_1) - v^{\star\star}(\mathbf{x}, t_1)| \le \max_{\mathbf{x}\in\bar\Omega} |v^\star(\mathbf{x}, 0) - v^{\star\star}(\mathbf{x}, 0)|$$

is valid for all $0 < t_1 < T$.

The main connections between the above properties of the operator $\mathcal{L}$ are listed in the next two theorems.

Theorem 2.5. *The operator $\mathcal{L}$ defined in* (2.1) *satisfies the MP if and only if it preserves the nonnegativity.*

Proof. The necessity of the condition is trivial. In order to show the sufficiency, we choose an arbitrary function $v \in \operatorname{dom}\mathcal{L}$ and apply the operator $\mathcal{L}$ to the function $\bar v = v - \min_{\Gamma_{t_1}} v - t \cdot \min\{0, \inf_{Q_{\bar t_1}} \mathcal{L}v\}$. Clearly, $\bar v|_{\Gamma_{t_1}} \ge 0$. Moreover, we obtain that

$$\mathcal{L}\bar v|_{Q_{\bar t_1}} = (\mathcal{L}v - \min\{0, \inf_{Q_{\bar t_1}} \mathcal{L}v\})|_{Q_{\bar t_1}} \ge 0,$$

which implies that $\bar{v}$ is nonnegative on $Q_{\bar{t}_1}$ by virtue of the NP assumption. Thus the lower estimation $\min_{\Gamma_{t_1}} v + t_1 \cdot \min\{0, \inf_{Q_{\bar{t}_1}} \mathcal{L}v\} \le v(\mathbf{x}, t_1)$ is satisfied.

By choosing

$$\bar{v} = \max_{\Gamma_{t_1}} v - v + t \cdot \max\{0, \sup_{Q_{\bar{t}_1}} \mathcal{L}v\},$$

the upper bound is proved similarly. This completes the proof. □

THEOREM 2.6. *If the operator $\mathcal{L}$ defined in* (2.1) *is nonnegativity preserving, then it is contractive in maximum norm.*

Proof. Let $v^\star$ and $v^{\star\star} \in \operatorname{dom}\mathcal{L}$ be two arbitrary functions with $\mathcal{L}v^\star|_{Q_{\bar{t}_1}} = \mathcal{L}v^{\star\star}|_{Q_{\bar{t}_1}}$ and $v^\star|_{\partial\Omega\times[0,t_1]} = v^{\star\star}|_{\partial\Omega\times[0,t_1]}$. We consider the functions $\bar{v}_\pm = \zeta \pm (v^\star - v^{\star\star})$ with $\zeta = \max_{\mathbf{x}\in\bar{\Omega}} |v^\star(\mathbf{x}, 0) - v^{\star\star}(\mathbf{x}, 0)|$. For these functions, the estimations $\mathcal{L}\bar{v}_\pm|_{Q_{\bar{t}_1}} = 0 \ge 0$ and $\min_{\Gamma_{t_1}} \bar{v}_\pm \ge 0$ are true, which implies the nonnegativity of $\bar{v}_\pm$ on $Q_{\bar{t}_1}$. Thus, we have

$$\max_{\mathbf{x}\in\bar{\Omega}} |v^\star(\mathbf{x}, t_1) - v^{\star\star}(\mathbf{x}, t_1)| \le \max_{\mathbf{x}\in\bar{\Omega}} |v^\star(\mathbf{x}, 0) - v^{\star\star}(\mathbf{x}, 0)|.$$

This completes the proof. □

In what follows, we consider the initial-boundary value problem in the form

$$\mathcal{L}v = f \quad \text{in } Q_T, \tag{2.2}$$

$$v|_{\Gamma_T} = g, \tag{2.3}$$

where $g : \Gamma_T \to \mathbb{R}$, $f : Q_T \to \mathbb{R}$ are arbitrary given functions. The operator $\mathcal{L}$ is defined in (2.1). (In the usual context, $g|_{\Omega\times\{0\}}$ is the initial and $g|_{\partial\Omega\times[0,T]}$ is the boundary condition, respectively.) We are interested in finding a function $v \in \operatorname{dom}\mathcal{L}$ that satisfies equalities (2.2)–(2.3). We say that the problem (2.2)–(2.3) is nonnegativity preserving and contractive in maximum norm and satisfies the MP if the operator $\mathcal{L}$ in (2.2) possesses the same properties. According to Theorems 2.5 and 2.6, the MP is valid for the problem (2.2)–(2.3) if and only if it is nonnegativity preserving; in this case the problem is contractive in maximum norm as well. If the MP is valid for the problem, then its solution (when it exists) is unique (cf. Theorem 2.2).

In this paper we investigate the parabolic problem

$$C\frac{\partial v}{\partial t} - \nabla(\kappa \cdot \nabla v) = f, \quad (\mathbf{x}, t) \in Q_T, \tag{2.4}$$

$$v|_{\Gamma_T} = g, \tag{2.5}$$

where $C : \Omega \to \mathbb{R}$ is a known bounded function with the property $0 < C_{\min} \le C \equiv C(\mathbf{x}) \le C_{\max}$, the known bounded function $\kappa : \Omega \to \mathbb{R}$ has continuous first derivatives and fulfills the property $0 < \kappa_{\min} \le \kappa \equiv \kappa(\mathbf{x}) \le \kappa_{\max}$, the function $g : \Gamma_T \to \mathbb{R}$ is continuous on Γ_T, the function $f : Q_T \to \mathbb{R}$ is bounded in Q_T, ∇ denotes the usual nabla operator, and the solution v is sought in $C^{2,1}(Q_T \cup \Gamma_T)$.

Remark 2.7. The problem (2.4)–(2.5) is generally applied for the description of heat conduction processes, where v denotes the temperature, C is the product of the specific heat and the density, κ is the thermal conductivity, and f gives the density of heat sources. The variables $\mathbf{x}$ and t play the role of the space and time variables, respectively. Problem (2.4)–(2.5) is suitable to describe diffusion and transport phenomena as well.

We show that the problem (2.4)–(2.5) is nonnegativity preserving and contractive in maximum norm and satisfies the MP. Because (2.4)–(2.5) can be written in the form of the problem (2.2)–(2.3) dividing both sides of the equation by the positive function C (that is, $\mathcal{L} \equiv \partial/\partial t - (1/C)\nabla(\kappa\nabla)$), we have to show only the validity of the NP (see Theorems 2.5 and 2.6). The proof will be based on the next theorem.

THEOREM 2.8. *For any $t_1 \in (0,T)$ and $\mathbf{x} \in \Omega$, the solution v of the problem* (2.4)–(2.5) *satisfies the inequality*

$$\sup_{\lambda>0}\left(e^{\lambda t_1}\min\left\{\min_{\Gamma_{t_1}} ve^{-\lambda t}, \frac{1}{\lambda}\inf_{Q_{\bar{t}_1}}\frac{fe^{-\lambda t}}{C}\right\}\right) \le v(\mathbf{x},t_1) \le \inf_{\lambda>0}\left(e^{\lambda t_1}\max\left\{\max_{\Gamma_{t_1}} ve^{-\lambda t}, \frac{1}{\lambda}\sup_{Q_{\bar{t}_1}}\frac{fe^{-\lambda t}}{C}\right\}\right), \tag{2.6}$$

where $Q_{\bar{t}_1} = Q_{t_1} \cup (\Omega \times \{t_1\})$.

Proof. For any arbitrary number $\lambda > 0$ we define the function $\hat{v}(\mathbf{x},t) = v(\mathbf{x},t)e^{-\lambda t}$, where v stands for the solution of (2.4). It can be seen easily that $\hat{v} \in C(Q_T \cup \Gamma_T)$, $\hat{v}|_{\Gamma_T} = (ve^{-\lambda t})|_{\Gamma_T}$, and $\hat{v}$ satisfies the equation

$$C\frac{\partial \hat{v}}{\partial t} - \nabla(\kappa\nabla\hat{v}) + C\lambda\hat{v} = fe^{-\lambda t} \tag{2.7}$$

in Q_T. As $\hat{v}$ is continuous on $\bar{Q}_{t_1}$, it takes its maximum either on the boundary Γ_{t_1} or in $\Omega \times (0,t_1]$ at some point $(\mathbf{x}^0,t^0)$. In the first case we trivially have

$$\sup_{Q_{t_1}} \hat{v} \le \max_{\Gamma_{t_1}} \hat{v}. \tag{2.8}$$

In the second case, the relations

$$\sup_{Q_{t_1}} \hat{v} \le \hat{v}(\mathbf{x}^0,t^0),\ \frac{\partial \hat{v}}{\partial t}(\mathbf{x}^0,t^0) \ge 0,\ \frac{\partial \hat{v}}{\partial x_i}(\mathbf{x}^0,t^0) = 0,\ \frac{\partial^2 \hat{v}}{\partial x_i^2}(\mathbf{x}^0,t^0) \le 0 \tag{2.9}$$

hold for $i = 1,\dots,d$, where the last two relations imply that $\nabla(\kappa\nabla\hat{v})(\mathbf{x}^0,t^0) \le 0$. This relation, combined with the second one in (2.9) and with (2.7), results in

$$0 \le f(\mathbf{x}^0,t^0)e^{-\lambda t^0} - C(\mathbf{x}^0)\hat{v}(\mathbf{x}^0,t^0)\lambda,$$

which can be rewritten in the form

$$\hat{v}(\mathbf{x}^0,t^0) \le \frac{1}{\lambda C(\mathbf{x}^0)} f(\mathbf{x}^0,t^0)e^{-\lambda t^0} \le \sup_{Q_{\bar{t}_1}} \frac{fe^{-\lambda t}}{C\lambda}. \tag{2.10}$$

Thus, in the general case, using the upper bounds (2.8) and (2.10) we obtain the estimation

$$\hat{v}(\mathbf{x},t_1) \le \sup_{Q_{t_1}} \hat{v} \le \max\left\{\max_{\Gamma_{t_1}} \hat{v}, \sup_{Q_{\bar{t}_1}} \frac{fe^{-\lambda t}}{C\lambda}\right\}.$$

Multiplying both sides by $e^{\lambda t_1}$ and taking into account that the relation is true for all positive numbers $\lambda > 0$, we obtain the inequality on the right-hand side of (2.6). The lower bound can be proved similarly. ∎

Remark 2.9. The proof of the above theorem is based on the proof of Theorem 2.1 in [20]. Let us notice, however, that we did not confine ourselves to second order operators, and leaving the zeroth order derivative out of the expression of $\mathcal{L}$, we arrived at a stronger estimation.

THEOREM 2.10. *The problem* (2.4)–(2.5) *preserves the nonnegativity, and consequently it is contractive in maximum norm and satisfies the MP.*

Proof. The proof is based on the previous theorem. Let $t_1 \in (0,T)$ be an arbitrary number, and let us suppose that $\mathcal{L}v|_{Q_{t_1}} \equiv (f/C)|_{Q_{t_1}} \geq 0$ and $v|_{\Gamma_{t_1}} \equiv g|_{\Gamma_{t_1}} \geq 0$. Then, for any $t_0 \in (0,t_1)$, we have $(f/C)|_{Q_{\bar{t}_0}} \geq 0$ and $v|_{\Gamma_{t_0}} \geq 0$, which result in $0 \leq v(\mathbf{x},t_0)$ in view of (2.6). That is v is nonnegative in Q_{t_1}. ∎

3. Numerical models of linear parabolic problems. The two most widely used numerical methods for solving the problem (2.4)–(2.5) are the finite difference and the Galerkin finite element methods. Finite difference methods are typically applied when the solution domain Ω is relatively simple (an interval, a rectangle, or a block), while finite element methods can be used also with more complicated geometrical structures (e.g., polyhedrons). Both methods start with the discretization of the computational space $\Omega \subset \mathbb{R}^d$ in order to get a system of ordinary differential equations. This is the so-called semidiscretization. Then the system of ordinary differential equations is solved by some time-integration method, such as the well-known θ-method. In this paper we consider only the three-dimensional (3D) case; the 1D and two-dimensional (2D) cases can be derived in the same manner by omitting the corresponding terms.

3.1. Spatial semidiscretization with the finite difference method. In the case of the finite difference method, the function v is approximated at the points of a rectangular mesh defined on the rectangular domain Ω. Let us denote the interior mesh points by $P_1,\dots,P_N \in \Omega$, and the points on the boundary by $P_{N+1},\dots,P_{N+N_\partial} \in \partial\Omega$, respectively. For the sake of simplicity, we also use the notation P_{i+x} (P_{i-x}) for the grid point adjoint to P_i in positive (negative) x-direction. We also define $\bar{N} = N + N_\partial$. After denoting the semidiscrete approximation of $v(\mathbf{x},t)$ at a grid point $\mathbf{x} = P_i$ by $v_i(t)$, we can approximate (2.4) at each inner point P_i of Ω as

$$C_i \frac{\mathrm{d}v_i(t)}{\mathrm{d}t} - (v_{ixx}(t) + v_{iyy}(t) + v_{izz}(t)) = f_i(t), \quad i = 1,\dots,N. \tag{3.1}$$

Here $f_i(t)$ denotes some approximation to $f(\mathbf{x},t)$ at the point P_i (the most typical choice is $f_i(t) = f(P_i,t)$); furthermore, $v_{ixx}(t)$ approximates the derivative

$$\frac{\partial}{\partial x}\left(\kappa \frac{\partial v}{\partial x}\right)$$

of the function v at the point P_i and at the time instant t, and has the form

$$v_{ixx}(t) = \frac{2}{h_{i+x} + h_{i-x}} \left(\kappa_{i+x/2} \frac{v_{i+x}(t) - v_i(t)}{h_{i+x}} - \kappa_{i-x/2} \frac{v_i(t) - v_{i-x}(t)}{h_{i-x}} \right). \tag{3.2}$$

The distances between the points P_{i+x}, P_i and P_{i-x}, P_i are denoted by h_{i+x} and h_{i-x}, respectively. The values $\kappa_{i-x/2}$ and $\kappa_{i+x/2}$ denote the approximate values of the material parameter κ on the segments $[P_{i-x}, P_i]$ and $[P_i, P_{i+x}]$ (typically the midpoint

values), and C_i denotes the approximate value of C at the point P_i (typically $C_i = C(P_i)$ for continuous functions). The terms $v_{iyy}(t)$ and $v_{izz}(t)$ are defined similarly.

Hence the semidiscretization (3.1) and the boundary condition (2.5) result in a Cauchy problem for the system of ordinary differential equations

$$\mathbf{M}\frac{\mathrm{d}\mathbf{v}(t)}{\mathrm{d}t} + \mathbf{K}\mathbf{v}(t) = \mathbf{f}(t), \tag{3.3}$$

$$\mathbf{v}(0) = [g(P_1,0),\dots,g(P_N,0),g(P_{N+1},0),\dots,g(P_{\bar N},0)]^\top,$$

where

$$\begin{aligned}\mathbf{v}(t) &\equiv [v_1(t),\dots,v_N(t),v_{N+1}(t),\dots,v_{\bar N}(t)]^\top\\ &= [v_1(t),\dots,v_N(t),g(P_{N+1},t),\dots,g(P_{\bar N},t)]^\top,\\ \mathbf{f}(t) &\equiv [f_1(t),\dots,f_N(t)]^\top\end{aligned} \tag{3.4}$$

and $\mathbf{M}$ and $\mathbf{K}$ are sparse $N \times \bar N$ matrices. For the entries of $\mathbf{M}$ we have

$$M_{ij} = \begin{cases} C_i & \text{if } i = j,\\ 0 & \text{if } i \neq j,\end{cases} \quad i = 1,\dots,N;\ j = 1,\dots,\bar N, \tag{3.5}$$

and the entries of $\mathbf{K}$ can be computed using (3.1) and (3.2). The nonzero elements of the ith row of $\mathbf{K}$ are $K_{i,i-x}, K_{i,i+x}, K_{i,i-y}, K_{i,i+y}, K_{i,i-z}, K_{i,i+z}, K_{i,i}$, where

$$K_{i,i-x} = \frac{-2\kappa_{i-x/2}}{h_{i-x}(h_{i-x}+h_{i+x})}, \quad K_{i,i+x} = \frac{-2\kappa_{i+x/2}}{h_{i+x}(h_{i-x}+h_{i+x})}, \tag{3.6}$$

$$K^x_{i,i} = \frac{2\kappa_{i+x/2}}{h_{i+x}(h_{i-x}+h_{i+x})} + \frac{2\kappa_{i-x/2}}{h_{i-x}(h_{i-x}+h_{i+x})},$$

$K_{i,i-y}, K_{i,i+y}, K_{i,i-z}, K_{i,i+z}, K^y_{i,i}, K^z_{i,i}$ are defined similarly, and

$$K_{i,i} = K^x_{i,i} + K^y_{i,i} + K^z_{i,i}. \tag{3.7}$$

3.2. Semidiscretization with the Galerkin finite element method. In the case of the Galerkin finite element method we cover the domain Ω with the following meshes $\mathcal{T}_h$ (h is a discretization parameter): the 1D mesh consists of intervals, the 2D mesh is a so-called hybrid mesh, which is a combination of triangles and rectangles, and the 3D mesh is a tetrahedron or a block mesh. Figure 3.1 shows a 2D hybrid mesh. As before, $P_1,\dots,P_N$ denote the interior nodes of $\mathcal{T}_h$, and $P_{N+1},\dots,P_{\bar N}$ the boundary ones. Let $\phi_1,\dots,\phi_{\bar N}$ be basis functions defined as follows: each ϕ_i is required to be continuous, piecewise linear (over intervals, triangular, or tetrahedral elements) or multilinear (over rectangular or block elements), such that $\phi_i(P_j) = \delta_{ij}$, $i,j = 1,\dots,\bar N$, where δ_{ij} is Kronecker's symbol. (Multilinearity means that it is bilinear in case of rectangles and trilinear in case of blocks.) It is obvious that such basis functions have the properties

$$\phi_i \geq 0,\ i = 1,\dots,\bar N,$$

$$\sum_{i=1}^{\bar N} \phi_i \equiv 1 \ \text{ in } \bar\Omega.$$

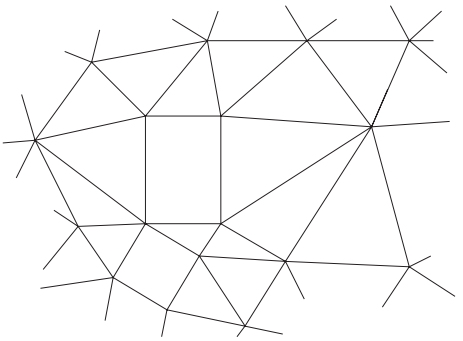


FIG. 3.1. *Hybrid mesh in two dimensions.*

We search for the semidiscrete solution of (2.4)–(2.5) in the form

$$\sum_{i=1}^{N} v_i(t)\phi_i(\mathbf{x}) + \sum_{i=1}^{N_\partial} g(P_{N+i}, t)\phi_{N+i}(\mathbf{x}).$$

Using the weak formulation of the problem, we arrive at a Cauchy problem again that has the form (3.1) with

$$\mathbf{M} = [M_{ij}]_{N\times\bar{N}}, \quad M_{ij} = \int_\Omega C(\mathbf{x})\phi_j(\mathbf{x})\phi_i(\mathbf{x})\,\mathrm{d}\mathbf{x},$$

$$\mathbf{K} = [K_{ij}]_{N\times\bar{N}}, \quad K_{ij} = \int_\Omega \kappa(\mathbf{x})\ \mathrm{grad}\phi_j(\mathbf{x})\ \mathrm{grad}\phi_i(\mathbf{x})\ \mathrm{d}\mathbf{x},$$

$$\mathbf{f}(t) = [f_i(t)]_{N\times 1}, \quad f_i(t) = \int_\Omega f(\mathbf{x}, t)\phi_i(\mathbf{x})\,\mathrm{d}\mathbf{x}$$

(see [8]). The above-defined matrices $\mathbf{M}$ and $\mathbf{K}$ are called *mass and stiffness matrices*, respectively.

3.3. Fully discretized model. To get a fully discrete numerical scheme, we choose a time-step $\Delta t > 0$. We denote the approximation to $\mathbf{v}(n\Delta t)$ by $\mathbf{v}^n$, and we set $\mathbf{f}^n = \mathbf{f}(n\Delta t)$ $(n = 0, 1, \ldots)$. The time-discretization of (3.1) with the θ-method ($\theta \in [0, 1]$ is a given parameter) can be written in the form of the systems of linear algebraic equations

$$\mathbf{M}\frac{\mathbf{v}^{n+1} - \mathbf{v}^n}{\Delta t} + \theta\mathbf{K}\mathbf{v}^{n+1} + (1-\theta)\mathbf{K}\mathbf{v}^n = \mathbf{f}^{(n,\theta)} := \theta\mathbf{f}^{n+1} + (1-\theta)\mathbf{f}^n, \tag{3.8}$$

where $n = 0, 1, \ldots$ and $\theta \in [0, 1]$ is a given parameter. Clearly, (3.8) can be rewritten as

$$(\mathbf{M} + \theta\Delta t\mathbf{K})\mathbf{v}^{n+1} = (\mathbf{M} - (1-\theta)\Delta t\mathbf{K})\mathbf{v}^n + \Delta t\ \mathbf{f}^{(n,\theta)}. \tag{3.9}$$

In what follows, the matrices $\mathbf{M} + \theta\Delta t\mathbf{K}$ and $\mathbf{M} - (1-\theta)\Delta t\mathbf{K}$ will be denoted by $\mathbf{A}$ and $\mathbf{B}$, respectively. We shall use the following partitions of the matrices and vectors:

$$\mathbf{A} = [\mathbf{A}_0|\mathbf{A}_\partial], \quad \mathbf{B} = [\mathbf{B}_0|\mathbf{B}_\partial], \quad \mathbf{v}^n = \begin{bmatrix} \mathbf{u}^n \\ \mathbf{g}^n \end{bmatrix},$$

where $\mathbf{A}_0$ and $\mathbf{B}_0$ are square matrices from $\mathbb{R}^{N\times N}$, $\mathbf{A}_\partial, \mathbf{B}_\partial \in \mathbb{R}^{N\times N_\partial}$, $\mathbf{u}^n \equiv [u_1^n, \dots, u_N^n]^\top \in \mathbb{R}^N$, and $\mathbf{g}^n \equiv [g_1^n, \dots, g_{N_\partial}^n]^\top \in \mathbb{R}^{N_\partial}$. Similar partition is used for the matrices $\mathbf{M}$ and $\mathbf{K}$. Then, iteration (3.9) can be written as

$$[\mathbf{A}_0|\mathbf{A}_\partial] \begin{bmatrix} \mathbf{u}^{n+1} \\ \mathbf{g}^{n+1} \end{bmatrix} = [\mathbf{B}_0|\mathbf{B}_\partial] \begin{bmatrix} \mathbf{u}^{n} \\ \mathbf{g}^{n} \end{bmatrix} + \Delta t\, \mathbf{f}^{(n,\theta)}. \tag{3.10}$$

The relation (3.10) defines a one-step iterative process with respect to the unknown vector $\mathbf{u}^{n+1}$.

Remark 3.1. For the finite difference method, the choice $\theta = 0$ results in an explicit scheme, due to the diagonality of $\mathbf{M}$. This is the so-called explicit Euler method. The choice $\theta = 1$ gives the so-called implicit Euler method and $\theta = 1/2$ is the Crank–Nicolson method. However, the schemes obtained by the Galerkin finite element method are always implicit for any choice of θ.

In order to guarantee the existence and the uniqueness of $\mathbf{u}^{n+1}$ in (3.10), we have to show that $\mathbf{A}_0$ is a nonsingular matrix.

THEOREM 3.2. *The matrix $\mathbf{A}_0 \in \mathbb{R}^{N\times N}$ is nonsingular for both the finite difference and the Galerkin finite element methods.*

Proof. The case of the finite difference method: Because of the relation $\mathbf{A}_0 = \mathbf{M}_0 + \theta\Delta t\mathbf{K}_0$ and the equalities in (3.6), the off-diagonal elements of $\mathbf{A}_0$ are nonpositive and the diagonal ones are positive. Moreover, considering (3.6) and (3.7) we have

$$K_{i1} + \cdots + K_{i\bar{N}} = 0 \quad (i = 1, \dots, N).$$

Thus, the estimate

$$\begin{aligned}(\mathbf{A}_0)_{i1} + \cdots + (\mathbf{A}_0)_{iN} &= C_i + \theta\Delta t(K_{i1} + \cdots + K_{iN}) \\ &\geq C_i + \theta\Delta t(K_{i1} + \cdots + K_{i\bar{N}}) = C_i \geq C_{\min} > 0\end{aligned}$$

is true $(i = 1, \dots, N)$, which shows that $\mathbf{A}_0$ is a strictly diagonally dominant matrix; hence it is nonsingular.

The case of the Galerkin finite element method: The elements of $\mathbf{A}_0$ are

$$(\mathbf{A}_0)_{ij} = \int_\Omega (C(\mathbf{x})\phi_j(\mathbf{x})\phi_i(\mathbf{x}) + \theta\Delta t\, \kappa(\mathbf{x})\, \mathrm{grad}\phi_j(\mathbf{x})\, \mathrm{grad}\phi_i(\mathbf{x}))\, \mathrm{d}\mathbf{x} \quad (i, j = 1, \dots, N).$$

The right-hand side defines a scalar product on the vector space $\mathrm{span}\{\phi_1, \dots, \phi_N\}$. Thus $\mathbf{A}_0$ is a Gram-matrix of the linearly independent functions $\phi_1, \dots, \phi_N$; hence it is nonsingular and also positive definite. ∎

Remark 3.3. It is important to notice that in the case of the finite difference method, $\mathbf{A}_0$ is a so-called M-matrix (see [1, p. 137]). This can be seen from the sign-structure and the strict diagonal dominance of $\mathbf{A}_0$. That is, in addition to the nonsingularity, $\mathbf{A}_0$ has a nonnegative inverse too. In the finite element method the nonnegativity of $\mathbf{A}_0^{-1}$ is not satisfied automatically.

4. Qualitative properties of one-step iterative models. As we mentioned in the introduction, it is a natural requirement for the numerical solution that it has to possess some basic qualitative properties. The numerical solution can be obtained by the iteration (3.10). Hence, the qualitative properties of the numerical solution will be defined as the qualitative properties of such iteration processes.

We use the denotations

$$\mathbf{e} = \mathbf{e}^{(\bar{N})},\ \mathbf{e}_0 = \mathbf{e}^{(N)},\ \mathbf{e}_\partial = \mathbf{e}^{(N_\partial)} \text{ with } \mathbf{e}^{(k)} = [1, \dots, 1]^\top \in \mathbb{R}^k.$$

4.1. Partitioned one-step (PO-) iterations. Based on the iteration (3.10), we will investigate the general one-step iteration

$$[\mathbf{A}_0|\mathbf{A}_\partial]\begin{bmatrix}\mathbf{u}^{n+1}\\ \mathbf{g}^{n+1}\end{bmatrix} = [\mathbf{B}_0|\mathbf{B}_\partial]\begin{bmatrix}\mathbf{u}^{n}\\ \mathbf{g}^{n}\end{bmatrix} + \mathbf{h}^n,\ n=0,1,\dots, \tag{4.1}$$

or equivalently

$$\mathbf{A}\mathbf{v}^{n+1} = \mathbf{B}\mathbf{v}^n + \mathbf{h}^n,\ n=0,1,\dots. \tag{4.2}$$

Here $\mathbf{h}^n \equiv [h_1^n,\dots,h_N^n]^\top \in \mathbb{R}^N$ and all matrices and vectors have the same dimension as in the previous section. We emphasize that now they are assumed to be *arbitrary* with the only restriction being that the matrix $\mathbf{A}_0$ is nonsingular. Iteration (4.2) is called a *partitioned one-step iteration* or shortly PO-iteration.

4.2. Qualitative properties of PO-iterations. Comparing the form

$$\mathbf{u}^{n+1}-\mathbf{u}^n = (\mathbf{A}_0^{-1}\mathbf{B}_0-\mathbf{I})\mathbf{u}^n - \mathbf{A}_0^{-1}\mathbf{A}_\partial\mathbf{g}^{n+1} + \mathbf{A}_0^{-1}\mathbf{B}_\partial\mathbf{g}^n + \mathbf{A}_0^{-1}\mathbf{h}^n$$

of (4.1) with (2.4), we define the equivalents of the qualitative properties for the PO-iterations.

DEFINITION 4.1. *A PO-iteration is said to satisfy the discrete nonnegativity preservation property (DNP) if the inequality* $\mathbf{u}^{n+1} \ge \mathbf{0}$ *is valid for all nonnegative vectors* $\mathbf{u}^n, \mathbf{g}^n, \mathbf{g}^{n+1}$, *and* $\mathbf{A}_0^{-1}\mathbf{h}^n$.

THEOREM 4.2. *A PO-iteration satisfies the DNP if and only if the inequalities* $-\mathbf{A}_0^{-1}\mathbf{A}_\partial \ge \mathbf{0}$ *and* $\mathbf{A}_0^{-1}\mathbf{B} \ge \mathbf{0}$ *hold.*

Proof. The statement follows directly from the following equivalent form of (4.1):

$$\mathbf{u}^{n+1} = -\mathbf{A}_0^{-1}\mathbf{A}_\partial\mathbf{g}^{n+1} + \mathbf{A}_0^{-1}\mathbf{B}\begin{bmatrix}\mathbf{u}^{n}\\ \mathbf{g}^{n}\end{bmatrix} + \mathbf{A}_0^{-1}\mathbf{h}^n,\quad n=0,1,\dots. \tag{4.3}$$

∎

DEFINITION 4.3. *A PO-iteration is said to satisfy the discrete maximum norm contractivity property (DMNC) if for all vectors* $\hat{\mathbf{u}}^n, \tilde{\mathbf{u}}^n, \mathbf{g}^n, \mathbf{g}^{n+1}, \mathbf{h}^n$ *the relation*

$$\|\hat{\mathbf{u}}^{n+1}-\tilde{\mathbf{u}}^{n+1}\|_\infty \le \|\hat{\mathbf{u}}^n-\tilde{\mathbf{u}}^n\|_\infty \tag{4.4}$$

is valid, where $\hat{\mathbf{u}}^{n+1}$ *and* $\tilde{\mathbf{u}}^{n+1}$ *are computed from* (4.1) *by setting* $\mathbf{u}^n = \hat{\mathbf{u}}^n$ *and* $\mathbf{u}^n = \tilde{\mathbf{u}}^n$.

THEOREM 4.4. *A PO-iteration satisfies the DMNC if and only if* $\|\mathbf{A}_0^{-1}\mathbf{B}_0\|_\infty \le 1$.

Proof. We apply (4.3) with the two different vectors $\mathbf{u}^n = \hat{\mathbf{u}}^n$ and $\mathbf{u}^n = \tilde{\mathbf{u}}^n$. Hence we obtain

$$\hat{\mathbf{u}}^{n+1}-\tilde{\mathbf{u}}^{n+1} = \mathbf{A}_0^{-1}\mathbf{B}_0(\hat{\mathbf{u}}^n-\tilde{\mathbf{u}}^n).$$

The relation (4.4) is valid for all $\hat{\mathbf{u}}^n$ and $\tilde{\mathbf{u}}^n$ vectors if and only if $\|\mathbf{A}_0^{-1}\mathbf{B}_0\|_\infty \le 1$. ∎

DEFINITION 4.5. *A PO-iteration is said to satisfy the discrete maximum-minimum principle (DMP) if the relation*

$$\begin{aligned}\min\{\mathbf{u}^n,\mathbf{g}^n,\mathbf{g}^{n+1}\} + \min\{0,\mathbf{A}_0^{-1}\mathbf{h}^n\}\\ \le u_i^{n+1} \le \max\{\mathbf{u}^n,\mathbf{g}^n,\mathbf{g}^{n+1}\} + \max\{0,\mathbf{A}_0^{-1}\mathbf{h}^n\}\end{aligned} \tag{4.5}$$

is valid for each choice $\mathbf{u}^n, \mathbf{g}^n, \mathbf{g}^{n+1}, \mathbf{h}^n$, $i=1,\dots,N$, *and* $n \ge 0$. *(The* max *and* min *are understood elementwise.)*

Now some necessary conditions are listed for the DMP.

LEMMA 4.6. *For PO-iterations the DMP implies the DNP.*

Proof. Let us suppose that the DMP is valid for the PO-iteration and $\mathbf{u}^n$, $\mathbf{g}^n$, $\mathbf{g}^{n+1}$, $\mathbf{A}_0^{-1}\mathbf{h}^n$ are nonnegative vectors. Then the left inequality in (4.5) shows the nonnegativity of $\mathbf{u}^{n+1}$. ∎

LEMMA 4.7. *If a PO-iteration satisfies the DMP, then* $\mathbf{Ae} = \mathbf{Be}$.

Proof. Choosing the vectors $\mathbf{h}^n = \mathbf{0}$, $\mathbf{u}^n = \mathbf{e}_0$, $\mathbf{g}^n = \mathbf{g}^{n+1} = \mathbf{e}_\partial$, (4.5) results in $\mathbf{u}^{n+1} = \mathbf{e}_0$. That is, $\mathbf{Ae} = \mathbf{Be}$ in view of (4.1), which completes the proof. ∎

LEMMA 4.8. *For PO-iterations the DNP and condition* $\mathbf{Ae} = \mathbf{Be}$ *imply the DMNC.*

Proof. The DNP ensures the relations $-\mathbf{A}_0^{-1}\mathbf{A}_\partial \geq \mathbf{0}$ and $\mathbf{A}_0^{-1}\mathbf{B} \geq \mathbf{0}$. Hence

$$\mathbf{A}_0^{-1}\mathbf{B}_0\mathbf{e}_0 \leq \mathbf{A}_0^{-1}\mathbf{Be} = \mathbf{A}_0^{-1}\mathbf{Ae} = \mathbf{A}_0^{-1}\mathbf{A}_0\mathbf{e}_0 + \mathbf{A}_0^{-1}\mathbf{A}_\partial\mathbf{e}_\partial \leq \mathbf{e}_0.$$

This shows that $\|\mathbf{A}_0^{-1}\mathbf{B}_0\|_\infty \leq 1$; that is, the DMNC is valid. ∎

Remark 4.9. Because the DMP implies both the DNP and condition $\mathbf{Ae} = \mathbf{Be}$, the DMP implies the DMNC.

LEMMA 4.10. *If a PO-iteration satisfies the DNP and condition* $\mathbf{Ae} = \mathbf{Be}$*, then* $\mathbf{0} \leq -\mathbf{A}_0^{-1}\mathbf{A}_\partial\mathbf{e}_\partial \leq \mathbf{e}_0$.

Proof. The DNP ensures the relations $-\mathbf{A}_0^{-1}\mathbf{A}_\partial \geq \mathbf{0}$ and $\mathbf{A}_0^{-1}\mathbf{B} \geq \mathbf{0}$. Thus, the relation $\mathbf{0} \leq -\mathbf{A}_0^{-1}\mathbf{A}_\partial\mathbf{e}_\partial$ is trivial. Let us multiply the equality $\mathbf{Ae} = \mathbf{A}_0\mathbf{e}_0 + \mathbf{A}_\partial\mathbf{e}_\partial = \mathbf{Be}$ by the matrix $\mathbf{A}_0^{-1}$, and let us express the term $-\mathbf{A}_0^{-1}\mathbf{A}_\partial\mathbf{e}_\partial$. We obtain the inequality

$$\mathbf{0} \leq -\mathbf{A}_0^{-1}\mathbf{A}_\partial\mathbf{e}_\partial = \mathbf{e}_0 - \mathbf{A}_0^{-1}\mathbf{Be} \leq \mathbf{e}_0$$

by virtue of the nonnegativity of the vector $\mathbf{A}_0^{-1}\mathbf{Be}$. This completes the proof. ∎

4.3. Qualitatively adequate one-step iterations. We have seen in the previous section that the properties

$$\begin{array}{ll} \text{(P1)} & \mathbf{A}_0^{-1}\mathbf{B} \geq \mathbf{0}, \\ \text{(P2)} & -\mathbf{A}_0^{-1}\mathbf{A}_\partial \geq \mathbf{0}, \\ \text{(P3)} & \mathbf{Ae} = \mathbf{Be} \end{array}$$

are necessary conditions of the DMP for PO-iterations. In the next theorem, we prove that they are sufficient conditions as well. In order to make the expressions much simpler, we introduce some notation. We define the values

$$v_{\min}^n = \min\{\mathbf{v}^n\}, \quad v_{\max}^n = \max\{\mathbf{v}^n\},$$

and vectors

$$\mathbf{v}_{\max}^n = v_{\max}^n\mathbf{e} \in \mathbb{R}^{\bar{N}}, \quad \mathbf{v}_0^n = v_{\max}^n\mathbf{e}_0 \in \mathbb{R}^N, \quad \mathbf{v}_\partial^n = v_{\max}^n\mathbf{e}_\partial \in \mathbb{R}^{N_\partial}.$$

THEOREM 4.11. *A PO-iteration satisfies the DMP if and only if it satisfies the conditions* (P1)–(P3).

Proof. The necessity of the condition follows directly from Theorem 4.2 and Lemmas 4.6 and 4.7. To show the sufficiency, we first prove the inequality on the right-hand side in (4.5). It follows from (P3) that $\mathbf{Bv}_{\max}^n = \mathbf{Av}_{\max}^n$. Using (P1), we

obtain

$$\begin{aligned}
\mathbf{u}^{n+1} &= -\mathbf{A}_0^{-1}\mathbf{A}_\partial\, \mathbf{g}^{n+1} + \mathbf{A}_0^{-1}\mathbf{B}\mathbf{v}^n + \mathbf{A}_0^{-1}\mathbf{h}^n \\
&\le -\mathbf{A}_0^{-1}\mathbf{A}_\partial\, \mathbf{g}^{n+1} + \mathbf{A}_0^{-1}\mathbf{B}\mathbf{v}^n_{\max} + \mathbf{A}_0^{-1}\mathbf{h}^n \\
&= -\mathbf{A}_0^{-1}\mathbf{A}_\partial\, \mathbf{g}^{n+1} + \mathbf{A}_0^{-1}\mathbf{A}\mathbf{v}^n_{\max} + \mathbf{A}_0^{-1}\mathbf{h}^n \\
&\le -\mathbf{A}_0^{-1}\mathbf{A}_\partial\, \mathbf{g}^{n+1} + \mathbf{A}_0^{-1}[\mathbf{A}_0|\ \mathbf{A}_\partial]\mathbf{v}^n_{\max} + \max\{0, \mathbf{A}_0^{-1}\mathbf{h}^n\}\mathbf{e}_0 \\
&= -\mathbf{A}_0^{-1}\mathbf{A}_\partial\, \mathbf{g}^{n+1} + \mathbf{v}_0^n + \mathbf{A}_0^{-1}\mathbf{A}_\partial\mathbf{v}_\partial^n + \max\{0, \mathbf{A}_0^{-1}\mathbf{h}^n\}\mathbf{e}_0.
\end{aligned}$$

Regrouping the above inequality, we get

$$\mathbf{u}^{n+1} - \mathbf{v}_0^n - \max\{0, \mathbf{A}_0^{-1}\mathbf{h}^n\}\mathbf{e}_0 \le -\mathbf{A}_0^{-1}\mathbf{A}_\partial \left(\mathbf{g}^{n+1} - \mathbf{v}_\partial^n\right).$$

Hence, for the ith coordinate of both sides we obtain

$$\begin{aligned}
u_i^{n+1} - v^n_{\max} - \max\{0, \mathbf{A}_0^{-1}\mathbf{h}^n\} &\le \sum_{j=1}^{N_\partial} \left(\left(-\mathbf{A}_0^{-1}\mathbf{A}_\partial\right)_{ij} \cdot \left(g_j^{n+1} - v^n_{\max}\right)\right) \\
&\le \left(\sum_{j=1}^{N_\partial} \left(-\mathbf{A}_0^{-1}\mathbf{A}_\partial\right)_{ij} \right) \cdot \max\{0, \max_j\{g_j^{n+1} - v^n_{\max}\}\} \\
&\le \max\{0, \max_j\{g_j^{n+1} - v^n_{\max}\}\},
\end{aligned}$$

where we applied property (P2) and Lemma 4.10. Finally, expressing u_i^{n+1} we obtain the required inequality. The inequality on the left-hand side of (4.5) can be proved similarly. This completes the proof. ∎

Theorem 4.11, Lemma 4.6, and Remark 4.9 yield that a PO-iteration possesses all three qualitative properties (DMP, DNP, and DMNC) if and only if it satisfies the properties (P1)–(P3). Hence the iterative process (4.1) with the properties (P1)–(P3) is called a *qualitatively adequate one-step iteration*.

5. Qualitative properties of the numerical solutions of parabolic problems. We apply the results, which were formulated for arbitrary PO-iterations, of the previous section for the fully discretized numerical model (3.9). Based on Theorem 3.2, the finite difference and Galerkin finite element methods can be written in the form of a PO-iteration with the choice $\mathbf{h}^n = \Delta t \mathbf{f}^{(n,\theta)}$. This is why the qualitative properties (DMP, DNP, and DMNC) of the numerical methods can be defined in the same manner as those of the PO-iterations.

THEOREM 5.1. *Relation* (P3) *holds both for the finite difference and finite element methods.*

Proof. Owing to the relation $\mathbf{A} - \mathbf{B} = (\mathbf{M} + \theta\Delta t\mathbf{K}) - (\mathbf{M} - (1-\theta)\Delta t\mathbf{K}) = \Delta t\mathbf{K}$, property (P3) is equivalent to the equality $\mathbf{K}\mathbf{e} = \mathbf{0}$. For the finite difference method the statement follows directly from the equalities in (3.6). For the finite element method, we have

$$\begin{aligned}
(\mathbf{K}\mathbf{e})_i &= \sum_{j=1}^{\bar N} K_{ij} = \sum_{j=1}^{\bar N} \int_\Omega \kappa(\mathbf{x}) \cdot \mathrm{grad}\phi_j(\mathbf{x}) \cdot \mathrm{grad}\phi_i(\mathbf{x})\, \mathrm{d}\mathbf{x} \\
&= \int_\Omega \kappa(\mathbf{x}) \cdot \mathrm{grad} \left(\sum_{j=1}^{\bar N} \phi_j(\mathbf{x}) \right) \cdot \mathrm{grad}\phi_i(\mathbf{x})\, \mathrm{d}\mathbf{x} \\
&= \int_\Omega \kappa(\mathbf{x}) \cdot \mathrm{grad}1 \cdot \mathrm{grad}\phi_i(\mathbf{x})\, \mathrm{d}\mathbf{x} = 0. \qquad \blacksquare
\end{aligned}$$

THEOREM 5.2. *For the finite difference and Galerkin finite element models of the problem* (2.4)–(2.5), *the implications*

$$DNP \Longleftrightarrow DMP \Longrightarrow DMNC$$

are valid. That is the DMP and DNP are equivalent and both imply the DMNC.

Proof. The first part of the statement follows from Theorem 4.2, Lemma 4.6, and Theorem 5.1. The second one is a consequence of Lemma 4.8. □

COROLLARY 5.3. *A finite difference or a Galerkin finite element method for the problem* (2.4)–(2.5) *is qualitatively adequate if and only if it satisfies properties* (P1)–(P2), *that is, if the method preserves the nonnegativity.*

Remark 5.4. In view of the results of [13], there exist parameter choices when the finite difference or the finite element method is contractive in maximum norm, but it does not preserve the nonnegativity.

Remark 5.5. Let us notice that our results can be applied not only to the finite difference and the Galerkin finite element method but also to any numerical method that can be written in the form of a PO-iteration.

6. Sufficient a priori conditions of the qualitative properties. The necessary and sufficient conditions in the previous sections cannot be checked easily without computing the elements of the matrices. More precisely, the theorems do not state anything explicitly about the choice of the mesh and the choice of the parameters θ and Δt (which define the elements of the matrices $\mathbf{M}$ and $\mathbf{K}$) in order to guarantee the qualitative properties. Our aim is to find conditions that can be checked a priori and imply the DNP / DMP. Then, these conditions, according to Theorem 5.2, imply the qualitative adequateness of the discrete models. In this section, we will give some a priori conditions. In order to present a complete work for researchers involved in scientific computing, we also added two corresponding results from the literature.

THEOREM 6.1. *The Galerkin finite element solution of problem* (2.4)–(2.5), *combined with the θ-method in the time discretization, satisfies the DMP if the conditions*

$$\begin{aligned}
&\text{(S1)} \quad K_{ij} \le 0, \quad i \ne j,\ i = 1, \dots, N,\ j = 1, \dots, \bar{N},\\
&\text{(S2)} \quad A_{ij} = M_{ij} + \theta \Delta t K_{ij} \le 0, \quad i \ne j,\ i = 1, \dots, N,\ j = 1, \dots, \bar{N},\\
&\text{(S3)} \quad B_{ii} = M_{ii} - (1-\theta)\Delta t K_{ii} \ge 0, \quad i = 1, \dots, N,
\end{aligned}$$

are fulfilled. The DMP is valid for the finite difference solution of the problem if (S3) *holds.*

Proof. We have to show that under the assumptions of the theorem the properties $(P1)$–$(P2)$ hold. For the finite difference method $\mathbf{A}_0^{-1} \ge \mathbf{0}$ and conditions (S1) and (S2) are fulfilled automatically. This implies property (P2). Condition (S3) implies that the main diagonal of $\mathbf{B}$ is nonnegative. The off-diagonal of $\mathbf{B}$ is also nonnegative because the off-diagonal of $\mathbf{K}$ is nonpositive. Hence, by virtue of $\mathbf{A}_0^{-1} \ge \mathbf{0}$, we obtain property (P1).

In the case of the finite element method, relations (S1) and (S3) yield $\mathbf{B} \ge \mathbf{0}$. Condition $\mathbf{A}_\partial \le \mathbf{0}$ follows from (S2). $\mathbf{A}_0$ is a Gram-matrix, and because of (S2), the off-diagonal elements of $\mathbf{A}_0$ are nonpositive. This implies that $\mathbf{A}_0$ is an M-matrix (see [1, p. 134]); thus $\mathbf{A}_0^{-1} \ge \mathbf{0}$. These facts yield (P1) and (P2), and thus the DMP. □

In the expressions of this section, for the sake of simplicity, fractions with zero denominators are understood as infinity.

6.1. Conditions for the finite difference method. Let us consider the finite difference method, where, because of the relations (3.5)–(3.7), condition (S3) is valid for 1D problems if

$$C_i - (1-\theta)\Delta t \left(\frac{2\kappa_{i+x/2}}{(h_{i+x}+h_{i-x})h_{i+x}} + \frac{2\kappa_{i-x/2}}{(h_{i+x}+h_{i-x})h_{i-x}} \right) \geq 0, \quad i = 1, \ldots, N.$$

The condition

$$\Delta t \leq \frac{C_{\min} h_{\min}^2}{2\kappa_{\max}(1-\theta)}$$

guarantees the validity of the above relations, where $h_{\min}$ is the minimal spatial step size in the mesh and $\kappa_{\max} = \sup_{\mathbf{x}\in\Omega}\{\kappa(\mathbf{x})\}$. In two dimensions, similarly, we obtain the sufficient condition

$$\Delta t \leq \frac{C_{\min} h_{\min}^2}{4\kappa_{\max}(1-\theta)},$$

and in three dimensions

$$\Delta t \leq \frac{C_{\min} h_{\min}^2}{6\kappa_{\max}(1-\theta)}.$$

The above results can be summarized as follows.

THEOREM 6.2. *The DMP is always satisfied for the implicit Euler ($\theta = 1$) finite difference method. For other finite difference methods, the DMP can be guaranteed by the condition*

$$\Delta t \leq \frac{C_{\min} h_{\min}^2}{2d\kappa_{\max}(1-\theta)} \quad (\theta \neq 1)$$

(d is the dimension of Ω).

Remark 6.3. For the explicit Euler method, Theorem 6.2 yields the condition

$$\Delta t \leq \frac{C_{\min} h_{\min}^2}{2d\kappa_{\max}},$$

and for the Crank–Nicolson method we obtain

$$\Delta t \leq \frac{C_{\min} h_{\min}^2}{d\kappa_{\max}}.$$

We remark that for 1D problems on uniform meshes of step-size h, with constant material parameters (C_0, κ_0), the necessary and sufficient condition of the DMP is [6]

$$\Delta t \leq \frac{C_0 h^2}{2\kappa_0(1-\theta)}.$$

6.2. Conditions for the finite element method. For finite element methods, condition (S1) is generally guaranteed by some geometrical requirements for the mesh. Moreover, conditions (S2) and (S3) can be achieved by some lower and upper bounds for the time-step.

6.2.1. Finite element method in one dimension. Computing the elements of the stiffness matrix, we can check that the off-diagonal elements are nonpositive, and thus condition (S1) is satisfied. Computing additionally the elements of the mass matrix we obtain that (S2) and (S3) are valid if

$$\frac{C_{\max}h_{\max}^2}{6\theta\kappa_{\min}} \le \Delta t \le \frac{C_{\min}h_{\min}^2}{3(1-\theta)\kappa_{\max}}.$$

Thus we obtain the next theorem.

THEOREM 6.4. *The Galerkin finite element solution of the* 1*D problem* (2.4)–(2.5) *with piecewise linear elements satisfies the DMP if*

$$\frac{C_{\max}h_{\max}^2}{6\theta\kappa_{\min}} \le \Delta t \le \frac{C_{\min}h_{\min}^2}{3(1-\theta)\kappa_{\max}}, \quad \theta \in [\theta_l, 1)\,,$$

$$\frac{C_{\max}h_{\max}^2}{6\kappa_{\min}} \le \Delta t, \quad \theta = 1,$$

where $\theta_l = C_{\max}\kappa_{\max}h_{\max}^2/(C_{\max}\kappa_{\max}h_{\max}^2 + 2C_{\min}\kappa_{\min}h_{\min}^2)$.

Let us notice that unlike in the case of finite difference methods, the above theorem yields a lower bound for the time-step. This is a general phenomenon for finite element methods. This lower bound exists not only in sufficient conditions but in necessary and sufficient conditions, too. This is shown by the next theorem obtained in [6] for the DNP.

THEOREM 6.5. *The Galerkin finite element solution of the* 1*D problem* (2.4)–(2.5) *with piecewise linear elements on uniform grid and with constant material parameters satisfies the DMP if and only if the relations*

$$\frac{C_0h^2}{6\theta\kappa_0} \le \Delta t \le \frac{C_0h^2}{3(1-\theta)\kappa_0}, \quad \theta \in [1/3, 1)\,,$$

$$\frac{C_0h^2}{6\kappa_0} \le \Delta t, \quad \theta = 1,$$

hold.

6.2.2. Finite element method in two dimensions. In two dimensions we apply the hybrid mesh in the spatial discretization. On triangular elements linear basis functions are used, and on rectangular elements bilinear basis functions are used.

Let us consider a hybrid mesh $\mathcal{T}_h$. Let us define the number

$$\sigma_T = \min\{\cot\alpha_1, \cot\alpha_2, \cot\alpha_3\}$$

for each triangle T of the mesh, where α_1, α_2, and α_3 denote the angles of the triangle. Let us define $\sigma = \min_{T\in\mathcal{T}_h}\sigma_T$. Furthermore, let us introduce the value

$$\mu_R = \frac{2\min^2\{a,b\} - \max^2\{a,b\}}{ab}$$

for each rectangle R, where a and b denote the edges of the rectangle. Let us define $\mu = \min_{R\in\mathcal{T}_h}\mu_R$. The hybrid mesh $\mathcal{T}_h$ is said to be of strictly compact type if $\sigma > 0$

and $\mu > 0$. The condition $\sigma > 0$ means that each angle of the triangles in the mesh is of acute type, and $\mu > 0$ means that the longer edges of the rectangles are not greater than $\sqrt{2}$ times the shorter ones. The main result of [8] is formulated as follows.

THEOREM 6.6. *The Galerkin finite element solution of the 2D problem* (2.4)–(2.5) *using linear/bilinear basis functions on a hybrid mesh of strictly compact type satisfies the DMP if*

$$\frac{C_{\max}}{6\,\theta\,\kappa_{\min}\,\min\{\frac{\mu}{3\lambda^{\#}_{\max}},\frac{\sigma}{\lambda^{\triangle}_{\max}}\}} \le \Delta t \le \frac{C_{\min}}{9\,(1-\theta)\,\kappa_{\max}\,\max\{\frac{\gamma^{\#}_{\max}}{3\lambda^{\#}_{\min}},\frac{\gamma^{\triangle}_{\max}}{4\,\lambda^{\triangle}_{\min}}\}} \tag{6.1}$$

is fulfilled, where

$$\gamma^{\triangle}_{\max} = \max_{T\in\mathcal{T}_h}\left\{\frac{l^2_{\max}}{\operatorname{area}(T)}\right\},\quad \gamma^{\#}_{\max} = \max_{R\in\mathcal{T}_h}\left\{\frac{a^2+b^2}{\operatorname{area}(R)}\right\},$$

$$\lambda^{\triangle}_{\min} = \min_{T\in\mathcal{T}_h}\operatorname{area}(T),\ \lambda^{\triangle}_{\max} = \max_{T\in\mathcal{T}_h}\operatorname{area}(T),$$

$$\lambda^{\#}_{\min} = \min_{R\in\mathcal{T}_h}\operatorname{area}(R),\ \lambda^{\#}_{\max} = \max_{R\in\mathcal{T}_h}\operatorname{area}(R).$$

The symbols $\operatorname{area}(T)$ *and* $\operatorname{area}(R)$ *denote the area of the triangle* T *and the rectangle* R*, respectively.* $l_{\max}$ *is the length of the longest edge in* T*.*

Thus, for strictly compact meshes, the off-diagonal elements of the stiffness matrix are nonpositive; hence condition (S1) is fulfilled. Relation (6.1) implies the conditions (S2) and (S3).

Remark 6.7. For purely rectangular meshes, we have the weaker lower bound for Δt in the form

$$\Delta t \ge \frac{C_{\max}\lambda^{\#}_{\max}}{3\,\theta\,\kappa_{\min}\,\mu}.$$

For a square mesh with step-size h and with constant material parameters a sufficient condition of the DMP is

$$\frac{C_0 h^2}{3\theta\kappa_0} \le \Delta t \le \frac{C_0 h^2}{6(1-\theta)\kappa_0},\quad \theta\in[2/3,1).$$

(In case of $\theta = 1$ the upper bound disappears.) This shows that in this case the DMP can be guaranteed only (with our sufficient condition) for methods with $\theta \ge 2/3$.

6.2.3. Finite element method in three dimensions. In three dimensions we investigate two different meshes. The first one is the rectangular mesh and the second one is the tetrahedral mesh. Let us start with the first one, where trilinear basis functions are applied. With simple but tedious calculations we can compute the elements of the mass and stiffness matrices, and we can notice that the condition (S1) can be valid only for uniform meshes. Moreover, we can observe that condition (S2) cannot be guaranteed, because there are also positive off-diagonal elements in the matrix $\mathbf{A}$. Thus $\mathbf{A}_0$ is not an M-matrix in this case. We have to guarantee the nonnegative matrix inverse by means of other tools. In [12], the so-called Lorenz-criterion [21] was applied.

THEOREM 6.8. *The Galerkin finite element solution of the* 3*D problem* (2.4)–(2.5) *using trilinear basis functions on a uniform rectangular mesh with constant material parameters satisfies the DMP if*

$$\frac{(259+13\sqrt{409})C_0h^2}{36\theta\kappa_0} \le \Delta t \le \frac{C_0h^2}{9(1-\theta)\kappa_0}, \quad \theta \in [0.9924, 1),$$

$$14.4975\frac{C_0h^2}{\kappa_0} \approx \frac{(259+13\sqrt{409})C_0h^2}{36\kappa_0} \le \Delta t, \quad \theta = 1.$$

For tetrahedral meshes the DMP was discussed in [11]. We will recall this result. Let us consider a tetrahedral mesh $\mathcal{T}_h$ of the solution domain Ω, and define the number σ_T for each tetrahedron T of $\mathcal{T}_h$ as $\sigma_T = \min\{\cos\beta_1, \dots, \cos\beta_6\}$, where $\beta_1, \dots, \beta_6$ denote the angles made by any two faces of the tetrahedron T. If each angle is less than $\pi/2$, then $\sigma_T > 0$. Let us set $\sigma = \min_{T\in\mathcal{T}_h} \sigma_T$. If $\sigma > 0$, then the mesh is called of strictly acute type.

THEOREM 6.9. *The Galerkin finite element solution of the* 3*D problem* (2.4)–(2.5) *with constant material parameters using linear basis functions on a strictly acute type tetrahedral mesh satisfies the DMP if the condition*

$$\frac{C_0p_{\max}^2}{20\sigma\theta\kappa_0} \le \Delta t \le \frac{C_0p_{\min}^2}{10(1-\theta)\kappa_0}$$

holds, where $p_{\min}$ *and* $p_{\max}$ *denote the minimal and maximal perpendicular length in the mesh.*

7. Numerical examples. In this section, we demonstrate the validity of our results on some numerical examples in one, two, and three dimensions. For the sake of simplicity, we suppose that the problem (2.4)–(2.5) describes a heat conduction process (see Remark 2.7). The material parameters are set to be constant one. Furthermore, we suppose that there are no heat sources or sinks present in the computational domain.

7.1. Examples in one dimension. In the first example, the 1D heat equation is solved with the Crank–Nicolson ($\theta = 1/2$) finite difference method on the interval [0,1]. We are interested in the approximation of the temperature at the time-level $t = 1$. We apply an equidistant mesh with the spatial step-size $h = 1/30$. The temperatures at the left and right boundaries are given by the functions $\mu_1(t) = 1$ and $\mu_2(t) = 0$, respectively. A nonnegative approximation of a continuous and nonnegative initial function can be seen in Figure 7.1. Using this initial grid-function, the approximations of the temperature at the time-level $t = 1$ can be seen in Figure 7.2. The figures on the left-hand and right-hand sides were obtained using the time-steps $\Delta t = 1/11$ and $\Delta t = 1/31$, respectively. These approximations show that neither the NP nor the MP is valid for the above chosen time-steps. In order to get a qualitatively correct approximation, Δt has to satisfy the condition

$$\Delta t \le \frac{h^2}{2(1-\theta)} = \frac{1}{900}$$

(cf. (6.3)). This bound shows that Δt was chosen to be too large in the previous numerical example. Naturally, the above upper bound is the necessary and sufficient

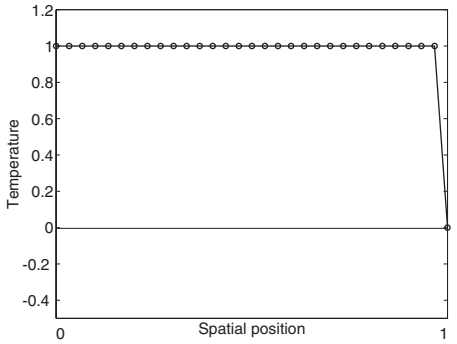


FIG. 7.1. *Approximation of the initial function.*

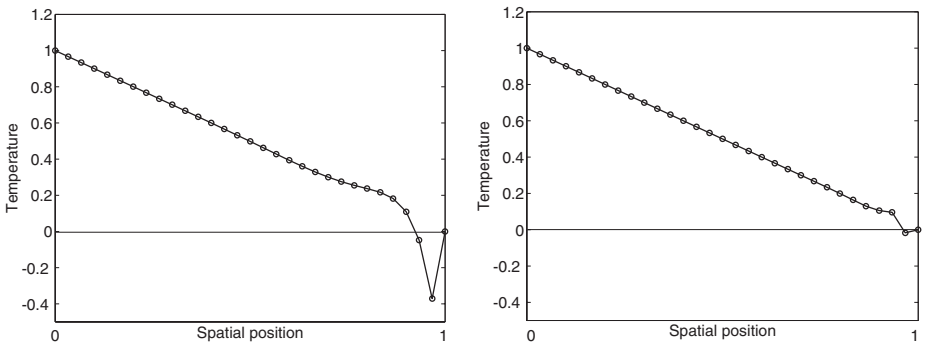


FIG. 7.2. *Approximations at the time-level* $t = 1$ *with the time-steps* $\Delta t = 1/11$ *and* $\Delta t = 1/31$.

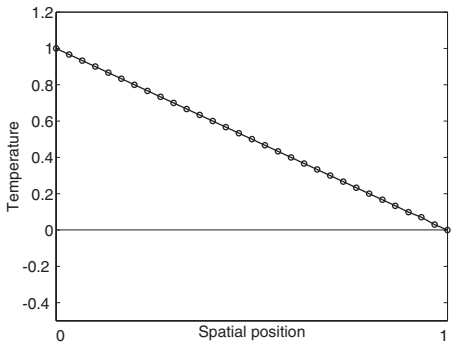


FIG. 7.3. *Approximation at the time-level* $t = 1$ *with the time-step* $\Delta t = 1/51$.

condition of the DMP (DNP) for all initial functions. This means that for certain initial functions, larger time-steps can also result in a qualitatively correct numerical solution. For example, Figure 7.3 shows a nonnegative approximation with the time-step $\Delta t = 1/51\,1/900$ at the time-level $t = 1$.

One can think that by choosing the time-step to be sufficiently small we can obtain a qualitatively adequate solution. The next example shows that this is generally not

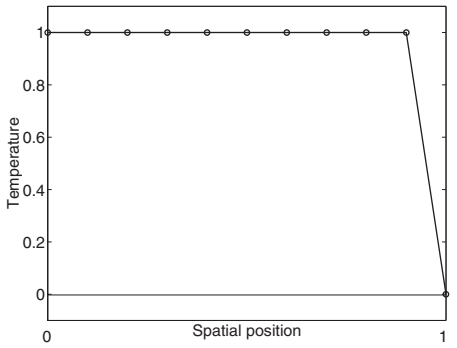


FIG. 7.4. *Approximation of the initial function.*

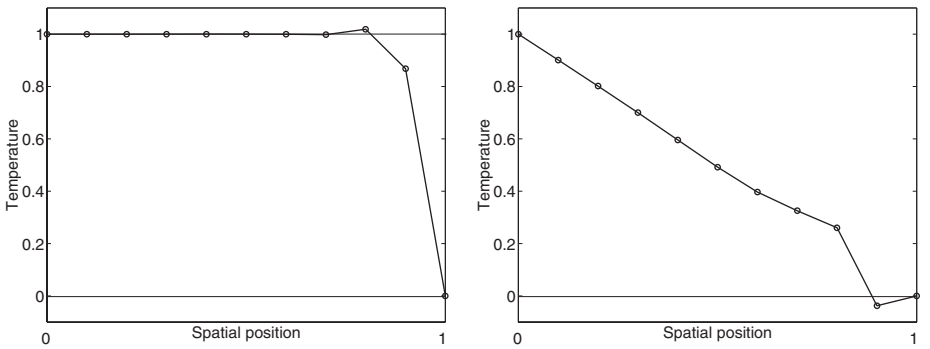


FIG. 7.5. *Approximations at the time-levels* $t = 1/1000$ *and* $t = 1$ *applying the time-steps* $\Delta t = 1/100000$ *and* $\Delta t = 1/9$*, respectively.*

the case. Namely, as we have shown, for finite element methods not only upper but also lower bounds do exist.

Let us solve the 1D heat equation with the Galerkin finite element method on the interval [0,1] using the Crank–Nicolson ($\theta = 1/2$) scheme in the time integration. Let us suppose that the initial function is a nonnegative continuous function and it takes on values only from the interval $[0, 1]$. The discretization of the initial function is carried out on an equidistant mesh with $h = 1/10$ (Figure 7.4). The temperatures at the left and right boundaries are given by the functions $\mu_1(t) = 1$ and $\mu_2(t) = 0$, respectively. The numerical solutions calculated at the time-levels $t = 1/1000$ and $t = 1$ with the time steps $\Delta t = 1/100000$ and $\Delta t = 1/9$ can be seen, respectively, on the left-hand and right-hand sides of Figure 7.5. Both numerical solutions are qualitatively incorrect. Namely, the MP is broken because the solutions have nonnegative values and values that are greater than one. In order to obtain qualitatively adequate solutions we can apply Theorem 6.4, which states that the MP is valid providing that Δt is chosen according to the relation $1/300 = 0.0033 \leq \Delta t \leq 0.0067 = 2/300$. In Figure 7.6, the numerical solution is calculated at the time-level $t = 1$ with the time-step $\Delta t = 0.005$, which falls into the above interval. As it was expected, the solution has values only from the interval $[0, 1]$.

In the third example we solve again the 1D heat equation on the interval [0,1]. The temperature at the right-hand side is equal to one ($\mu_2(t) = 1$), while the temperature

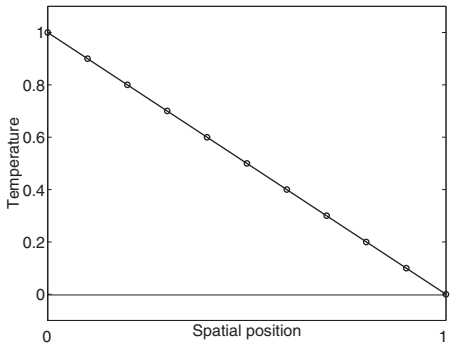


FIG. 7.6. *Approximation at the time-level* $t = 1$ *with the time-step* $\Delta t = 0.005$.

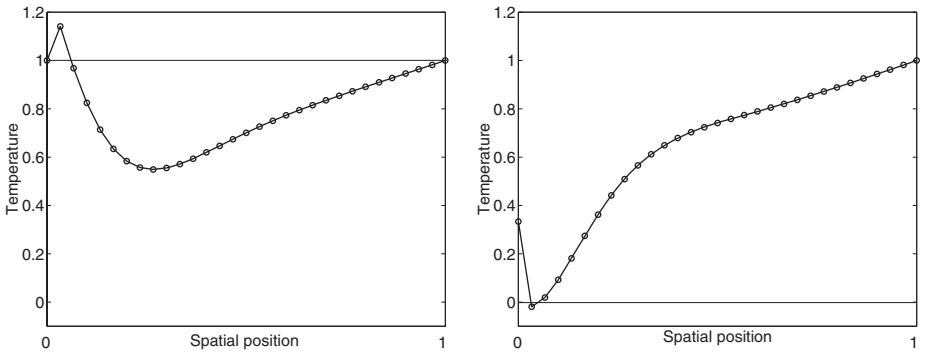


FIG. 7.7. *Approximations at the time-levels* $t = 300/30$ *and* $t = 302/30$ *applying the time-step* $\Delta t = 1/30$.

at the left-hand side changes periodically according to the function

$$\mu_1(t) = 2|-\mathrm{frac}(100t/2) + 0.5|$$

(frac$(100t/2)$ means the fractional part of $100t/2$). Naturally, $0 \le \mu_1(t) \le 1$. Let the initial function be the constant one function in $(0, 1)$.

Our aim is to approximate the solution at the time-level $t = 10$. We apply the finite difference method on the equidistant mesh with $h = 1/30$. We choose the Crank–Nicolson method for the time-integration with $\Delta t = 1/30$. The numerical solution is depicted on the left-hand side of Figure 7.7. Two time-levels further, that is, at $t = 302/30$, we obtain the solution shown on the right-hand side of the same figure. In the case of the numerical solutions shown in Figure 7.7, the MP is broken. The time-step was chosen to be too large (cf. (6.3)). We can observe that the numerical solution will be periodic after the transient period ($t > 3$) (the boundary condition is periodic and the effect of the initial function disappears). The approximations shown in Figure 7.7 appear regularly in every third time-step. Thus the numerical example demonstrates that it is possible that the qualitatively bad behavior of the numerical solution does not disappear while increasing the time-level at which the numerical solution is computed.

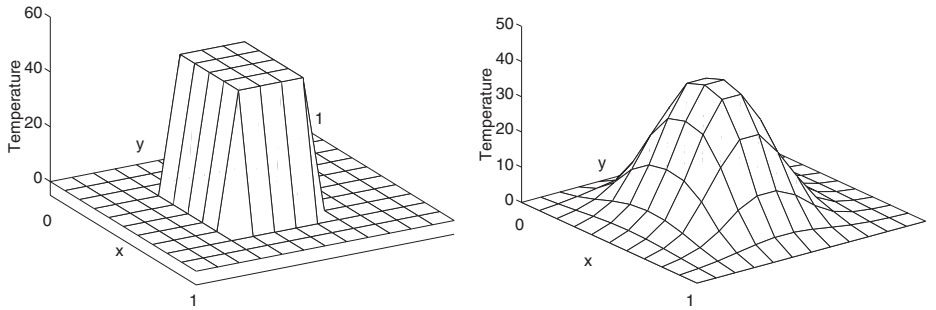


FIG. 7.8. *Approximation of the initial function. Approximation at the first time-level with* $\Delta t = 0.01$.

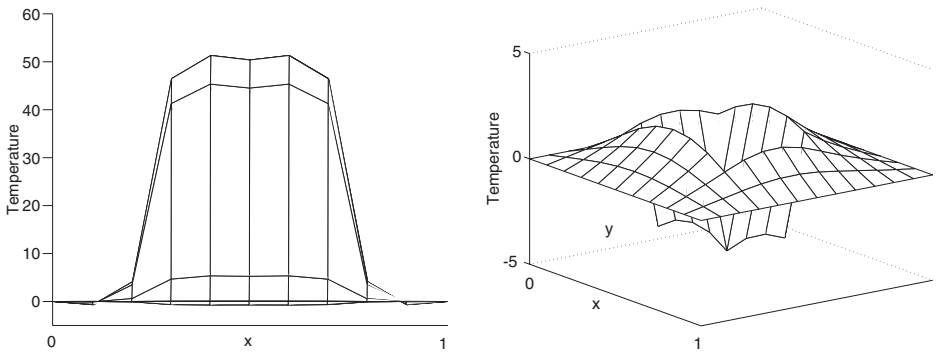


FIG. 7.9. *Approximations at the first time-level with* $\Delta t = 0.0005$ *and* $\Delta t = 0.5$.

7.2. Examples in two dimensions. We solve the 2D heat equation on the unit square $[0,1]\times[0,1]$ with homogeneous boundary condition. We apply the finite element method with bilinear elements on a rectangular mesh with $\Delta x = 1/10$ and $\Delta y = 1/12$. In the time-discretization the θ-method is used with $\theta = 0.9$ and with a fixed time-step Δt. A nonnegative discretization of a nonnegative and continuous initial function can be seen on the left-hand side of Figure 7.8. With this discretization, however, the approximation of the temperature at the first time-level has negative value both for $\Delta t = 0.0005$ and $\Delta t = 0.5$ (see the left-hand and right-hand sides in Figure 7.9). We remark that, with $\Delta t = 0.0005$, the numerical solution has negative components at the first 11 time-levels, while with $\Delta t = 0.5$, the solution has negative components at every time-level. Choosing the time-step to be $\Delta t = 0.01$, we obtain an adequate approximation (right-hand side of Figure 7.8).

We can calculate the necessary and sufficient condition of the DNP using Theorem 4.2, which results in the condition $0.0050 \le \Delta t \le 0.0155$. This shows that the chosen time-steps were too small or too large. A sufficient condition obtained applying the results of Theorem 6.6 is $0.0066 \le \Delta t \le 0.0136$.

In the second example, we apply the finite difference method with $\theta = 0.9$ for the initial approximation seen on the left-hand side of Figure 7.8. The results at the

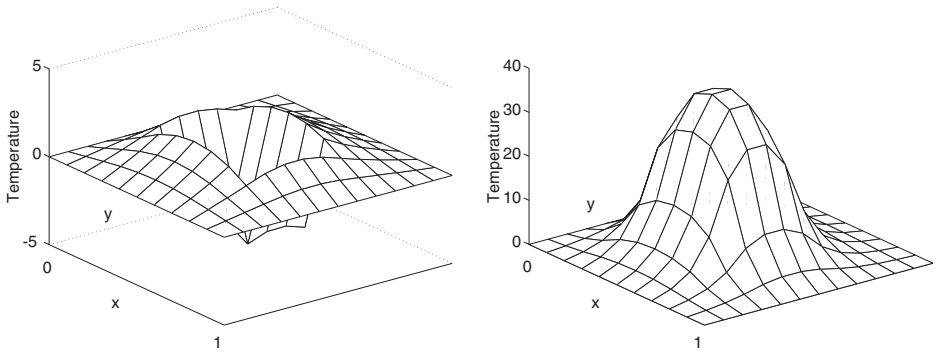


FIG. 7.10. *Approximations at the first time-level with* $\Delta t = 0.5$ *and* $\Delta t = 0.01$.

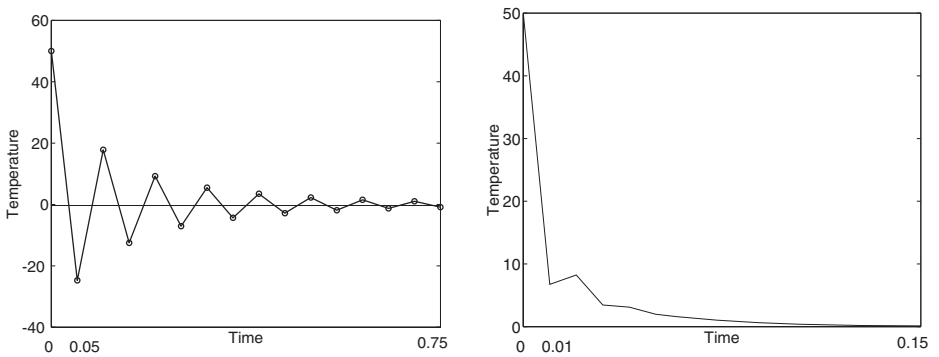


FIG. 7.11. *Approximations of the temperature at the point* $(5/10, 4/10, 4/10)$ *using the time-steps* $\Delta t = 0.05$ *and* $\Delta t = 0.01$.

first time-levels with $\Delta t = 0.5$ and $\Delta t = 0.01$ can be seen, respectively, on the left-hand and right-hand sides of Figure 7.10. In order to obtain a qualitatively adequate numerical approximation, it is enough to choose Δt to be not greater than 0.017 (Theorem 6.2).

7.3. Example in three dimensions. In order to give a 3D example, we solve the homogeneous heat equation on the unit cube. We apply the finite difference method with the Crank–Nicolson time-integration. The spatial discretization is performed with an equidistant mesh, where $\Delta x = \Delta y = \Delta z = 1/10$. The approximation of a continuous nonnegative initial function is zero in every grid point excepting 27 grid points in the middle of the region, where the temperature is approximated by 50. The approximations of the temperature at the point $(5/10, 4/10, 4/10)$ using the time-steps $\Delta t = 0.05$ and $\Delta t = 0.01$ can be seen in Figure 7.11. The time-step $\Delta t = 0.05$ results in negative temperatures, which contradicts the NP property. Theorem 6.2 gives a sufficient condition of the nonnegativity, namely, Δt must be not greater than $1/300$.

8. Conclusions. In this paper, we have shed light on the connections between the main qualitative properties of parabolic partial differential equations and their numerical solution methods. We have found that the MP, the NP, and the MNC are equivalent properties for the continuous problem, but this is not the case when we

consider the finite difference or Galerkin finite element solutions of the problem. In this case, the MP is equivalent to the NP, and the MNC yields a weaker condition as this property is implied by the properties mentioned earlier. Thus, to achieve a qualitatively adequate method, we have to apply an NP method, which can be guaranteed by the sufficient conditions of Theorem 6.1 and by its special cases listed in section 6.

Acknowledgments. The authors thank the referees for reading the manuscript thoroughly. All of their suggestions, especially the ones regarding the inclusion of demonstrative numerical tests, contributed to the improvement of the paper.

REFERENCES

[1] A. Berman and R. J. Plemmons, *Nonnegative Matrices in the Mathematical Sciences*, Academic Press, New York, 1979.

[2] V. S. Borisov, *On discrete maximum principles for linear equation systems and monotonicity of difference schemes*, SIAM J. Matrix Anal. Appl., 24 (2003), pp. 1110–1135.

[3] V. S. Borisov and S. Sorek, *On the monotonicity of difference schemes for computational physics*, SIAM J. Sci. Comput., 25 (2004), pp. 1557–1584.

[4] P. G. Ciarlet, *Discrete maximum principle for finite difference operators*, Aequationes Math., 4 (1970), pp. 338–352.

[5] P. G. Ciarlet and P. A. Raviart, *Maximum principle and uniform convergence for the finite element method*, Comput. Methods Appl. Mech. Engrg., 2 (1973), pp. 17–31.

[6] I. Faragó, *Nonnegativity of the difference schemes*, Pure Math. Appl., 6 (1996), pp. 38–56.

[7] I. Faragó and R. Horváth, *On the nonnegativity conservation of finite element solutions of parabolic problems*, in Proceedings of the Conference on Finite Element Methods: Three-Dimensional Problems, P. Neittaanmäki and M. Krizek, eds., GAKUTO Internat. Ser. Math. Sci. Appl. 15, Gakkōtosho, Tokyo, 2001, pp. 76–84.

[8] I. Faragó, R. Horváth, and S. Korotov, *Discrete maximum principle for linear parabolic problems solved on hybrid meshes*, Appl. Numer. Math., 53 (2005), pp. 249–264.

[9] I. Faragó, and T. Pfeil, *Preserving concavity in initial-boundary value problems of parabolic type and its numerical solution*, Period. Math. Hungar., 30 (1995), pp. 135–139.

[10] A. Friedmann, *Partial Differential Equations of Parabolic Type*, Prentice-Hall, Englewood Cliffs, NJ, 1964.

[11] H. Fujii, *Some remarks on finite element analysis of time-dependent field problems*, in Theory and Practice in Finite Element Structural Analysis, University of Tokyo Press, Tokyo, 1973, pp. 91–106.

[12] A. H. Hariton, *Some Qualitative Properties of the Numerical Solution to the Heat Conduction Equation*, Thesis for Cand. of Math. Science, Eötvös Loránd University, Budapest, 1995.

[13] R. Horváth, *Maximum norm contractivity in the numerical solution of the one-dimensional heat equation*, Appl. Numer. Math., 31 (1999), pp. 451–462.

[14] R. Horváth, *On the sign-stability of the numerical solution of the heat equation*, Pure Math. Appl., 11 (2000), pp. 281–291.

[15] R. Horváth, *Some integral properties of the heat equation*, Comput. Math. Appl., 42 (2001), pp. 1135–1141.

[16] R. Horváth, *On the monotonicity conservation in numerical solutions of the heat equation*, Appl. Numer. Math., 42 (2002), pp. 189–199.

[17] J. Karátson and S. Korotov, *Discrete maximum principles in finite element solutions of nonlinear problems with mixed boundary conditions*, Numer. Math., 99 (2005), pp. 669–698.

[18] S. Korotov, M. Krizek, and P. Neittaanmäki, *Weakened acute type condition for tetrahedral triangulations and the discrete maximum principle*, Math. Comp., 70 (2001), pp. 107–119.

[19] J. F. B. M. Kraaijevanger, *Maximum norm contractivity of discretization schemes for the heat equation*, Appl. Numer. Math., 9 (1992), pp. 475–492.

[20] O. A. Ladyženskaja, V. A. Solonnikov, and N. N. Ural'ceva, *Linear and Quasilinear Equations of Parabolic Type*, Transl. Math. Monogr. 23, AMS, Providence RI, 1967 (in Russian).

[21] J. Lorenz, *Zur Inverzmonotonie discreter probleme*, Numer. Math., 27 (1976/77), pp. 227–238.

[22] T. Pfeil, *On the monotonicity in time of the solutions of linear second order homogeneous parabolic equations*, Ann. Univ. Sci. Budapest. Eötvös Sect. Math., 36 (1993), pp. 139–146.

[23] V. Ruas Santos, *On the strong maximum principle for some piecewise linear finite element approximate problems of non-positive type*, J. Fac. Sci. Univ. Tokyo Sect. IA Math., 29 (1982), pp. 473–491.

[24] T. Vejchodsky, *On the nonnegativity conservation in semidiscrete parabolic problems*, in Conjugate Gradient Algorithms and Finite Element Methods, M. Krizek et al., eds., Springer-Verlag, Berlin, 2004, pp. 282–295.

SIAM J. SCI. COMPUT.
Vol. 28, No. 6, pp. 2337–2358
© 2006 Society for Industrial and Applied Mathematics

ITERATIVE VALIDATION OF EIGENSOLVERS: A SCHEME FOR IMPROVING THE RELIABILITY OF HERMITIAN EIGENVALUE SOLVERS*

JAMES R. McCOMBS† AND ANDREAS STATHOPOULOS†

Abstract. Iterative eigenvalue solvers for large, sparse matrices may miss some of the required eigenvalues that are of high algebraic multiplicity or tightly clustered. Block methods, locking, a posteriori validation, or simply increasing the required accuracy are often used to avoid missing or to detect a missed eigenvalue, but each has its own shortcomings in robustness or performance. To resolve these shortcomings, we have developed a postprocessing algorithm, *iterative validation of eigensolvers* (IVE), that combines the advantages of each technique. IVE detects numerically multiple eigenvalues among the approximate eigenvalues returned by a given solver, adjusts the block size accordingly, then calls the given solver using locking to compute a new approximation in the subspace orthogonal to the current approximate eigenvectors. This process is repeated until no additional missed eigenvalues can be identified. IVE is general and can be applied as a wrapper to any Rayleigh–Ritz-based, Hermitian eigensolver. Our experiments show that IVE is very effective in computing missed eigenvalues even with eigensolvers that lack locking or block capabilities, although such capabilities may further enhance robustness. By focusing on robustness in a postprocessing stage, IVE allows the user to decouple the notion of robustness from that of performance when choosing the block size or the convergence tolerance.

Key words. deflation, locking, block, Krylov methods, multiple eigenvalues

AMS subject classification. 65F15

DOI. 10.1137/050627617

1. Introduction.

Sparse, iterative eigenvalue solvers are effective at computing the extremal eigenvalues of large, Hermitian matrices or matrices represented as functions [17, 7, 28, 27]. However, iterative methods may miss some of the desired eigenvalues if these are of high algebraic multiplicity or are tightly clustered. Moreover, certain initial guesses, eigenvalue distributions, and preconditioning can cause eigenvalues to converge out of the expected order and thus to be missed. Some techniques have been developed to improve robustness, i.e., to avoid missing eigenvalues or to detect any missed ones, but a robust and automated process that combines the advantages of each technique has yet to be investigated.

A few a posteriori techniques have been used to deal with missed eigenpairs. Sylvester's matrix inertia [11] is one technique that can be used to determine how many eigenvalues of a Hermitian matrix exist within a particular interval. Although this technique has enabled the development of robust eigenvalue software [13], it requires the LDL^T factorization of the matrix, which can be prohibitively expensive for large, sparse matrices, or even impossible if the matrix is represented as a function. In such cases, practitioners may call the solver again with a random initial guess to obtain an additional eigenvector in the orthogonal complement of the current set of converged approximations, hopefully identifying a missed eigenpair.

*Received by the editors March 24, 2005; accepted for publication (in revised form) July 31, 2006; published electronically December 21, 2006. This work was supported by National Science Foundation grants ITR/DMR 0325218 and ITR/AP-0112727 and performed using the computational facility at the College of William and Mary, which was enabled by National Science Foundation grant EIA-9977030 and Sun Microsystems grant SAR EDU00-03-793.

http://www.siam.org/journals/sisc/28-6/62761.htm

†Department of Computer Science, College of William and Mary, Williamsburg, VA 23187-8795 (mccombjr@cs.wm.edu, andreas@cs.wm.edu).

Locking and block iterations are two techniques that attempt to reduce the number of eigenvalues missed *during* the iterative process. Locking [4, 31, 18] is performed by freezing converged eigenvector approximations (locked vectors) and keeping current and future vectors in the search space orthogonal to them. The search space can then approximate other eigenpairs without duplicating the locked ones. Alternatively, solvers may be implemented in a nonlocking way [19, 33], where converged eigenvectors keep participating in the search space, thereby improving their accuracy. Regardless of implementation, locking a set of a priori known eigenvectors can be performed outside a solver. Locking can be effective in obtaining more than one eigenvector belonging to a multiple eigenvalue, especially with Krylov methods. It also allows for a smaller active search space, which reduces iteration costs, and this has made it popular with with many preconditioned or non-Krylov solvers with more expensive iterations [4, 29]. Whether locking delivers the purported robustness depends significantly on the implementation details, and typically the most robust implementation may not be the most efficient.

Block iterative methods [27, 3, 26, 37, 25, 36, 13] that extend the subspace by k vectors per iteration, where k is the block size, are a more robust alternative to locking. With appropriate initial guesses, the block Lanczos method with block size k will find all k eigenvectors of a multiple eigenvalue [24]. Although not guaranteed in general, this property is observed in other methods as well. Block methods can also result in better cache utilization and improved performance if a fine-tuned implementation is available. However, block methods are not without drawbacks. Excessively large block sizes can increase the number of matrix-vector multiplications and result in more frequent restarts, thereby slowing convergence.

Locking and blocking can be combined to improve the robustness of iterative eigensolvers [5, 10, 2, 20]. Still, the resulting robustness depends on the problem solved and the parameters chosen by the user. What makes this choice difficult is the conflicting requirements between robustness and performance. A lower than required convergence threshold is less likely to miss eigenvalues but it takes more time. Moreover, the choice of a block size that optimizes simultaneously cache performance, convergence, and robustness is not possible to know a priori. Our approach in this paper is to let the user choose the most efficient parameters as default, and to provide a postprocessing technique that corrects any robustness shortcomings, and only when these are needed. This way, we decouple the issue of robustness from that of performance (cache or convergence).

Our goal is not to provide another Hermitian eigensolver but to develop a robust detection algorithm that uses any given solver in a methodical way to identify whether the requirements posed by the user have been met by the solver. Borrowing a term from software engineering, we call our algorithm *iterative validation of eigensolvers* (IVE), as it does not verify whether the approximations are correct but rather adjusts the solver to find the correct approximations. IVE works in conjunction with any Rayleigh–Ritz-based Hermitian eigenvalue solver. The IVE algorithm accepts as input and locks the converged eigenvector approximations. If a block solver is available, IVE adjusts the block size based on the largest numerical multiplicity found and calls the solver again so that any missed eigenvalues in the orthogonal complement of the locked vectors can be computed. Any missed eigenvalues detected by IVE are locked out and the process is repeated until no additional missed eigenvalues are computed. Although the structure of the IVE algorithm is general, we apply it only to Hermitian eigenproblems where the monotonic convergence of eigenvalues facilitates a robust detection of missed eigenvalues.

We have used our IVE to wrap several well-known eigensolvers. The solvers may differ substantially in their ability to use locking or blocking efficiently and robustly, but they invariably miss eigenvalues. Our results show that IVE restores robustness in several pathologically hard cases. Moreover, running the solver through default parameters followed by a certain IVE configuration improves performance, thus demonstrating the desired decoupling of performance and robustness.

In section 2, we discuss why eigensolvers miss eigenvalues and the available techniques for increasing robustness. In section 3, we describe in detail the IVE algorithm. In section 4, we present results using IVE on a variety of solvers. Throughout the paper we assume that we seek the lowest nev eigenvalues $\tilde{\lambda}_i$ and eigenvectors $\tilde{x}_i$, $i = 1, \ldots, nev$, of a sparse, Hermitian matrix A.

2. Reasons behind missing eigenvalues and current approaches. A large arsenal of sparse Hermitian eigenvalue solvers is available, some based on Krylov methods, others on nonlinear optimization, preconditioning, and a host of other underlying principles [32]. Most of these methods employ the Rayleigh–Ritz (RR) minimization to extract their approximations from a search space. The eigenvalue and eigenvector approximations are called Ritz values and Ritz vectors, respectively. For Hermitian eigenproblems, the Ritz values minimize the Rayleigh quotient over the search space, thus leading to a monotonic convergence toward the required eigenvalues.

Despite their individual strengths, solvers can miss some of the required extreme eigenvalues. Three, often interrelated, reasons are (a) the presence of multiple or highly clustered eigenvalues, (b) the out-of-order convergence of some eigenvalues, and (c) a starting vector that is (almost) deficient in certain required eigenvector components.

It is well known that in exact arithmetic Krylov methods cannot compute more than one eigenvector belonging to a multiple eigenvalue. Krylov methods produce the next vector in the search space as $r = p(A)v_0$, where v_0 is the starting vector and $p(A)$ is the characteristic polynomial of the Lanczos tridiagonal matrix [27]. If a multiple eigenvalue λ_i and one of its corresponding eigenvectors have already been found, then $p(\lambda_i) = 0$ and r contains no components in the kernel: $\mathrm{Null}(A - \lambda_i I)$. In practice, floating-point arithmetic introduces noise toward these directions, allowing their eventual computation. To allow the solver enough time to amplify the missing eigencomponents, a low convergence tolerance must be required. Otherwise, a more interior eigenpair may converge first as one of the nev smallest ones. A similar situation may occur with non-Krylov methods, as these use the (possibly preconditioned) residual to expand the search space. When the Rayleigh quotient in the residual is an exact multiple eigenvalue, the residual is devoid of kernel components. Although preconditioning may reintroduce some kernel components, more interior eigenpairs may already be close to convergence, causing multiple eigenvalues to be missed.

Clustered eigenvalues present similar difficulties as multiple ones. This is not surprising because, until the solver has resolved an eigenvalue to an error that is smaller than its distance from a nearby eigenvalue, it views both eigenvalues as multiple and could miss one if larger tolerances are specified. The slow convergence of solvers toward clustered eigenvalues exacerbates the problem.

More generally, eigenvalues are missed when an interior eigenvalue converges before the solver has identified the existence of an outer one. In Krylov methods this is not the expected order of convergence, causing robustness problems to solvers that are based on the assumption that the nev smallest eigenvalues will be found first. The assumption, although usually true, does not hold in general. As shown recently [16],

eigenvalue distributions exist for which Krylov methods converge to the eigenvalues in the middle of the spectrum first and to the extreme ones last. In practice, smaller order reversals may be observed, which can cause eigenvalues to be missed.

The out-of-order convergence is more frequently observed with non-Krylov methods that use preconditioning or solve approximately a correction equation to speed up convergence [7, 29, 23]. For example, consider a matrix A, whose absolute smallest eigenvalues are required, and the generalized Davidson method using a preconditioner $M \approx A^{-1}$. Because M and A do not share the same eigenvectors, the convergence benefits from M are not necessarily greater for eigenvalues closest to zero, thus yielding an arbitrary convergence order. For methods that solve a correction equation at every step the choice of shift in the equation strongly biases the order in which eigenvalues converge. Moreover, fast convergence implies that even small biases of the starting vector toward interior eigendirections may lead to misconvergence. Davidson-type methods also use various targeting schemes [33], i.e., which Ritz vector is improved at every step, directly affecting the order in which eigenvalues converge. In floating point arithmetic, targeting effects are observed even without preconditioning [22, 35].

2.1. Current approaches. To increase confidence in the computed eigenpairs, practitioners typically ask for more accuracy than needed. Although beneficial in many cases, in general we may not know the accuracy that is sufficient for resolving multiple eigenvalues or avoiding out-of-order convergence. Also, this approach may be unnecessarily expensive, especially when a very low tolerance is used to find a large number of eigenvalues of which only a few would have been missed with a larger tolerance. Finally, the problem with higher multiplicities in Krylov methods persists.

Another common strategy is to compute more eigenpairs than required. This works if the reversals in the order of convergence are localized, e.g., the tenth smallest eigenvalue converges before the ninth, but not before the second. In general, we do not know how many more eigenpairs to compute, so the strategy can be unnecessarily expensive. Most important, it still may not find multiple or clustered eigenpairs.

Deflation of converged eigenpairs is commonly used in many eigensolvers so that more eigenpairs can be obtained without repeating the already computed ones. Locking is the preferred form of deflation because of its numerical stability [27, 4]. When an eigenvector converges, it is extracted from the search space and all current and future vectors of the search space are kept orthogonal to that (locked) vector. Locked vectors are inactive otherwise, they do not participate in the RR (as future Ritz vectors should not have any components in them), and therefore they are not modified further. Computationally, the smaller active search space reduces the costs of the RR and restarting phases of eigensolvers, more so when a large number of eigenpairs are sought. This is one of the reasons that locking is preferred to non-locking implementations for subspace iteration and Jacobi–Davidson-type methods [5, 29].

Locking provides also a more reliable means of computing the invariant spaces of multiple eigenvalues, especially with Krylov methods. Assume that a subset X of the invariant subspace associated with λ has been computed and locked. This is equivalent to calling the solver with the deflated operator $A|_{X^\perp}$, in which all the eigenvalues associated with X are absent. Unlike the polynomial deflation $p(A)v_0$ in Krylov methods which removes all the invariant subspace of λ from the vector iterates, locking removes only the computed part of $\mathrm{Null}(A - \lambda I)$. More eigenvectors of λ can then be obtained. This deflation behavior is not restricted only to Krylov methods. Finally, locking also speeds up the convergence to any missed, single eigenvalues which are

amidst other converged ones. In the deflated operator such eigenvalues are better isolated and therefore are easier to find. However, even with locking, misconvergence can still occur if some tightly clustered or multiple eigenpairs converge out of order. Moreover, not all solvers implement locking, and when they do, their implementation details and thus robustness may vary. For example, locking can be implemented transparently to the solver if we use the deflated operator in the matrix-vector multiplication, but the solver must return the converged eigenpairs before they are locked. Alternatively, some solvers implement locking as part of the orthogonalization phase. We discuss these issues further in the next section.

Block methods are usually more effective in computing multiple eigenvalues. As long as the k initial guesses in the block contain sufficient components in the direction of the eigenvectors associated with the multiple eigenvalue, then block Krylov methods are guaranteed to find at least k of the multiple values [24]. Moreover, more uniform convergence of the block vectors is observed, reducing the likelihood of out-of-order misconvergence. Blocking is known to improve robustness in all iterative methods, but the effects are more emphasized in Krylov methods [12]. Block methods also improve cache performance with larger k [8], seemingly offering a panacea to all problems.

Despite the better cache utilization, robustness comes at a cost in execution time. Although the number of iterations decreases, the overall number of matrix-vector operations (and thus flops) performed is usually higher than their single-vector counterparts [14]. Especially with spectra that are easily computed, block methods tend to require close to k times more matrix-vector operations than single-vector methods, completely canceling any caching benefits. Additionally, the number of iterations may also increase if k is too large relative to the maximum basis size. This would cause very frequent restarts impairing the convergence of most methods; this is apparent in some experiments later in this paper. When k equals the maximum basis size, the solver reverts to subspace iteration, losing the subspace acceleration [32]. In general, an optimal choice for k must ensure appropriate cache utilization, robustness, and the smallest possible increase in flops. Except for caching which depends on the architecture, none of the other variables are known ahead of time. In particular, the k needed to detect all the required eigenvectors is not known, and even if it were, it could conflict with the optimal choice for cache performance and convergence. Finally, unlike locking, blocking can be used only if implemented by the available solver.

We conclude our discussion with the importance of the initial guess. In all the aforementioned techniques, a critical assumption is that the initial vector has sufficient components in the desired directions. In lack of better information, a random vector is typically the best initial guess. Yet, as the iterative process progresses, certain required directions may be diminished or removed as in the case of Krylov methods and multiple eigenvalues. A possibility is to insert a random vector in the search space or in the block, replacing an eigenpair that converges. This can be performed trivially in (Jacobi–)Davidson or subspace iteration methods, but techniques have also been proposed for the implicitly restarted Lanczos method [31, 30]. Still, the newly introduced directions may not be amplified fast enough to prevent out-of-order convergence of an interior eigenpair which was almost converged before the insertion of the random vector. A drastic solution would be to dispense with the whole search space after an eigenpair converges and restart the solver anew with a random guess. A related approach that does not completely get rid of the search space is suggested in [21]. Based on our discussion on locking, a newly rebuilt search space would offer the best possible robustness. However, it would also be very slow, especially if no eigenvalues were going to be missed.

A recurring theme is the dichotomy between robustness and performance. To achieve robustness, solvers would have to make extreme choices on parameters such as block size and tolerance that may impair performance. In the following section we present our postprocessing approach to validating the parameters of a given solver that are required to achieve robustness.

3. Iterative validation of eigensolvers. Our proposed IVE algorithm is implemented as a wrapper around any existing sparse, iterative, Hermitian eigensolver that utilizes the RR minimization procedure. IVE automates the process of computing eigenvalues missed by the solver, by combining as many of the techniques mentioned in section 2.1 as the solver implements. In other words, IVE is not a solver, nor does it compute eigenvalues. IVE merely identifies difficulties in the computed spectrum and sets locked vectors, initial guesses, and block size appropriately so that the chosen eigensolver may compute any missed eigenvalues on its own. In this sense, it is similar to the validation process of a software engineering cycle, where a software is modified appropriately (different solver parameters) to meet the customer requirements (which eigenpairs to find), rather than verifying whether the software meets the specifications (produces converged eigenpairs). Therefore, the measures of performance and robustness achieved by IVE can only be in reference to the underlying solver. Also, by being a postprocessing algorithm, IVE may be switched off if no validation is needed.

The enabling property for IVE is that the Ritz values of RR-based Hermitian eigensolvers converge monotonically, if their Ritz vectors are retained in the search space during restarts or truncations. Once a Ritz value becomes smaller than a converged eigenvalue, we are sure that a previously undetected eigenvalue exists. Although Newton eigensolver schemes can be derived without RR, most current algorithms and software incorporate it, so our assumption is not restrictive in practice.

Assume the solver has returned an initial set of nev converged Ritz pairs (λ_i, x_i), where $\lambda_i \leq \lambda_{i+1}$, $i = 1, \ldots, nev - 1$. The IVE algorithm analyzes the converged λ_i and the norms of their residuals, $r_i = Ax_i - \lambda_i x_i$, to determine an appropriate block size. IVE then calls the solver with new initial guesses, using the converged $X = \{x_i\}$ as locked vectors, in an attempt to find any missed eigenvalues. An eigenvalue is a missed eigenvalue if it is smaller than the largest locked eigenvalue λ_{nev}. If it is, then its eigenvector replaces the locked eigenvector x_{nev} in the set of locked vectors. The IVE algorithm then repeats an analysis of the block size and proceeds as before. IVE works also with solvers that do not provide locking or blocking. First, we discuss some issues on locking and how to choose a block size. Then we present the IVE algorithm, describing how it chooses initial guesses and how many additional eigenpairs it finds.

3.1. Locking variants. One way to compute eigenvectors orthogonal to the set X is for the solver to provide a mechanism to lock (orthogonalize against) a set of externally provided vectors [20, 1]. This feature is a relatively simple extension to the orthogonalization procedure that most solvers employ. It is also numerically stable because the orthogonalization procedure should guarantee that no components of X appear in the search space. In this case, IVE simply passes X to the solver. Note that a solver that locks against an externally provided X does not have to implement locking internally for the eigenpairs it computes.

When a solver does not provide the above external locking feature, an alternative is to provide all the converged X together with a random vector as initial guesses to the solver and ask for a few more eigenpairs. This works with many solvers such as (Jacobi–)Davidson, subspace iteration, or methods where X can be provided as part

of the initial block, but it does not work with Lanczos algorithms. In addition, the solver must accommodate a large enough search space, and all X eigenvectors have to be rechecked and possibly modified.

A third alternative is to use explicit deflation, by calling the solver not with the matrix A, but with a function that implements the action of the operator:

$$(I - XX^T)(A - \sigma I). \tag{3.1}$$

The preconditioner may be deflated in a similar way as shown in [28]. The shift σ plays an important role. If $\sigma = 0$ and the smallest eigenvalues of A are positive, the eigenvalues of (3.1) associated with X are zero and thus they are still the smallest. In exact arithmetic this is equivalent to the external locking above, and no X components should appear in the search space. In floating point, however, when new vectors are orthogonalized against the search space inside any solver, they may lose their orthogonality against X. As X is not available within the solver, orthogonality cannot be recovered through reorthogonalization. Thus, components of X will start to emerge in the RR, and their zero eigenvalues will be recomputed.

The role of σ is to shift the spectrum so that zero is far from the smallest eigenvalues, ideally in the middle or at the other end of the spectrum. Then, the X components that emerge in the search space will be in the unwanted part of the spectrum and they will not be chosen by RR (purged internally [18]). When explicit locking is necessary, we have used $\sigma = \|A\|_F/\sqrt{n}$ as a lower bound to the largest eigenvalue of positive spectra, where $\|A\|_F$ is the Frobenius norm and n the dimension of A. Other shifts can also be used, if estimates of λ_n are known [27]. This technique yields significant autonomy to IVE, not depending on the locking features of the solver.

3.2. Determining block sizes. Assume that a block eigensolver is available. Our theoretical premise is that a block size equal to the largest multiplicity should be sufficient to resolve any problems. The IVE algorithm selects such a block size based on the largest "numerical multiplicity" observed in the converged Ritz pairs.

Let δ_i be an error bound and $[\lambda_i - \delta_i, \lambda_i]$ the error interval corresponding to the Ritz value λ_i. Because the matrix is Hermitian, $\tilde{\lambda}_i \leq \lambda_i$. Depending on the eigenvalue distribution and on the size of the user-specified tolerance, the error intervals associated with the Ritz values may overlap. Two Ritz values whose error intervals overlap are *numerically multiple*. This means that we cannot distinguish between the two with certainty at this accuracy level. It is possible for three or more Ritz values to be numerically multiple, but this requires a more general definition. The Ritz values $\{\lambda_i, i \in \{i_1, i_2, \ldots, i_k\}\}$ are *numerically multiple* if and only if $[\lambda_i - \delta_i, \lambda_i]$ overlaps with $[\lambda_j - \delta_j, \lambda_j]$ for all $i, j \in \{i_1, i_2, \ldots, i_k\}$.

Sharp bounds on the δ_i are well known when the approximations are computed through the RR procedure [24]. Let $\tilde{\lambda}$ be the single eigenvalue closest to a Ritz value λ, and let r be the residual corresponding to λ. The following inequality holds:

$$|\tilde{\lambda} - \lambda| \leq \delta = \min\left(\|r\|_2, \frac{\|r\|_2^2}{\min\left(|\tilde{\lambda}_i - \lambda|,\ \tilde{\lambda}_i \neq \tilde{\lambda}\right)}\right). \tag{3.2}$$

At convergence (after the solver returns), the upper bound δ in (3.2) can be estimated by substituting the Ritz value closest to λ for $\tilde{\lambda}_i$. At that time, this is the best available estimate. Note that the bound can be influenced by the inaccuracy in the Ritz values only if the estimated gap is larger than $\|r\| \leq Tol$. However, in that case

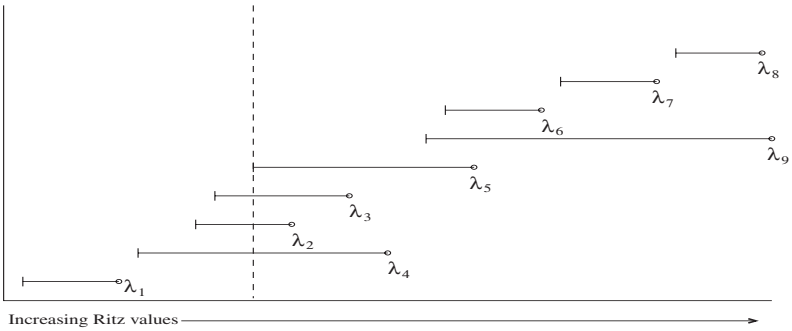


FIG. 3.1. *Sample distribution of converged Ritz values and their associated error intervals. The maximum algebraic numerical multiplicity in this example is four because a point of maximum overlap, indicated by the dashed vertical line, crosses four error intervals. The rest of the eigenvalues have multiplicities* $\lambda_1(1), \lambda_2(4), \lambda_3(4), \lambda_4(4), \lambda_5(4), \lambda_6(3), \lambda_7(2), \lambda_8(2), \lambda_9(3)$.

the computed Ritz values are separated by more than $\|r\|$ and therefore cannot be numerically multiple.

Let the *algebraic numerical multiplicity*, m_i, of a Ritz value λ_i be defined as the size of the largest set of Ritz values that include λ_i and are numerically multiple. Thus, an algebraic numerical multiplicity is an upper bound to the algebraic multiplicity of an eigenvalue, and it becomes equal to it when the $\delta_i \to 0$.

A robust choice for block size can be the maximum m_i taken over all Ritz values. The rationale is that other required eigenvalues may have been missed that have the same multiplicity. Although this works well for small m_i, for large m_i it is highly uncommon that the eigenvalue with the maximum m_i is still missing half its actual multiplicity, making this a pessimistic choice. In addition, the block size should not become too large relative to the maximum size of the search space, depending on the solver. For example, for LOBPCG [15] the block size must be a third of the basis size; Lanczos and Davidson solvers work better when the block size is much smaller than the basis size. Because of these external to IVE factors, or because some information about the problem may be known (multiplicity or cluster size), the user may put a maximum cap on the block size. We emphasize that a cap based on the underlying solver and its maximum basis size does not reduce robustness, because for larger block sizes that solver's own robustness and convergence would deteriorate. Also, blocking is just one of the robustness techniques that IVE employs.

Computing the maximum algebraic numerical multiplicity is equivalent to finding a "point of maximum overlap," i.e., a point that has the largest number of error intervals overlapping it. It can be shown that there will always be a point of maximum overlap at one of the endpoints of the intervals [6]. Figure 3.1 shows an example distribution of Ritz values and their associated error intervals. Figure 3.2 gives an algorithm, MaxAlgNumMult, for computing the maximum m_i in $O(nev^2)$. Interestingly, the p_i in the algorithm do not correspond to the m_i for all i, yet the maximum m_i is computed correctly. A more efficient, $O(nev \log_2 nev)$ time algorithm exists, but its implementation is involved. Moreover, as $nev \ll n$ and RR already requires $O(nev^3)$ time, the execution time of our algorithm is negligible, obviating the use of the lower complexity algorithm. To our knowledge, this is the first algorithm to find the maximum multiplicity in a set of approximate eigenpairs.

3.3. The IVE algorithm. The IVE algorithm given in Figure 3.3 first computes the residual norms of the converged Ritz vectors (step 4). This step is not required

ALGORITHM. MaxAlgNumMult(λ, r, nev).
1. Compute δ_i the error estimate from (3.2) corresponding to λ_i.
2. MaxAlgNumMult = 0
3. lowerBound(i) = $\lambda_i - \delta_i$, $i = 1, \ldots, nev$
   Let p_i be the number of overlapping intervals at endpoint $\lambda_i - \delta_i$
4. **for** $i = 1, nev$
5. $p_i = 0$
6. **for** $j = 1, nev$
7. **if** (lowerBound(i) $\geq$ lowerBound(j) **and** lowerBound(i) $\leq \lambda_j$)
8. $p_i = p_i + 1$
9. **end**
10. MaxAlgNumMult = **max**(MaxAlgNumMult, p_i)
11. **end**

FIG. 3.2. *Algorithm for computing the maximum algebraic numerical multiplicity among a set of Ritz values corresponding to the smallest eigenvalues of a Hermitian matrix.*

ALGORITHM. [X, λ] = IVE(A, X, λ, resNorms, initialBlockSize, maxBasisSize).
Let X and λ be the converged Ritz vectors and Ritz values.
Let initialBlockSize be the block size used to compute X.
Let maxBlockSize be the maximum block size allowed
1. numNew = 1
2. X_{missed} = [ ], $X_{unconverged}$ = [ ]
3. **repeat**
4. resNorms = ComputeResNorms(A, X, λ)
5. newMult = MaxAlgNumMult(λ, resNorms)
6. **if** (solver implements block)
   blockSize = **min**(maxBlockSize, **max**(newMult, initialBlockSize, 2))
   else
   blockSize = 1
7. X_0 = [X_{missed}, **rand**(blockSize - **size**($X_{unconverged}$))]
8. **if** (solver implements external locking)
   [$X_{new}, \lambda_{new}, X_{unconverged}, \lambda_{unconverged}$] = Solver($A$, numNew, X_0, X)
   else
   [$X_{new}, \lambda_{new}, X_{unconverged}, \lambda_{unconverged}$] =
   Solver($(I - XX^T)(A - \sigma I)$, numNew, X_0), as described in section 3.1
9. [X, λ] = InsertionSort(X_{new}, λ_{new}, numNew, X, λ, resNorms)
10. numMissedUnconv = NumMissedValues(λ, $\lambda_{unconverged}$, blockSize-numNew)
11. X_{missed} = $X_{unconverged}$(:,1:numMissedUnconv)
12. numNew = **max**(1, numMissedUnconv)
13. **until** (X did not change **and** numMissedUnconv = 0)

FIG. 3.3. *IVE algorithm for computing missed eigenpairs. IVE is a wrapper around an existing Hermitian, Rayleigh–Ritz-based eigensolver.*

if the eigensolver returns this information. Then, IVE determines the maximum algebraic numerical multiplicity using MaxAlgNumMult (step 5). In step 6, if the solver implements a block method, the block size is chosen so that it does not increase beyond the maximum value maxBlockSize. Below that value, it is set at the maximum

ALGORITHM. $[X, \lambda]$ = InsertionSort(X_{new}, λ_{new}, numNew, X, λ, resNorms, nev).
1. **for** $i = 1$:numNew
2. $j = nev$
3. **while** $(j > 0$ **and** $\lambda(j) > \lambda_{new}(i))$, $j = j - 1$; **end**
4. **if** $j == nev$, **break**;
5. $X = [X(:, 1 : j),\ X_{new}(:, i),\ X(:, j + 1 : nev - 1)]$
6. $\lambda = [\lambda(1 : j),\ \lambda(i),\ \lambda(j + 1 : nev - 1)]$
7. $r_{new} \equiv$ the residual with respect to $(X_{new}(i),\ \lambda(i))$
8. $resNorms = [resNorms(1 : j),\ r_{new},\ resNorms(j + 1 : nev - 1)]$
9. **end**

FIG. 3.4. *Insertion sort algorithm for identifying and inserting missed eigenvalues into λ. Handling of boundary cases is not shown. In actual implementations, the eigenvalues are inserted first, and the eigenvectors are inserted only at the end to avoid unnecessary memory copying.*

of the current maximum m_i, the block size chosen by the user to compute the initial set of converged Ritz values, and two. A block size of at least two is used to force a block method when no numerically multiple eigenvalues have been detected. If the initial run of the solver used a block size of k, there is no incentive to use anything else for validation. Still, the user can overwrite it by passing initialBlockSize = 1. In step 7, the initial guesses, X_0, are chosen. In the first IVE iteration, X_0 is a set of all random initial guesses. In subsequent IVE iterations, X_0 includes also any unconverged Ritz vectors that are known to be missed during the previous IVE iteration (see below). As we discussed in section 2.1, it is always important to include some random guesses.

In step 8, the eigensolver is called with the initial guesses X_0 and locked vectors X to find numNew more eigenpairs. Depending on the solver, external or explicit locking can be performed. Note that the IVE algorithm assumes that the chosen solver returns not only the numNew converged Ritz values in λ_{new} but also the remaining blockSize−numNew unconverged Ritz values in $\lambda_{unconverged}$. Solvers that do not return the unconverged Ritz pairs by default can be easily modified to do so, if the source code is available. Otherwise, we can set numMissedUnconv = 0.

In step 9, an insertion sort (see Figure 3.4 for details) is called to insert any converged, missed eigenpairs in λ_{new} and X_{new} into λ and X, respectively. For every missed Ritz value inserted into the λ array, the largest λ_{nev} in the array and the corresponding vector in X are removed. If enough storage is available, those removed converged eigenvectors can still remain in the locked array to avoid possible recomputation. We choose not to take advantage of this feature in our experiments. If none of the recently converged Ritz values were missed, then the IVE algorithm will terminate at step 13.

Step 10 calls the algorithm NumMissedValues in Figure 3.5 to determine how many of the unconverged Ritz values in $\lambda_{unconverged}$ are missed values. A Ritz value in $\lambda_{unconverged}$ is considered missed if it is smaller than λ_{nev}. Because of monotonic convergence, these Ritz values will only continue to decrease and approach missed eigenvalues. Therefore, it is wise to use the corresponding unconverged Ritz vectors in $X_{unconverged}$ as initial guesses (step 11). These can significantly reduce validation time when many eigenvalues are missed, because they enable the solver to avoid repeating convergence through random guesses.

Finally, step 12 sets the new number of Ritz values the solver must compute. Clearly, we should continue and find the numMissedUnconv missed ones that step 10 may have identified. If numMissedUnconv = 0, we have no indication that more

ALGORITHM. NumMissedValues(λ, nev, $\lambda_{unconverged}$, numUconv).

```
1.  NumMissedValues = 0
2.  for i = 1, numUconv
3.      j = nev
4.      while (j > 0 and  λ(j) < λ_unconverged(i)) , j = j − 1 ;  end
5.      if j == nev,  break ;
6.      λ = [λ(1 : j), λ_unconverged(i), λ(j + 1 : nev − 1)]
7.      NumMissedValues = NumMissedValues + 1
8.  end
```

FIG. 3.5. *Algorithm for computing the number of unconverged Ritz values that have been skipped. The corresponding Ritz vectors will be used as initial guesses at the next IVE iteration.*

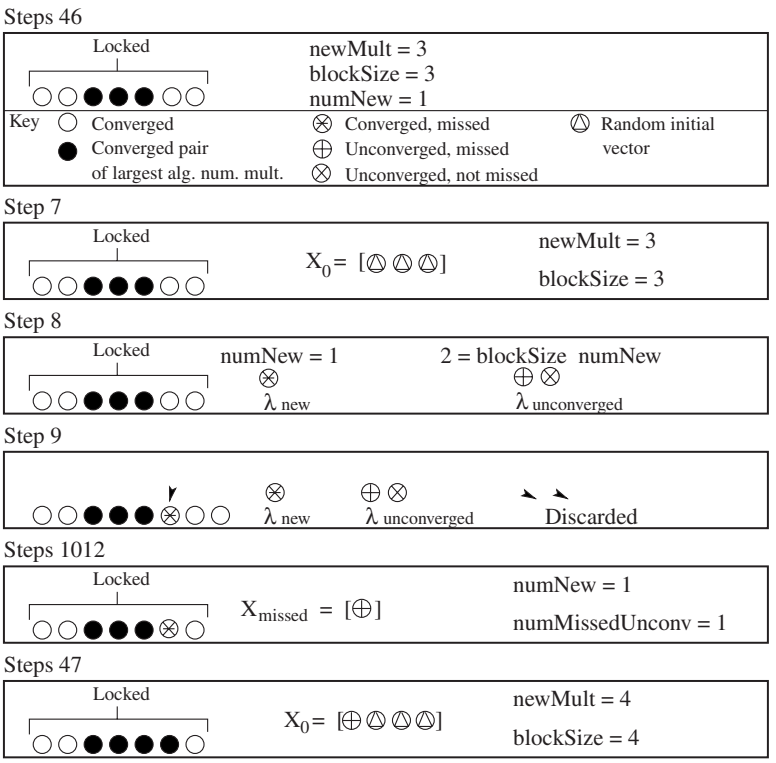


FIG. 3.6. *Example validation problem.*

eigenvalues may be missing, so it would be wasteful to find more than one additional eigenvalue. The IVE algorithm terminates in step 13 if no missed eigenvalues (converged or unconverged) were discovered, or it continues with the next IVE iteration.

We present an example in Figure 3.6 to illustrate the steps taken by the iterative validation algorithm. Suppose we seek the $nev = 8$ smallest eigenvalues of a symmetric matrix. In steps 4 through 6 the residual norms are computed and the maximum algebraic numerical multiplicity is determined to be 3. All random initial guesses are selected in step 7 because it is still the first iteration. In step 8 the solver is called and returns numNew = 1 converged Ritz value in λ_{new}, and two other Ritz values in

$\lambda_{unconverged}$, because the block size is three. Next, InsertionSort is called in step 9 to determine if λ_{new} is a missed eigenvalue. It is, so it is inserted into its proper position and the largest element in λ is discarded. In steps 10 through 12, NumMissedValues determines that one of the $\lambda_{unconverged}$ was converging toward a missed eigenvalue. The corresponding missed vector is placed in X_{missed} to be used as an initial guess in the next IVE iteration, and numNew is set to 1 to indicate that an additional eigenvalue has been skipped. At the next IVE iteration, steps 4 through 7 determine the algebraic numerical multiplicity to be 4, set the block size accordingly, and assign X_{missed} and three random vectors as initial guesses. The IVE iterations continue until no more missed eigenvalues are detected.

4. Experimental evaluation. Our primary goal is to show that IVE restores robustness in any Hermitian eigensolver, even for pathological cases with large multiplicities, or tightly clustered eigenvalues. The importance of increased confidence in the computed results, sometimes at any cost, cannot be overstated. Of particular interest are cases of eigensolvers that fail to find all the required eigenvalues, regardless of the initial choice of parameters, yet guided by IVE, they succeed. A secondary goal is to demonstrate how IVE can decouple robustness from performance, so that solvers are run with default settings for best performance and any missed eigenpairs are obtained later in a shorter IVE cycle.

4.1. Eigensolvers used in the evaluation. We have used five different eigensolvers from three eigensolver packages. The first is IRBL, the implicitly restarted Block Lanczos method [2, 1], which uses implicit restarting combined with Leja shifts to improve convergence of the standard Lanczos method. IRBL is a block method and implements both external and internal locking. For our tests, we have added three lines of code in IRBL to allow it to return all Ritz pairs in the block, not only the converged ones. IRBL is implemented in MATLAB, and therefore it is used mainly to demonstrate the IVE benefits on robustness and convergence; not on timings.

The second symmetric solver is the function dsaupd from the popular software package ARPACK [19]. Henceforth, we will refer to the particular eigensolver as ARPACK. ARPACK is based also on implicit restarting, but unlike IRBL, it uses the Ritz values as the restarting shifts. ARPACK is a single vector method and does not implement external or internal locking. Thus, in a sense, it represents a worst-case scenario, as IVE cannot employ all its techniques. In particular, IVE is restricted to using blockSize = 1, numMissedUnconv = 0, and thus numNew = 1, and locking must be implemented explicitly in IVE. No modifications were made to the ARPACK code which is implemented in Fortran 77.

The remaining three solvers come from PRIMME, a software package that we have developed in C and which is freely available [20]. PRIMME (PReconditioned Iterative MultiMethod Eigensolver) is based on a Davidson-type main iteration, but it implements various techniques such as blocking, external and internal locking, CG-type restarting, and adaptive inner-outer iterations that allow it to transform to any current eigenvalue method. We choose three methods for their popularity and efficiency.

The first is the generalized Davidson method with CG-restarting, or GD+k [34]. We use it with block and locking enabled. Its main characteristic is the near optimal convergence in terms of matrix-vector products for small nev. But this also means that there is little room for convergence improvement through larger block sizes. Therefore, for one of the easier matrices, we also use the original generalized Davidson without CG-restarting [23].

```
N = 30000; clusters = 30; multiplicity = 8; DeltaCluster = 1e-8;
D = zeros(N,1); Ones = ones(multiplicity,1);
D(1:multiplicity) = eps*Ones; value = 1.e-6;
for i = 1:clusters-1
   D(i*multiplicity+1:(i+1)*multiplicity) = value*Ones;
   value = value+DeltaCluster;
end
% The rest well separated and equidistant
row = clusters*multiplicity+1; separation = (1-1e-3)/(N-row);
D(row:N) = 1e-3+ (row-1:N-1)*separation;
A = spdiags(D,0,N,N);
```

FIG. 4.1. *The Matlab code that generates the diagonal matrix DIAG.*

The second method is the JDQMR variant of the Jacobi–Davidson method [32, 29]. It is used with block and locking options enabled, although the correction equation corresponding to each block vector is solved independently. JDQMR adaptively stops the inner iteration for each correction equation, yielding convergence near that of GD+k, but the per-iteration cost of JDQMR is much less expensive.

The third method is LOBPCG-W, which is a variant of the increasingly popular LOBPCG method [15], and it differs from it in the following two ways. First, LOBPCG-W maintains an orthogonal search space to guarantee numerical stability of the process. Second, LOBPCG-W uses a window approach, i.e., the blockSize is allowed to be smaller than *nev*, and as eigenvalues are found in the block they are locked out, and the window progresses until all *nev* pairs have been found.

4.2. Experimental setup and environment. We perform experiments on three matrices, each highlighting a different aspect of the IVE capabilities. The first matrix, BCSSTK16 from the Harwell–Boeing collection [9], has an eigenvalue with unusually high multiplicity. The second matrix, DIAG, is diagonal and is constructed as shown in Figure 4.1 to have a large number of clusters of multiple eigenvalues. The third matrix, LAPLACE, comes from the usual seven-point discretization of the anisotropic three-dimensional Laplacian operator $-\partial^2/(\partial x)^2 - \partial^2/(\partial y)^2 - 100\partial^2/(\partial z)^2$ on the uniform grid $30 \times 30 \times 30$ with Dirichlet boundary conditions, and most of its eigenvalues have multiplicity two.

Although we make an effort to use similar parameters for all methods (tolerance, basis and block sizes, etc.), the reader should not focus on comparisons across methods. Different implementations make such comparison difficult, but most important, IVE improvements can be understood only in reference to the same method.

We perform experiments with two or three different tolerances, which is the only variable from section 2.1 that IVE does not vary. For IRBL, we request that the 2-norm of the residual is less than $\tau\lambda_{max}$. For ARPACK, we provide the tolerance τ to the method. For all PRIMME solvers we request that the 2-norm of the residual is less than $\tau\|A\|_F$. If $\epsilon = 2.2204^{-16}$ is the machine precision, we consider three cases; $\tau = \epsilon$, $\tau = \sqrt{\epsilon}$, and a large τ that is just small enough to differentiate between wanted and unwanted eigenvalues. Thus, the actual value of τ may vary across methods to achieve similar convergence accuracy. For these tolerances and for a variety of block sizes, we first perform an initial run of a solver and record the number of desired eigenvalues it was able to find and its performance. Then, using the approximations produced by a certain initial run, we let IVE guide the solver under various values

TABLE 4.1

*IRBL on matrix BCSSTK*16. *Table A shows results for computing the* 74 *smallest eigenvalues. For each block size (k) and tolerance ($\tau\lambda_{max}$) we report (nevFound, MV), where nevFound is how many of the* 74 *computed eigenvalues are the required ones and MV the number of matrix-vector multiplications. We underline the case that provided the initial set of vectors for the later IVE run. Table B shows the IVE results. For all maximum block sizes (mxk) and tolerances, IVE recovers all the* 74 *required eigenvalues, so we report only the number of MV performed during validation alone. The bottom of Table B shows the total MV required (initial run plus IVE) considering the fastest IVE run. The initial block k and the best mxk are also provided.*

A: IRBL/BCSSTK16 Initial runs			
	τ		
k	ϵ	$\epsilon^{1/2}$	2.023e-5
1	57, 162390	28, 15728	21, 5514
2	68, 211526	32, 17616	23, 7706
3	74, 265125	32, 17844	23, 7665
4	73, 298884	32, 13748	24, 7168
5	74, 283880	32, 489830	24, 20700
6	74, 272868	38, 11340	20, 5562
7	74, 287693	29, 9541	21, 9723
8	74, 318568	32, 9560	19, 4896
9	74, 363123	36, 11007	23, 5121
10	74, 398170	32, 12890	20, 4470
15	74, 541230	39, 12600	22, 4875
20	74, 679760	56, 9140	37, 4560
74	74, 118696	74, 114848	74, 8362

B: IRBL/BCSSTK16 IVE runs			
	τ		
mxk	ϵ	$\epsilon^{1/2}$	2.023e-5
1	72992	6921	5828
2	97292	7821	6028
3	76223	9229	4286
4	59392	7721	3628
5	53892	6621	4528
6	41756	6745	3680
7	64478	6981	4244
8	40988	6937	3968
9	52166	7546	4484
10	58092	6121	4528
Best total MV for Initial + IVE run			
	203378	21849	9142
k,mxk	1, 8	1, 10	1, 4

for maxBlockSize and record the total number of desired eigenvalues found and the corresponding performance. For each method and matrix, results appear in one table.

IRBL experiments are run under Matlab 6 on a Sun Microsystems Ultra 60 workstation. All other experiments are run on a PowerMac G5, with 1 GB of memory, 512 KB of L2 cache, 1 GHz memory bus, and two 2 GHz PowerPC processors, of which only one is used. ARPACK is compiled with g77 and -O optimization, while PRIMME solvers are compiled with gcc version 4.0.0, and -O optimization. All codes are linked with the Mac optimized VecLib library that includes LAPACK and BLAS.

4.3. Results with matrix BCSSTK16. The dimension of the matrix is 4884, and we seek the 74 smallest eigenvalues which are all 1.0. The next largest eigenvalue is O(1e6), and the largest one is 4.5e+9. This matrix is useful for showing the behavior of solvers and IVE in the presence of an exact, high multiplicity.

First, we test IRBL in Table 4.1. In Table 4.1A we perform several initial runs with block sizes k, each using a maximum basis size of $m = k\lfloor 100/k \rfloor$, a maximum degree of the dampening polynomial equal to the dimension of the matrix, and a size of the dampening interval of $m/2$. For the smallest tolerance, IRBL finds all needed eigenvalues with $k > 4$. The best performance is with $k = 74$, but we assume we do not know the multiplicity a priori. For the larger tolerances, IRBL cannot produce the invariant space of the multiple eigenvalue without $k = 74$.

Table 4.1B shows the performance of various IVE runs, each for a different maximum block size (mxk) and starting after the underlined initial run in Table 4.1A was performed. IVE helps IRBL find all 74 required eigenvalues for all tolerances τ and mxk. Because IRBL is a method that benefits in general from larger block sizes, it is advisable to allow larger values of mxk. Indeed, the number of matrix-vector operations taken by the initial run followed by the IVE($mxk = 10$) is comparable to or significantly smaller than the most robust initial run for $\tau = \epsilon, \tau =$2e-5. For certain

TABLE 4.2

GD+k on matrix BCSSTK16. Table A shows results for computing the 74 smallest eigenvalues. For each block size (k) and tolerance ($\tau\|A\|_F$) we report (nevFound, MV, time), where nevFound is how many of the 74 computed eigenvalues are the required ones, MV the number of matrix-vector multiplications, and time is in seconds. We underline the case that provided the initial set of vectors for the later IVE run. Table B shows the IVE results. For all maximum block sizes (mxk) and tolerances, IVE recovers all the 74 required eigenvalues, so we only report the number of MV and time taken during validation alone. The bottom of Table B shows the total MV and time required (initial run plus IVE), considering the fastest IVE run. The initial block k and the best mxk are also provided. IVE is required for robustness ($\sqrt{\epsilon}$ and 1e-6), and can improve time (ϵ).

A: GD+k/BCSSTK16 Initial runs

	τ		
k	ϵ	$\epsilon^{1/2}$	1e-6
1	37, 15097, 93	20, 5469, 37	13, 3232, 25
2	39, 16994, 97	24, 6883, 42	16, 4059, 27
3	41, 19591, 113	25, 7593, 46	17, 4510, 30
4	45, 20247, 121	27, 7700, 48	19, 4652, 30
5	49, 22214, 135	28, 7974, 50	20, 4968, 33
6	44, 28814, 168	27, 10640, 67	19, 6046, 40
7	54, 23754, 149	30, 8834, 56	22, 5451, 37
8	51, 25012, 155	29, 9761, 63	21, 5884, 39
9	48, 35213, 225	28, 12922, 85	21, 7476, 53
10	74, 40940, 305	37, 9557, 72	27, 6167, 47
15	74, 37138, 370	36, 8134, 83	32, 5181, 49
20	74, 31794, 362	42, 9027, 104	26, 5018, 60
30	73, 24647, 364	42, 7249, 102	30, 4609, 71
40	74, 26451, 497	42, 7111, 135	42, 4473, 87
50	74, 20936, 458	55, 7089, 165	52, 4789, 111
74	74, 19058, 600	73, 6946, 220	71, 4575, 153

B: GD+k/BCSSTK16 IVE runs

	τ		
mxk	ϵ	$\epsilon^{1/2}$	1e-6
1	9572, 82	5943, 45	4274, 33
2	9464, 77	6070, 42	4564, 32
3	8679, 69	6028, 41	4803, 33
4	9300, 77	6284, 45	4784, 34
5	9585, 78	6256, 47	5056, 38
6	9555, 78	5980, 45	5060, 36
7	11044, 95	6547, 49	4973, 38
8	15584, 152	6944, 52	4802, 35
Best total MV/times for Initial + IVE run			
	28270, 182	11497, 78	7796, 57
k,mxk	3, 3	1, 3	1, 2

moderate values of mxk, performance of IRBL-IVE improves further. Therefore, users are free to tune their code for performance without worrying about its robustness.

Table 4.2 shows similar results for the GD+k method. The maximum basis size used for both initial and IVE runs was 30 for $k < 10$ and $6k$ for $k \geq 10$. The GD+k method requires a larger block size to find the desired invariant space, however, at a great cost in performance. It is thus more effective to select a block size for its caching performance in the initial run (e.g., $k = 3$ in $\tau = \epsilon$ case) and then let IVE resolve the multiplicity with a similar block size. For the $\tau = \epsilon$ case, this strategy improves performance often by a factor of two or three over just using GD+k with a sufficiently large block. For the larger tolerance cases, IVE is the only way to compute the required eigenvalues.

Table 4.3 shows the results with the JDQMR method. The maximum basis sizes used were as in the GD+k method. Despite its efficiency, JDQMR has trouble identifying high multiplicities even with large block sizes. For inner-outer methods, such as JDQMR, this is not surprising as they focus mostly on improving particular eigenvectors, and thus are prone to misconvergence. Also, large block sizes are usually not beneficial. For all three tolerances, IVE restores robustness to JDQMR, and for a relatively low cost. Note that the best times observed would not increase substantially if a larger mxk were used.

Table 4.4 shows results with LOBPCG-W. This method requires that the maximum basis size is $3k$. LOBPCG is inherently a block method that should benefit from larger k. Although, the initial run of the method finds more required eigenvalues with small k than other methods do, a few are always missed, even with $k = 74$. A surprising exception is the $k = 1$ in the $\tau = \epsilon$ case, which finds all of them, but such

TABLE 4.3

*JDQMR on matrix BCSSTK*16. *Similar to Table* 4.2, *Table A shows results for initial runs and Table B for various IVE runs, including the best performance obtained. IVE is required for robustness. Block sizes larger than one do not increase execution time significantly.*

A: JDQMR/BCSSTK16 Initial runs			
	τ		
k	ϵ	$\epsilon^{1/2}$	1e-6
1	37, 18116, 38	21, 7345, 16	12, 4233, 10
2	38, 20625, 44	22, 8089, 18	14, 4957, 12
3	38, 21991, 47	23, 8684, 20	15, 5347, 13
4	39, 24272, 52	24, 9431, 22	17, 6009, 15
5	40, 26171, 57	26, 10437, 25	18, 6744, 17
6	41, 25733, 56	26, 10282, 24	18, 6370, 16
7	42, 29610, 64	27, 11450, 28	20, 7760, 21
8	41, 27342, 59	28, 11423, 27	19, 7347, 19
9	43, 30933, 68	27, 11109, 27	19, 6898, 18
10	47, 37911, 85	30, 12470, 32	21, 8082, 23
15	47, 37919, 93	30, 11769, 36	23, 7434, 24
20	48, 30928, 81	32, 11106, 36	24, 8393, 32
30	49, 31587, 95	35, 11626, 47	28, 7076, 32
40	53, 29998, 104	40, 10522, 52	37, 8164, 43
50	70, 43311, 163	48, 10829, 61	43, 7212, 41
74	71, 35835, 181	62, 11150, 84	61, 9407, 83

B: JDQMR/BCSSTK16 IVE runs			
	τ		
mxk	ϵ	$\epsilon^{1/2}$	1e-6
1	11522, 29	6886, 20	5029, 19
2	12592, 33	8034, 23	6136, 22
3	13362, 33	8656, 23	6179, 20
4	13927, 34	8196, 24	6381, 20
5	15783, 37	8921, 25	6765, 20
6	16415, 40	8684, 25	7047, 21
7	14493, 34	8546, 22	7135, 21
8	16882, 39	8449, 24	7908, 22
Best total MV/times for Initial + IVE run			
	29638, 67	14231, 38	9269, 29
k,mxk	1, 1	1, 1	1, 1

TABLE 4.4

*LOBPCG-W on matrix BCSSTK*16. *Similar to Table* 4.2, *Table A shows results for initial runs and Table B for various IVE runs, including the best performance obtained. IVE is required for robustness, and beyond the optimal block size, time does not increase significantly.*

A: LOBPCG-W/BCSSTK16 Initial runs			
	τ		
k	ϵ	$\epsilon^{1/2}$	1e-6
1	74,450005,1848	60, 76687, 323	44,34279,156
2	62, 85413, 370	45, 29439, 148	32, 14384,75
3	62, 68730, 325	43, 23491, 120	30, 10998,61
4	65, 65387, 336	42, 19570, 105	28, 9467, 54
5	66, 58429, 319	39, 16659, 97	27, 8763, 51
6	67, 59416, 354	39, 16738, 104	27, 7996, 49
7	68, 53514, 330	41, 15692, 105	27, 8057, 52
8	67, 51463, 337	39, 13773, 92	27, 7440, 50
9	67, 46950, 310	39, 13534, 98	27, 7396, 54
10	69, 49260, 336	37, 11749, 80	27, 6590, 45
15	68, 39809, 322	38, 9898, 82	28, 5999, 50
20	68, 36599, 344	39, 9277, 91	30, 5678, 54
30	68, 34169, 401	43, 8277, 95	31, 4916, 58
40	72, 31679, 454	44, 7875, 118	41, 5074, 78
50	72, 38744, 655	52, 7845, 132	48, 5139, 87
74	73, 31654, 704	69, 7908, 169	64, 5240,116

B: LOBPCG-W/BCSSTK16 IVE runs			
	τ		
mxk	ϵ	$\epsilon^{1/2}$	1e-6
1	5799, 30	6225, 32	4554, 24
2	7938, 46	5774, 31	4656, 25
3	12846, 86	5580, 31	4412, 24
4	9800, 68	6052, 37	4692, 28
5	10998, 81	6025, 36	4641, 27
6	11442, 90	6272, 41	4540, 29
7	7496, 60	6020, 41	4704, 31
8	7586, 62	6704, 48	5256, 36
Best total MV/times for Initial + IVE run			
	52749, 340	25150, 136	11002, 69
k,mxk	9, 1	4, 3	10, 3

behavior is not expected in general. IVE guided LOBPCG-W finds all eigenvalues, and at very competitive times.

Table 4.5 shows results with ARPACK, for three different tolerances. ARPACK, as classical Krylov method without locking, can only identify a few of the eigenvectors in the multiplicity. As with the experiments in LOBPCG-W and JDQMR, decreasing the convergence tolerance does not solve the problem. Yet, with the guidance of the simplified IVE, ARPACK can solve the problem with no difficulties.

TABLE 4.5

ARPACK on matrix BCSSTK16. This is not a block method, so we vary only the tolerance τ. We report (nevFound, MV, time) for the initial run of ARPACK and the total (nevFound, MV, time) taken by both initial and the subsequent IVE runs. The robustness gains are significant.

	Initial runs			IVE run		
τ	evals	MV	Time	evals	Total MV	Total time
1e-16	29	4435	86	74	20567	259
1e-8	6	1240	20	74	18233	226
1e-7	5	1150	18	74	16974	211

TABLE 4.6

GD+k on matrix DIAG. Table A shows results for computing the eight smallest eigenvalues, and Table B shows the subsequent IVE results. We follow the same format as in previous tables, reporting (nevFound, MV, time) for the initial as well as the IVE runs. A higher block size is needed in case $\tau = 5.49E-9$ to find all the desired eigenvalues. Although GD+k does not benefit from large block size in general, performance improves through an efficient initial run followed by a robust IVE run.

A: GD+k/DIAG Initial runs

	τ	
k	ϵ	$5.49E-9$
1	5, 6777, 178.1	1, 567, 14.9
2	6, 13118, 305.4	2, 564, 11.6
3	8, 203927, 5127.2	3, 1267, 28.7
4	8, 24124, 521.0	4, 1548, 31.1
5	8, 48110, 1147.3	5, 2230, 50.1
6	8, 43310, 1056.9	6, 2401, 55.1
7	8, 86573, 2142.4	7, 4376, 105.0
8	8, 88696, 2264.9	8, 7008, 168.1

B: GD+k/DIAG IVE runs

	τ	
mxk	ϵ	$5.49E-9$
1	8, 3858, 89.1	4, 598, 13.2
2	8, 7184, 162.0	6, 976, 19.5
3	8, 8337, 159.6	8, 1480, 29.1
4	8, 34628, 770.1	8, 2216, 44.6
5	8, 47253, 1014.4	8, 2300, 39.9
6	8, 508381, 11340.8	8, 3159, 59.2
7	8, >600K 13384.4	8, 3827, 76.9
8	8, >600K 16084.9	8, 3532, 78.5
Best total MV/times for Initial + IVE run		
	10635, 267.2	2044, 40.7
k,mxk	1, 1	2, 3

For BCSSTK16 and for all methods tested, a maximum block size (mxk) of one was sufficient for robustness in IVE, showing the synergetic effects of the rest of its features, such as locking, computing more eigenpairs, and systematically introducing random guesses. As we show next, larger mxk may be needed in other cases.

4.4. Results with matrix DIAG. We seek the lowest eight eigenvalues which are equal to machine precision. Despite the lower multiplicity, most eigensolvers misconverge to some of the eigenvalues in the second nearby cluster.

Results from GD+k tests appear in Table 4.6. For $\tau = \epsilon$, the method misses a few eigenvalues if the block size is less than three, which is a typical block size for GD+k. With the larger τ, GD+k finds a dimension of the required invariant subspace equal to the block size. This is a typical case, where the block size must be increased for robustness in a code whose performance deteriorates with larger block sizes. IVE avoids this trade-off, thus decoupling performance and robustness. By choosing an mxk such as 2 or 3, which is reasonable for low tolerances in GD+k (and not depending on the particular problem), IVE helps GD+k find all required eigenvalues and much faster than any initial run by GD+k. As expected, a slightly larger block may be needed for larger tolerances, but its exact value does not significantly affect performance. Note that for $\tau = 5.49E-9$, IVE must be allowed to set a block size larger than one to recover robustness.

In Table 4.7 we show results from JDQMR on the DIAG matrix. As with GD+k, IVE can recover robustness for JDQMR if it is allowed to use a larger block size.

TABLE 4.7

JDQMR on matrix DIAG. Initial and IVE runs are shown similarly to Table 4.6. *Typically, JDQMR is used with block size of one, although a larger one does not affect it significantly. Except for the larger block sizes in $\tau = \epsilon$, a combination of initial and IVE runs matches the performance of the most robust JDQMR, but without a priori knowledge of optimal block.*

A: JDQMR/DIAG Initial runs		
	τ	
k	ϵ	$5.49E-9$
1	5, 7412, 21.5	1, 776, 4.5
2	6, 9382, 29.1	2, 765, 4.2
3	7, 12230, 38.5	3, 1453, 6.9
4	8, 12998, 38.8	4, 1650, 7.6
5	8, 14537, 47.7	5, 1401, 6.3
6	8, 16583, 52.7	6, 1648, 7.4
7	8, 17990, 61.4	7, 2822, 12.7
8	8, 11402, 36.7	8, 3908, 17.4

B: JDQMR/DIAG IVE runs		
	τ	
mxk	ϵ	$5.49E-9$
1	8, 4471, 13.6	1, 406, 2.3
2	8, 7967, 25.6	1, 1462, 8.4
3	8, 11723, 33.8	6, 1855, 8.4
4	8, 18817, 57.2	8, 3229, 15.2
5	8, 25618, 77.9	6, 2597, 11.0
6	8, 33243, 99.8	8, 3950, 18.4
7	8, 35727, 108.7	8, 4431, 19.6
8	8, 39984, 118.1	8, 3592, 17.8
Best total MV/times for Initial + IVE run		
	11883, 35.1	4005, 19.7
k,mxk	1, 1	1, 4

TABLE 4.8

LOBPCG-W on matrix DIAG. Initial and IVE runs are shown similarly to Table 4.6. *For $\tau = \epsilon$, LOBPCG-W is robust for most block sizes, but the IVE does not add substantial overhead. For $\tau = 5.49E-9$, IVE is the only way to obtain the required eigenvalues. Requiring the smaller $\tau = \epsilon$ instead would be unnecessarily more expensive.*

A: LOBPCG-W/DIAG Initial runs		
	τ	
k	ϵ	$5.49E-9$
1	8, 166099, 1416.6	1, 836, 7.3
2	7, 192388, 1948.3	2, 1370, 13.1
3	8, 432877, 4949.2	3, 1419, 16.6
4	8, 196039, 2542.5	3, 1367, 16.7
5	8, 189875, 2548.1	3, 1360, 16.5
6	8, 187032, 2807.7	4, 2686, 40.0
7	8, 230628, 3955.3	5, 2961, 48.7
8	8, 120816, 2053.3	4, 2072, 31.8

B: LOBPCG-W/DIAG IVE runs		
	τ	
mxk	ϵ	$5.49E-9$
1	8, 8714, 69.5	6, 829, 8.1
2	8, 23982, 268.3	5, 654, 7.8
3	8, 62862, 841.4	7, 3897, 59.1
4	8, 89604, 1296.8	8, 4040, 53.7
5	8, 144105, 2092.0	8, 7688, 125.3
6	8, 94350, 11915.2	8, 8089, 116.5
7	8, >600K, 13285.9	8, 8230, 119.8
8	8, >600K, 14419.4	8, 8302, 121.1
Best total MV/times for Initial + IVE run		
	201102, 2017.8	5407, 70.4
k,mxk	2, 1	4, 4

Moreover, mxk does not have to match the multiplicity sought (as is necessary in the initial runs), because the synergy of the many IVE features compensates for smaller block sizes. For this solver, IVE execution times are relatively insensitive to mxk, matching the best performance of the initial runs and with the added robustness.

In Table 4.8, LOBPCG-W finds all the required eigenvalues for $\tau = \epsilon$ and for all k, except for $k = 2$ for which IVE recovers the missing eigenvalue for a relatively small additional cost. Surprisingly, for $\tau = 5.49E-9$ LOBPCG-W cannot identify the full invariant subspace in the initial run, regardless of block size. When IVE is allowed to increase the block size above three, all eigenvalues are found.

Finally, in Table 4.9, we show the results from ARPACK on the DIAG matrix and for three different tolerances. Through a combination of locking, restarting, and finding more eigenvalues, IVE is able to recover all needed eigenvalues, except when

TABLE 4.9

ARPACK on matrix DIAG. As in Table 4.5*, IVE is able to achieve the required robustness on the DIAG matrix for the two smallest tolerances, and with minimal additional overhead. The need of a larger block size is evident for such pathological cases and for large tolerances (τ =1e-7).*

	Initial runs			IVE run		
τ	evals	MV	Time	evals	Total MV	Total time
1e-16	5	305990	5004	8	309889	5092
1e-8	5	27678	443	8	30885	515
1e-7	2	4167	68	2	4256	70

TABLE 4.10

GD on matrix LAPLACE. Table A shows results for computing the 19 *smallest eigenvalues, and Table B shows the subsequent IVE results. We follow the format of Table* 4.2*, testing only one tolerance and not reporting the nevFound in IVE, which finds all eigenvalues. We test block sizes up to the maximum multiplicity of two. IVE restores robustness, and improves time in all cases.*

A: GD/LAPLACE Initial runs	
k	τ = 1e-9
1	18, 10812, 116.9
2	19, 37497, 391.8

B: GD/LAPLACE IVE runs	
mxk	τ = 1e-9
1	3944, 49.9
2	22186, 254.9
Best total MV/times for Initial + IVE run	
	14756, 166.8
k,mxk	1, 1

$\tau = 1e{-}7$, which is close to the first intercluster gap (`DeltaCluster` in Figure 4.1). As with the other methods, a slightly larger block size would have resolved the problem.

Up to now, we have seen test cases where keeping $mxk = 1$ achieved both the required robustness and the minimum time, cases where a larger mxk was required for performance, and cases where a larger mxk was required for robustness. But the optimum choice of mxk does not depend on the problem but depends mainly on the solver and the accuracy needed. For example, given a Krylov solver with basis size m, it makes no sense in terms of convergence to set $mxk > m/3$, with values less than $m/5$ being more reasonable. In the above IVE tests and for all solvers, the maximum multiplicity was larger than $m/5$, hence the observed decrease in performance with larger mxk. Other solvers with larger basis sizes could behave differently. Nevertheless, IVE restores robustness even with very modest block sizes, and therefore users can decouple the choices of parameters that are needed for convergence and cache performance from the parameters needed for robustness on a particular problem.

4.5. Results with a Laplacian matrix. The LAPLACE matrix is closer to a real world problem. We seek the 19 algebraically smallest eigenvalues, 16 of which come in multiple pairs. In particular, the 18th and 19th eigenvalues are a multiple pair. We give results only for τ =1e-9, and for $mxk = 1, 2$ as IVE will not go beyond the maximum observed multiplicity. Table 4.10 shows the results of the GD method, which is identical to GD+k but without the CG-type restarting. An initial block size of 1 misses one eigenvalue, which is easily restored by IVE and in shorter time than to run an initial block size of 2. Table 4.11 for GD+k shows that even when the method does not miss an eigenvalue, the validation step is not expensive relatively. Table 4.12 shows similar results with JDQMR, where IVE restores robustness in better or similar time as the most robust initial run. Table 4.13 shows that LOBPCG-W with block size of one is neither robust nor efficient, unlike the BCSSTK16 case, but IVE

TABLE 4.11
GD+k on matrix LAPLACE. Similar to Table 4.10*, Table A shows results for initial runs and Table B for various IVE runs. Best performance is not included because the initial run does not miss eigenvalues. Yet, the additional IVE expense is not significant.*

A: GD+k/LAPLACE Initial runs	
k	τ = 1e-9
1	19, 5062, 62.7
2	19, 7852, 86.4

B: GD+k/LAPLACE IVE runs	
mxk	τ = 1e-9
1	654, 8.6
2	1180, 14.9

TABLE 4.12
JDQMR on matrix LAPLACE. Similar to Table 4.10*, Table A shows results for initial runs and Table B for various IVE runs, including the best performance obtained. Robustness is restored for a smaller or comparable cost.*

A: JDQMR/LAPLACE Initial runs	
k	τ = 1e-9
1	18, 5360, 18.7
2	19, 7144, 25.0

B: JDQMR/LAPLACE IVE runs	
mxk	τ = 1e-9
1	1213, 4.7
2	2779, 11.2
Best total MV/times for Initial + IVE run	
	6573, 23.4
k,mxk	1, 1

TABLE 4.13
LOBPCG-W on matrix LAPLACE. Similar to Table 4.10*, Table A shows results for initial runs and Table B for various IVE runs, including the best performance obtained. LOBPCG-W with block size of one is slow, and misses an eigenvalue, while IVE recovers it with relatively small cost.*

A: LOBPCG-W/LAPLACE Initial	
k	τ = 1e-9
1	18, 65722, 596.8
2	19, 13271, 145.8

B: LOBPCG-W/LAPLACE IVE runs	
mxk	τ = 1e-9
1	3142, 29.1
2	2428, 28.9
Best total MV/times for Initial + IVE run	
	68864, 625.7
k,mxk	1, 2

TABLE 4.14
ARPACK on matrix DIAG. As in Table 4.5*, IVE is able to achieve the required robustness on the LAPLACE matrix. Note that ARPACK missed three eigenvalues despite a relatively low tolerance.*

	Initial runs			IVE run		
τ	evals	MV	Time	evals	Total MV	Total time
1e-9	16	2874	72	19	6042	137

recovers the missing eigenvalue quite inexpensively. Finally Table 4.14 demonstrates the robustness of IVE with ARPACK, even with only a few IVE features enabled.

5. Conclusions. Hermitian, iterative eigenvalue solvers cannot guarantee, without factorizing the matrix, that the eigenvalues they compute are the required ones. Eigenvalues that are tightly clustered or of high algebraic multiplicity can be missed by any type of eigensolver. There are many techniques to alleviate this: ask for more than the required eigenpairs, and to better than sufficient accuracy, or use a block size larger than the maximum multiplicity sought. Choosing these parameters, however, requires a priori knowledge about the problem, and increasing them beyond a certain value slows convergence and reduces cache performance. Alternatively, locking

converged eigenvectors, and restarting the solver with new random vectors improves robustness, also at the cost of convergence. Typically, robust choices lead to unnecessarily slow codes, especially for the easier parts of the required spectrum.

Our a posteriori IVE is essentially a wrapper around any given eigensolver that automates all the above choices of techniques, except for accuracy, to provide robustness when this is needed. Relying on the synergy of these techniques, IVE keeps calling the solver to find additional eigenpairs, until no missed eigenvalues can be identified. IVE works with both single-vector and block solvers, requiring no changes to the solver.

There are two key ideas in IVE. The first is the repeated Rayleigh–Ritz minimization in the space orthogonal to all converged eigenvectors, each time starting with at least one random initial guess. The second is an automatic way to increase the block size based on the largest multiplicity computed thus far, hence increasing the chances of finding missed eigenvalues of similar multiplicity.

Both key ideas can be very expensive during the initial run of any eigensolver, but as an a posteriori technique they focus only on the problematic part of the spectrum, and only if needed. Moreover, the ability to resolve multiple or clustered eigenvalues during a postprocessing phase frees the user to employ in the solver the block size that gives the best cache performance and the tolerance required by the problem. Even when the block size is limited to modest values in IVE, the synergetic effects of the rest of the techniques can deliver the desired robustness. Our experiments support both the robustness of IVE on some very hard problems and its ability to decouple robustness from performance choices.

In future work, we plan to investigate the applicability and effectiveness of IVE for interior eigenvalues and for non-Hermitian eigenproblems. The major obstacle is that the Rayleigh–Ritz procedure does not provide monotonic convergence toward interior eigenvalues. Thus, the existence of a Ritz value that is closer to a value σ than any previously computed eigenvalue is does not imply that an eigenvalue was missed. A harmonic-Ritz procedure, which may provide a solution to this problem, unfortunately does not extend to non-Hermitian problems.

Acknowledgment. We thank the referees for helping us shape a better paper.

REFERENCES

[1] J. Baglama, D. Calvetti, and L. Reichel, *IRBL: An implicitly restarted block Lanczos method for large-scale Hermitian eigenproblems*, SIAM J. Sci. Comput., 24 (2003), pp. 1650–1677.

[2] J. Baglama, D. Calvetti, and L. Reichel, *IRBLEIGS: A MATLAB program for computing a few eigenpairs of a large sparse Hermitian matrix*, ACM Trans. Math. Software, 29 (2003), pp. 337–348.

[3] Z. Bai, D. Day, and Q. Ye, *ABLE: An adaptive block Lanczos method for non-Hermitian eigenvalue problems*, SIAM J. Matrix Anal. Appl., 20 (1999), pp. 1060–1082.

[4] Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. van der Vorst, eds., *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide*, SIAM, Philadelphia, 2000.

[5] M. Clint and A. Jennings, *The evaluation of eigenvalues and eigenvectors of a real symmetric matrix by simultaneous iteration*, Computer J., 13 (1970), pp. 76–80.

[6] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 2nd ed., MIT Press, Cambridge, MA, 2003.

[7] E. R. Davidson, *The iterative calculation of a few of the lowest eigenvalues and corresponding eigenvectors of large real-symmetric matrices*, J. Comput. Phys., 17 (1975), pp. 87–94.

[8] J. J. Dongarra, I. S. Duff, D. C. Sorensen, and H. A. van der Vorst, *Numerical Linear Algebra for High Performance Computers*, SIAM, Philadelphia, 1998.

[9] I. Duff, R. G. Grimes, and J. G. Lewis, *Users' Guide for the Harwell-Boeing Sparse Matrix Collection (Release* I*)*, Technical Report TR/PA/92/86, CERFACS, Toulouse, France, Oct. 1992.

[10] R. Geus, *The Jacobi-Davidson Algorithm for Solving Large Sparse Symmetric Eigenvalue Problems with Application to the Design of Accelerator Cavities*, Ph.D. thesis, ETH, Zurich, Switzerland, 2002.
[11] G. H. Golub and C. F. Van Loan, *Matrix Computations*, John Hopkins University Press, Baltimore, MD, 1989.
[12] G. H. Golub and R. Underwood, *The block Lanczos method for computing eigenvalues*, in Mathematical Software III, J. R. Rice, ed., Academic Press, New York, 1977, pp. 361–377.
[13] R. G. Grimes, J. G. Lewis, and H. D. Simon, *A shifted block Lanczos algorithm for solving sparse symmetric generalized eigenproblems*, SIAM J. Matrix Anal. Appl., 15 (1994), pp. 228–272.
[14] C. Hsu and J. Demmel, *Effects of Block Size on the Block Lanczos Algorithm*, Technical report, Department of Mathematics, University of California, Berkeley, June 2003.
[15] A. V. Knyazev, *Toward the optimal preconditioned eigensolver: Locally optimal block preconditioned conjugate gradient method*, SIAM J. Sci. Comput., 23 (2001), pp. 517–541.
[16] A. B. J. Kuijlaars, *Convergence analysis of Krylov subspace iterations with methods from potential theory*, SIAM Rev., 48 (2006), pp. 3–40.
[17] C. Lanczos, *An iterative method for the solution of the eigenvalue problem of linear differential and integral operators*, J. Res. Nat. Nur. Stand., 45 (1950), pp. 255–282.
[18] R. B. Lehoucq and D. C. Sorensen, *Deflation techniques for an implicitly restarted Arnoldi iteration*, SIAM J. Matrix Anal. Appl., 17 (1996), pp. 789–821.
[19] R. B. Lehoucq, D. C. Sorensen, and C. Yang, *ARPACK USERS GUIDE: Solution of Large Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods*, SIAM, Philadelphia, 1998.
[20] J. R. McCombs and A. Stathopoulos, *PRIMME: PReconditioned Iterative Multimethod Eigensolver*, http://www.cs.wm.edu/∼andreas/software/.
[21] R. B. Morgan, *A harmonic restarted Arnoldi algorithm for calculating eigenvalues and determining multiplicity*, Linear Algebra Appl., 415 (2006), pp. 96–113.
[22] R. B. Morgan, *On restarting the Arnoldi method for large nonsymmetric eigenvalue problems*, Math. Comput., 65 (1996), pp. 1213–1230.
[23] R. B. Morgan and D. S. Scott, *Generalizations of Davidson's method for computing eigenvalues of sparse symmetric matrices*, SIAM J. Sci. Comput., 7 (1986), pp. 817–825.
[24] B. N. Parlett, *The Symmetric Eigenvalue Problem*, SIAM, Philadelphia, 1998.
[25] H. Ruthishauser, *Simultaneous iteration method for symmetric matrices*, Numer. Math., 16 (1970), pp. 205–223.
[26] Y. Saad, *On the rate of convergence of the Lanczos and the block-Lanczos methods*, SIAM J. Numer. Anal., 17 (1980), pp. 687–706.
[27] Y. Saad, *Numerical Methods for Large Eigenvalue Problems*, Manchester University Press, Manchester, UK, 1992.
[28] G. L. G. Sleijpen, A. G. L. Booten, D. R. Fokkema, and H. A. van der Vorst, *Jacobi-Davidson type methods for generalized eigenproblems and polynomial eigenproblems*, BIT, 36 (1996), pp. 595–633.
[29] G. L. G. Sleijpen and H. A. van der Vorst, *A Jacobi-Davidson iteration method for linear eigenvalue problems*, SIAM J. Matrix Anal. Appl., 17 (1996), pp. 401–425.
[30] D. C. Sorensen, *Implicit application of polynomial filters in a K-step Arnoldi method*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 357–385.
[31] D. C. Sorensen, *Deflation for Implicitly Restarted Arnoldi Methods*, Technical Report TR98-12, Department of Computational and Applied Mathematics, Rice University, Houston, TX, 1998.
[32] A. Stathopoulos, *Nearly Optimal Preconditioned Methods for Hermitian Eigenproblems Under Limited Memory. Part* I: *Seeking One Eigenvalue*, SIAM J. Sci. Comput., to appear.
[33] A. Stathopoulos and C. F. Fischer, *A Davidson program for finding a few selected extreme eigenpairs of a large, sparse, real, symmetric matrix*, Comput. Phys. Comm., 79 (1994), pp. 268–290.
[34] A. Stathopoulos and Y. Saad, *Restarting techniques for (Jacobi-)Davidson symmetric eigenvalue methods*, Electron. Trans. Numer. Algebra, 7 (1998), pp. 163–181.
[35] A. Stathopoulos, Y. Saad, and K. Wu, *Dynamic thick restarting of the Davidson, and the implicitly restarted Arnoldi methods*, SIAM J. Sci. Comput., 19 (1998), pp. 227–245.
[36] W. J. Stewart and A. Jennings, *ALGORITHM* 570: *LOPSI a simultaneous iteration method for real matrices*, ACM Trans. Math. Software, 7 (1981), pp. 230–232.
[37] T. Yang, *Theoretical error bounds on the convergence of the Lanczos and block-Lanczos methods*, Comput. Math. Appl., 38 (1999), pp. 19–38.

SIAM J. SCI. COMPUT.
Vol. 28, No. 6, pp. 2359–2389
© 2006 Society for Industrial and Applied Mathematics

MULTISCALE MODELING OF PHYSICAL PHENOMENA: ADAPTIVE CONTROL OF MODELS*

J. TINSLEY ODEN†, SERGE PRUDHOMME†, ALBERT ROMKES‡, AND PAUL T. BAUMAN†

Abstract. It is common knowledge that the accuracy with which computer simulations can depict physical events depends strongly on the choice of the mathematical model of the events. Perhaps less appreciated is the notion that the error due to modeling can be defined, estimated, and used adaptively to control modeling error, provided one accepts the existence of a base model that can serve as a datum with respect to which other models can be compared. In this work, it is shown that the idea of comparing models and controlling model error can be used to develop a general approach for multiscale modeling, a subject of growing importance in computational science. A posteriori estimates of modeling error in so-called quantities of interest are derived and a class of adaptive modeling algorithms is presented. Several applications of the theory and methodology are presented. These include preliminary work on random multiphase composite materials, molecular statics simulations with applications to problems in nanoindentation, and analysis of molecular dynamics models using various techniques for scale bridging.

Key words. multiscale modeling, goal-oriented adaptive modeling, a posteriori error estimation

AMS subject classifications. 65G20, 74G10, 74H15

DOI. 10.1137/050632488

1. Introduction.

1.1. Introductory remarks. The development of mathematical and computational modeling techniques for simulating physical events that occur across several spatial and temporal scales has become one of the most challenging and important areas of computational science. Advances in nanotechnolgy, semiconductors, cell and molecular biology, biomedicine, geological and earth sciences, and many other areas hinge upon understanding the interactions of events at scales that could range from subatomic dimensions and nanoseconds to macroscales and hours or even centuries. Available computational methods have largely been designed to model events that take place over one, or rarely two, spatial scales and are generally nonapplicable to multiscale phenomena. Molecular dynamics (MD), for example, may provide an acceptable basis for modeling atomistic or molecular motions over a time period on the order of nanoseconds, while nanodevice fabrications may involve events occurring over seconds or minutes.

In the present exposition, we develop a general approach to multiscale modeling based on the notion of a posteriori estimation of modeling error and on adaptive modeling using so-called Goals algorithms. By a mathematical model we mean a collection of mathematical constructions—partial differential, integral, ordinary, or algebraic equations, boundary and initial conditions, constraints, and data—that provide

*Received by the editors May 26, 2005; accepted for publication (in revised form) June 16, 2006; published electronically December 26, 2006. The support of this work by ONR under contract N00014-99-1-0124 and by DOE under contract DE-FG02-05ER25701 is gratefully acknowledged.
http://www.siam.org/journals/sisc/26-6/63248.html
†Institute for Computational Engineering and Sciences, The University of Texas at Austin, Austin, TX 78712 (oden@ices.utexas.edu, serge@ices.utexas.edu, pbauman@ices.utexas.edu). The fourth author thanks the DOE Computational Science Graduate Fellowship for financial support.
‡Department of Mechanical Engineering, The University of Kansas, Lawrence, KS 66045-2234 (romkes@ku.edu).

an abstraction of a class of physical phenomena covered by scientific theory. Traditionally, a model is selected by a modeler, the analyst, who chooses a model, based on experience or empirical evidence, heuristics, and personal judgment, to depict events assumed to involve well-defined spatial and temporal scales. Events at different scales require, in general, the use of different models. Thus, multiscale models should generally involve a blending or adaptation of models of one scale of events with those of another. It would seem to follow, therefore, that successful multiscale modeling techniques should be able to compare models of different structure and to adapt features of different models so that they deliver results of an accuracy sufficient to capture essential features of the response or to make engineering decisions. This is the basis for the methodologies developed in the present work.

This presentation reviews and extends the theory and methodology developed in earlier works (e.g., [25]) and extends the scope of applications to a broad range of problems, including problems in multiphase random materials and molecular statics. The general tact is different from other techniques proposed in the rapidly growing literature on multiscale modeling.

Several surveys of the literature on multiscale modeling have been published, and we mention in particular the articles of Liu et al. [19], E, Li, and Vanden-Eijnden [12], and Curtin and Miller [8]. In [19], it is asserted that multiscale methods can be naturally grouped into two categories: concurrent and hierarchical. Concurrent methods simultaneously solve a fine-scale model in some local region of interest and a coarser-scale model in the remainder of the domain. Hierarchical, or serial coupling methods [11], use results of a fine-scale model simulation to acquire data for a coarser-scale model that is used globally, e.g., to determine parameters for constitutive equations. The methodologies we develop here are more akin to concurrent approaches since the notion of adaptive modeling fits naturally within that framework. Additionally, the various applications we present here are largely described in terms of problems in mechanics and materials science, although the framework is very general and is applicable to multiscale problems in virtually all areas of science.

We identify three main thrusts in the current literature for implementing multiscale methods. The first involves dimensional reduction approaches such as that used by Tadmor, Ortiz, and Phillips in the quasicontinuum method (QCM) [44, 46]. The QCM has been applied to quasistatic fracture [20, 23], grain-boundary interaction [39], and nanoindentation [45, 43]. In these applications, the QCM has led to a dramatic reduction in the number of degrees of freedom while still resolving physical features of interest.

A second line of research in concurrent multiscale methods is the decomposition-of-scales technique used by Liu et al. in the bridging-scale method (BSM) [51, 32, 47, 48]. The scale decomposition provides a natural separation of spatial and temporal scales for coupling MD and continuum models and has successfully reduced spurious wave reflection in preliminary tests. A static formulation has been applied to the deformation of carbon nanotubes in [35], and a three-dimensional generalization to problems in dynamic fracture is presented in [31].

Finally, widely used concurrent methods are those that directly interface atomistic and continuum models using "pad," "overlap," or "hand-shake" regions. The first method to use such schemes was the FEAt method [18], which was used to simulate the fracture of body-centered cubic (b.c.c.) crystals using a molecular and continuum model. The FEAt method later provided the foundations of the well-known molecular, atomistic, ab initio dynamics (MAAD) or coupling of length scales (CLS) method [7], which was the first to concurrently couple quantum, molecular,

and continuum scales. The atomistic-continuum interface in the MAAD method was extended by Belytschko and Xiao with the bridging-domain method [5, 52] by using an augmented Lagrange method to enforce displacement boundary conditions in the overlap region as well as separating temporal scales in the numerical implementation. The coupled atomistic and discrete dislocation plasticity (CADD) method [41] was developed by Shilkrot, Miller, and Curtin to interface an atomistic model, similar to the QCM, with a more complex continuum model that incorporates descriptions for dislocation movement [49]. In this way, dislocations can be passed to the continuum model, or vice-versa, thereby preserving degrees of freedom that would otherwise be needed to track the dislocation movement. Published implementations of the CADD method are two-dimensional and static, although work in developing three-dimensional and dynamic models is reportedly underway [42].

There also exist more general schemes that develop modeling frameworks rather than concentrating on the details of model interfaces. The heterogeneous multiscale method (HMM) [10], proposed by E and Engquist, provides such a framework for designing multiscale methods based upon the particular nature of the problem. The HMM has been applied to many mathematical problems that possess solutions with multiple scales (see, e.g., [11]), and some work has been done in a priori error estimation for elliptic problems [13].

The idea of estimating modeling error and controling error through model adaptivity was inspired by work on a posteriori estimation of numerical approximation error and control of error through adaptive meshing [1, 24, 4]. Oden, Zohdi, and coworkers [53, 30, 29] used global error estimates to drive adaptivity of models of multiphase heterogeneous elastic materials. Upper and lower bounds of modeling errors in local quantities of interest for this class of problems were presented in [28] and model adaptivity based on the Goals algorithm was first presented in [50]. Further extensions of these approaches are described in [36, 26], and a general theory for model error estimation is proposed in [25].

In the remainder of this introduction, we review the basic ideas behind our error estimation methodology and the Goals algorithm for adaptive modeling, and we lay down notation and assumptions prerequisite to subsequent developments. The basic theorems on model error estimation are taken up in section 2. Discussions of some technicalities connected with ensemble averaging and homogenization are given in section 3. The Goals algorithms are discussed in section 4. There we describe two general versions of these types of algorithms that have proved to be very effective in several application areas. A brief account of global error estimates is given in section 5. Applications are discussed in section 6, and conclusions are collected in a final section.

1.2. Basic ideas. The basic idea behind the multiscale modeling approach to be described here is one of computing and controling modeling error. We first assume that we can define a general mathematical model of a physical system which can serve as a datum with respect to which other models can be compared. This *fine* or *base* model may be highly complex, even intractable, it may involve incomplete data, and, in general, it is never "solved." In studying multiscale phenomena, this base model will depict events that take place on the finest (smallest) spatial and temporal scales. An abstract setting for many base models is embodied in the problem

$$\text{Find } u \in V \text{ such that } A(u) = F \text{ in } V',$$

where V is an appropriate topological vector space, $A(\cdot)$ is a nonlinear operator mapping V into its dual, V', and F is given data in V'. If $\langle \cdot, \cdot \rangle$ denotes duality pairing on $V' \times V$, then the above problem is equivalent to

(1)
$$\boxed{\begin{array}{l}\text{Find } u \in V \text{ such that} \\ \qquad B(u; v) = F(v) \quad \forall v \in V\end{array}}$$

where

$$B(u; v) = \langle A(u), v \rangle, \quad F(v) = \langle F, v \rangle, \tag{2}$$

the semicolon signaling that $B(\cdot; \cdot)$ may be a nonlinear function of the argument u to the left of the semicolon, but linear in the test vector v to the right of it.

Now, in all applications of interest, we do not merely want to characterize the vectors u satisfying (1), but rather we want to find a particular physical event or feature of the system that depends upon u. Such *quantities of interest* are characterized by functionals Q of the solutions to (1). Thus, our actual target problem is this:

(3)
$$\boxed{\begin{array}{l}\textit{If } u \textit{ is a solution of } (1), \textit{ find } Q(u), \textit{ where } Q : V \to \mathbb{R} \\ \textit{is a given functional defining a quantity of interest.}\end{array}}$$

If (1) is intractable, so is (3). We must therefore seek an approximation to $Q(u)$.

Let u_0 be an arbitrary member of V. In the case that the forms $B(\cdot; \cdot)$ and $Q(\cdot)$ are sufficiently smooth (twice differentiable in a certain functional sense), it is shown in [25] that the error

$$\varepsilon = Q(u) - Q(u_0) \tag{4}$$

coincides with the sum of a *residual functional*,

$$\mathcal{R}(u_0, p) = F(p) - B(u_0; p), \tag{5}$$

p being the solution of an appropriately defined dual or adjoint problem, and higher order terms involving the error $e_0 = u - u_0$. We examine this result in more detail in section 2.

An important observation is that the trial solution u_0 may itself be the solution of some other *surrogate* mathematical models, characterized by a different semilinear form $B_0(\cdot; \cdot)$ and linear form $F_0(\cdot)$; i.e., u_0 is then the solution of the problem

$$\text{Find } u_0 \in V : \quad B_0(u_0; v) = F_0(v) \quad \forall v \in V. \tag{6}$$

Thus, the fact that the error between the quantity of interest $Q(\cdot)$ computed using the base model (1) and that computed using any other surrogate model (6) can be estimated in some manner provides a means for comparing the appropriateness of different models and, presumably, controlling modeling error relative to the base model. The surrogate problem(s) (6) will therefore correspond to models of coarser (larger) scales than (1). Still, the base model may never be solved; it remains a datum with respect to which various surrogates are measured.

If modeling error can be effectively estimated, the next challenge is to adaptively control it by systematically changing the surrogate models. This is the goal of the Goals algorithms: to adapt the mathematical model so that preset tolerances in the

error in the specific quantity of interest can be met. Here we will describe a general family of algorithms that include earlier methods as special cases (see [28, 50]). The surrogate model that can approximate the base model well enough to lead to a small tolerable error may often be a hybrid consisting of components involving several different spatial or temporal scales. We take this subject up in section 4.

1.3. Notation: Functional derivatives and Taylor formulas. We define the functional or Gâteaux derivative of the semilinear form $B(u;v)$ in (1) as

$$B'(u;w,v) = \lim_{\theta\to 0} \theta^{-1}[B(u+\theta w;v) - B(u;v)]$$

assuming this limit exists. Higher derivatives of $B(u;v)$ are defined analogously:

$$\begin{aligned}
B''(u;w_1,w_2,v) &= \lim_{\theta\to 0} \theta^{-1}[B'(u+\theta w_2;w_1,v) - B'(u;w_1,v)],\\
B'''(u;w_1,w_2,w_3,v) &= \lim_{\theta\to 0} \theta^{-1}[B''(u+\theta w_3;w_1,w_2,v) - B''(u;w_1,w_2,v)]\\
&\vdots
\end{aligned}$$

Thus, for each $u \in V$, $B'(u;w,v)$ is a bilinear form in v and w, $B''(u;w_1,w_2,v)$ is a trilinear form in w_1, w_2, and v, etc. Similarly, if the target output functionals (the quantities of interest Q) are differentiable, we use the following notation for various Gâteaux derivatives:

$$\begin{aligned}
Q'(u;v_1) &= \lim_{\theta\to 0} \theta^{-1}[Q(u+\theta v_1) - Q(u)],\\
Q''(u;v_1,v_2) &= \lim_{\theta\to 0} \theta^{-1}[Q'(u+\theta v_2;v_1) - Q'(u;v_1)],\\
Q'''(u;v_1,v_2,v_3) &= \lim_{\theta\to 0} \theta^{-1}[Q''(u+\theta v_3;v_1,v_2) - Q''(u;v_1,v_2)]\\
&\vdots
\end{aligned}$$

Taylor formulas with integral remainders can easily be constructed for such differentiable functionals and semilinear forms. Among many such expansions, we list as examples the following:

$$\begin{aligned}
Q(u+v) - Q(u) &= \int_0^1 Q'(u+sv;v)ds,\\
Q(u+v) - Q(u) &= Q'(u;v) + \int_0^1 Q''(u+sv;v,v)(1-s)ds,\\
Q(u+v) - Q(u) &= \frac{1}{2}Q'(u;v) + \frac{1}{2}Q'(u+v;v)\\
&\qquad + \frac{1}{2}\int_0^1 Q'''(u+sv;v,v,v)(s-1)s\ ds
\end{aligned}$$

and

$$\begin{aligned}
B(u+w;v) - B(u;v) &= \int_0^1 B'(u+sw;w,v)ds,\\
B(u+w;v) - B(u;v) &= B'(u;w,v) + \int_0^1 B''(u+sw;w,w,v)(1-s)ds.
\end{aligned}$$

We shall say that $B(\cdot;\cdot)$ and $Q(\cdot)$ belong to $C^r(V)$ whenever the limits defining functional derivatives of order r exist. In all of the theory and applications we consider in subsequent sections, V is a reflexive Banach space with norm $\|\cdot\|$. The theory can be extended to more general settings.

2. A posteriori estimates of modeling error. We begin with the observation that the target problem (3) can be viewed as a problem of optimal control: find an extremum of Q subject to the constraint (1):

$$Q(u) = \inf_{v \in M} Q(v); \quad M = \{w \in V;\ B(w;q) = F(q) \quad \forall q \in V\}. \tag{7}$$

The Lagrangian associated with (7) is

$$L(v,q) = Q(v) + F(q) - B(v;q), \tag{8}$$

and, assuming Q and B are Gâteaux differentiable, solutions to the constrained optimization problem (7) are pairs (u,p) such that

$$L'((u,p);(v,q)) = 0 \quad \forall (v,q) \in V \times V;$$

i.e., (u,p) is a solution of the coupled problem

$$\boxed{\begin{array}{l}\text{Find } (u,p) \in V \times V \text{ such that} \\ \qquad B(u;q) = F(q) \qquad \forall q \in V, \\ \qquad B'(u;v,p) = Q'(u;v) \qquad \forall v \in V\end{array}} \tag{9}$$

We refer to $(9)_1$ as the *primal* base problem and to $(9)_2$ as the *dual* base problem. The dual problem is thus always linear in p. The dual solution p is called the *influence vector* (or *function*) or the *generalized Green's function*. In the cases in which $B(\cdot;\cdot)$ is a bilinear form and $Q(\cdot)$ is a linear functional, the dual problem $(9)_2$ reduces to

$$B(v,p) = Q(v) \quad \forall v \in V$$

so that

$$Q(u) = B(u,p) = F(p). \tag{10}$$

This is recognized as a generalization of the property of Green's functions for linear initial- and boundary-value problems.

Now let us suppose that the system (9) is intractable and that we seek an approximation (u_0,p_0) to (u,p). The following theorem [25] provides the error in the quantity of interest Q.

THEOREM 1. *Let the semilinear form $B(\cdot;\cdot)$ in (9) belong to $C^3(V)$ and let the quantity of interest $Q(\cdot) \in C^3(V)$. Let $(u,p) \in V \times V$ be a solution of the base problem (9) and let (u_0,p_0) be an arbitrary pair in $V \times V$. Then the error in $Q(u)$ produced by u_0 is given by*

$$\mathcal{E}(u_0) = Q(u) - Q(u_0) = \mathcal{R}(u_0;p_0) + \mathcal{R}(u_0;\varepsilon_0) + \Delta_1(u_0,p_0,e_0,\varepsilon_0), \tag{11}$$

where $\mathcal{R}(u_0;v)$ is the residual functional,

$$\mathcal{R}(u_0;v) = F(v) - B(u_0;v), \quad v \in V, \tag{12}$$

and $\Delta_1 = \Delta_1(u_0, p_0, e_0, \varepsilon_0)$ *is the remainder,*

$$\begin{aligned}\Delta_1 = \frac{1}{2}\int_0^1 \{&B''(u_0 + se_0; e_0, e_0, p_0 + s\varepsilon_0) \\ &- Q''(u_0 + se_0; e_0, e_0)\}ds \\ + \frac{1}{2}\int_0^1 \{&Q'''(u_0 + se_0; e_0, e_0) - 3B''(u_0 + se_0; e_0, e_0, \varepsilon_0) \\ &- B'''(u_0 + se_0; e_0, e_0, e_0, p_0 + s\varepsilon_0)\}(s-1)sds\end{aligned} \tag{13}$$

with

$$e_0 = u - u_0 \text{ and } \varepsilon_0 = p - p_0. \tag{14}$$

We observe that $\Delta_1 \equiv 0$ whenever $B(\cdot;\cdot)$ is bilinear and $Q(\cdot)$ is linear. In that case, $Q(u) - Q(u_0) = \mathcal{R}(u_0; p_0) + \mathcal{R}(u_0; \varepsilon_0) = \mathcal{R}(u_0; p_0) + B(e_0, \varepsilon_0)$. Equation (11) establishes that the error $\mathcal{E}(u_0)$ is dominated by the residual functional evaluated at the influence functional p_0 to within terms of higher order in the error components (14).

An alternative representation of the error $\mathcal{E}(u_0)$, involving the residuals associated with the primal and dual problems, is given in the following corollary [25].

COROLLARY 1. *Let the assumptions of Theorem* 1 *hold. Then the error in* $Q(u)$ *produced by* u_0 *is given by*

$$\mathcal{E}(u_0) = \mathcal{R}(u_0; p_0) + \frac{1}{2}(\mathcal{R}(u_0; \varepsilon_0) + \bar{\mathcal{R}}(u_0, p_0; e_0)) + \Delta_2(u_0, p_0, e_0, \varepsilon_0), \tag{15}$$

where $\bar{\mathcal{R}}(u_0, p_0; v)$ *is the residual functional for the dual problem*

$$\bar{\mathcal{R}}(u_0, p_0; v) = Q'(u_0; v) - B'(u_0; v, p_0), \quad v \in V, \tag{16}$$

and $\Delta_2 = \Delta_2(u_0, p_0, e_0, \varepsilon_0)$ *is given by*

$$\begin{aligned}\Delta_2 = \frac{1}{2}\int_0^1 \{&Q'''(u_0 + se_0; e_0, e_0) - 3B''(u_0 + se_0; e_0, e_0, \varepsilon_0) \\ &- B'''(u_0 + se_0; e_0, e_0, e_0, p_0 + s\varepsilon_0)\}(s-1)sds.\end{aligned} \tag{17}$$

We note that the remainder in (17) should be smaller in practical applications than the one in (11) at the expense of having to determine or estimate both e_0 and ε_0. Results similar to those in Theorem 1 and Corollary 1 for finite element approximation errors were presented in Oden and Prudhomme [24] and Becker and Rannacher [4]. A detailed account of the versions of the methods of [4] is given in [3].

It is emphasized that the surrogate functions (u_0, p_0) in (11)–(14) are completely arbitrary vectors in $V \times V$. However, the most natural choice of these vectors are those which are solutions of a *surrogate problem* of the form

$$\boxed{\begin{array}{l}\text{Find } (u_0, p_0) \in V \times V \text{ such that} \\ \qquad B_0(u_0; v) = F_0(v) \qquad \forall\, v \in V, \\ \qquad B_0'(u_0; v, p_0) = Q_0'(u_0; v) \qquad \forall\, v \in V\end{array}} \tag{18}$$

where $B_0(\cdot;\cdot)$ is a semilinear form for a coarser-scale problem and $F_0(\cdot)$ and $Q_0(\cdot)$ are coarse-scale approximations of $F(\cdot)$ and $Q(\cdot)$. In most cases, B_0, F_0, and Q_0 are obtained through an appropriate homogenization or averaging process. Thus, (18) could represent models of events that occur at scales larger than those of the base model. Theorem 1 then provides a means for comparing models of different scales.

3. Homogenization and construction of surrogates.

3.1. Some general remarks. Symbolically, one can envision a multiscale model in which the quantity of interest $Q(u)$ of the base model is the target output, $Q(u_1)$ is the approximation of that output produced by a coarser-scale model obtained through a homogenization process, $Q(u_2)$ is the approximation due to a still coarser scale, and so on. If u_N denotes the surrogate solution to the coarsest-scale model, the various modeling error components are

$$Q(u) - Q(u_N) = \underbrace{Q(u) - Q(u_1)}_{\text{error in scale 1 model}} + \underbrace{Q(u_1) - Q(u_2)}_{\substack{\text{error in scale 2 model} \\ \text{relative to scale 1}}} + \cdots + \underbrace{Q(u_{N-1}) - Q(u_N)}_{\substack{\text{error in scale } N \text{ model} \\ \text{relative to scale } N-1}}.$$

Each error component, of course, depends upon how the successive coarser models are obtained through a homogenization process. As will be seen in applications discussed later, if a target error level is specified, the models that can ultimately deliver that accuracy may involve a mosaic of component models of several different scales.

It can be argued that the limit processes carrying models of one scale to those of larger scales, such as the limiting process from molecular models to continuum models, is a central problem to the mathematical foundations of multiscale modeling. It is a subject in which some progress has been made in recent years (see, e.g., [6, 15]) but for which much additional work remains to be done. It is a remarkable fact, however, that even when these processes are not understood and when only crude averaging procedures are employed, the adaptive modeling algorithms described later may still produce acceptable results, albeit at slow rates of convergence.

Finally, we note that in all applications we are only able to compute a numerical approximation u_0^h of any particular surrogate problem solution u_0. However, we can write

$$\underbrace{Q(u) - Q(u_0^h)}_{\text{total error}} = \underbrace{Q(u) - Q(u_0)}_{\text{modeling error}} + \underbrace{Q(u_0) - Q(u_0^h)}_{\text{approximation error}}.$$

The total error $Q(u) - Q(u_0^h)$ can be estimated with the aid of Theorem 1 using u_0^h instead of u_0. The approximation error $Q(u_0) - Q(u_0^h)$ can be accurately estimated and controlled using the theory and methodology discussed in [24] (see also [1, 4, 3, 27]). Finally, an estimate of the modeling error is obtained by substracting the approximation error from the total error.

It should be understood that while models involving an ascending sequence of scales can, in theory, be obtained through some sort of ensemble averaging (homogenization) process, the actual surrogate pairs (u_0, p_0) appearing in the error estimate (11) are not necessarily solutions of a homogenized problem defined for coarser-scale models. In fact, the coarse-scale models obtained through homogenization processes may even belong to different function spaces than those appearing in the base problem. For example, if the base model is discrete, such as is the case in which the base model corresponds to a crystalline lattice model or an MD model, and a homogenization process produces a continuum model, then the continuum model cannot be used as the surrogate within the context of our theory. Another step is needed.

Let the following denote a model obtained from the base model through some homogenization

$$\begin{aligned} &(\tilde{u}, \tilde{p}) \in V_0 \times V_0 : \\ &\qquad \tilde{B}(\tilde{u}; \tilde{v}) = \tilde{F}(\tilde{v}) \qquad \forall \tilde{v} \in V_0, \\ &\qquad \tilde{B}'(\tilde{u}; \tilde{v}, \tilde{p}) = \tilde{Q}'(\tilde{u}; \tilde{v}) \qquad \forall \tilde{v} \in V_0, \end{aligned}$$

where $\tilde{B}(\cdot;\cdot)$, $\tilde{F}(\cdot)$, and $\tilde{Q}(\cdot)$ are functionals produced from $B(\cdot,\cdot)$, $F(\cdot)$, and $Q(\cdot)$ through homogenization. Let $\Pi(\cdot)$ denote a map,

$$\Pi:\ V_0 \longrightarrow V; \quad \Pi\tilde{u} = u_0,\ \Pi\tilde{p} = p_0.$$

Then, for any appropriate choice of Π, we use (u_0, p_0) as the solutions of the surrogate models, and the error in the quantity of interest is estimated with respect to $Q(u_0)$. We provide examples of such surrogates in section 6.

3.2. Fine-scale fluctuations about a mean. Various methods for treating the interactions of events at two or more scales begin with the separation of the fine-scale solution u into a "mean" component $\bar{u}$ and a fluctuation u' about the mean (see, e.g., [47, 48])

$$u = \bar{u} + u'. \tag{19}$$

Comparing this to our notation, if $V_0 \subset V$, the component $\bar{u}$ and u' can be identified with u_0 and $e_0 = u - u_0$ as

$$\bar{u} = u_0 \quad \text{and} \quad u' = e_0. \tag{20}$$

Using the developments laid down thus far, it is a straightforward exercise to determine approximate relationships for coarse-scale/fine-scale interactions and for computing the fluctuations.

Ignoring higher order terms in the component u' (assumed small compared to $\bar{u}$), we have, for any $v \in V$,

$$F(v) = B(\bar{u} + u'; v) \approx B(\bar{u}; v) + B'(\bar{u}; u', v).$$

Thus

$$\boxed{B'(\bar{u}; u', v) \approx \mathcal{R}(\bar{u}; v) \qquad \forall v \in V} \tag{21}$$

That is, the fluctuations u' satisfy the transpose of the dual problem with u replaced by $\bar{u}$ and with $Q'(u; v)$ replaced by the residual functional $\mathcal{R}(\bar{u}; v)$. Then, with p the solution of the corresponding primal dual problem,

$$\boxed{B'(\bar{u}; u', p) \approx \mathcal{R}(\bar{u}; p) \approx Q(u) - Q(\bar{u})} \tag{22}$$

The result establishes a relationship between the fine-scale fluctuations u' and the error $\mathcal{E}(\bar{u})$ in the quantity of interest. The fine-scale fluctuations thus affect the error in the quantity of interest at positions of the space-time domain within the support of the dual solution p.

At the heart of many multiscale methods is the particular homogenization strategy used to compute the mean component $\bar{u}$. If $\Pi : V_0 \to V$ is a given extension of the space containing the surrogate solution such that $\Pi u_0 = \bar{u}$, then one can construct the "filter," $Pu = \bar{u}$, so that the fluctuations $u - Pu = u'$ are minimized in some sense. We describe one such approach in section 6.

4. The Goals algorithms. The Goals algorithms, introduced in [28, 50] and implemented in various versions in [26, 36, 37], are designed to reduce estimated modeling error by systematically adapting the model using a sequence of surrogates. In [28], reference is made to the "Goal-Oriented Adaptive Local Solutions" algorithms; hence we have the term "Goals." We refer to these general algorithms as Goals or goal-oriented algorithms. As will be shown, not all employ "local" solutions.

The general ideas can be described in the following setting:

1. Let the domain of the solution $(u, p) \in V \times V$ of the base problem (9) be an open region $D \subset \mathbb{R}^d$, and let $\{\Theta_j\}_{j=0}^k$ be a partition of D:

$$\overline{D} = \bigcup_{j=0}^{k} \overline{\Theta}_j, \quad \Theta_i \bigcap \Theta_j = \varnothing, \quad i \neq j,\ 0 \leq i,\ j \leq k.$$

The semilinear and linear forms in (9) can be broken into $k+1$ components so that

$$B(u; v) = \sum_{j=0}^{k} B_{\Theta_j}(u; v), \quad F(v) = \sum_{j=0}^{k} F_{\Theta_j}(v) \tag{23}$$

for $u, v \in V$, with $B_{\Theta_j}(\cdot;\cdot)$, $F_{\Theta_j}(\cdot)$ being the values of the forms produced by restrictions of u and v to Θ_j.
2. Next, through an averaging or homogenization process, we construct a surrogate problem of the form (18) and compute the surrogate solutions $(u_0, p_0) \in V \times V$ defined over all of D.
3. To assess the accuracy of the surrogate solution, we seek to estimate the modeling error in the quantity of interest and call upon Theorem 1 to note that to within higher order terms,

$$\mathcal{E}(u_0) \approx \mathcal{R}(u_0, p) = \mathcal{R}(u_0; p_0) + \mathcal{R}(u_0; p - p_0). \tag{24}$$

The first term is computable, whereas the remaining term is generally intractable and has to be estimated. However, the derivation of an estimate depends heavily on the type of problem that is analyzed. Examples of error estimates for problems in molecular statics, elastostatics, and elastodynamics are given in [26, 28, 37, 36].
4. If the error estimate is to within a user defined error tolerance δ_{TOL}, the analysis stops and provides the analyst with the prediction of the quantity of interest $Q(u_0)$.
5. If the error exceeds the tolerance, the algorithm proceeds by enhancing or improving the surrogate model in a subdomain $D_L \subset D$, called the *domain of influence*. Suppose that the quantity of interest Q involves features of the solution confined to a subdomain $D_Q \subset D$ (e.g., $Q(u)$ could be the average of u or its derivatives over D_Q). Then the initial choice of D_L is generally the union of subdomains Θ_i that intersect with D_Q.
6. There are several possibilities to then compute a correction $(\tilde{u}, \tilde{p})$ of (u_0, p_0) using the enhanced surrogate model.
   I. *Global Goals algorithms.* Let

$$\tilde{B}(\tilde{u}; v) = B_{D_L}(\tilde{u}_{D_L}; v_{D_L}) + B_{0,D\backslash D_L}(\tilde{u}_{D\backslash D_L}; v_{D\backslash D_L}), \tag{25}$$

where $\tilde{u}_{D_L} = \tilde{u}|_{D_L}$, etc., and $B_{0,D\backslash D_L}(\cdot;\cdot)$ is the semilinear form for the surrogate problem defined on restrictions of functions in V to $D\backslash D_L$. Thus, $\tilde{B}(\cdot,\cdot)$ includes fine-scale features of the base model over domain D_L while it is characterized by coarse-scale features of the surrogate model over $D\backslash D_L$. We calculate $(\tilde{u}, \tilde{p}) \in V \times V$ such that

$$\begin{aligned} &\tilde{B}(\tilde{u}; v) = F(v) \quad \forall v \in V, \\ &\tilde{B}'(\tilde{u}; v, \tilde{p}) = Q'(\tilde{u}; v) \quad \forall v \in V. \end{aligned} \tag{26}$$

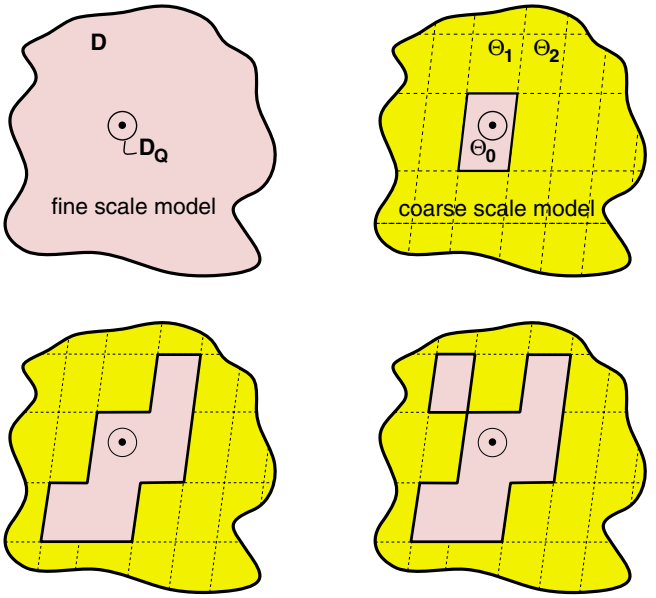


FIG. 1. *(Top left) A fine-scale base model and a subdomain D_Q surrounding features of a quantity of interest Q; (top right) a partition of Ω into subdomains with an initial domain of influence $D = \Theta_0$; (bottom) successive enlargements of D designed to reduce the modeling error.*

II. *Local Goals algorithms.* In this case, we compute a correction $(\tilde{u}, \tilde{p})$ that coincides with (u_0, p_0) outside the initial domain of influence but uses the base problem data (the "fine-scale data") within D_L:

$$\begin{aligned} &B_{D_L}(\tilde{u}; v) = F_{D_L}(v) \quad \forall v \in V(D_L) \quad \text{with} \quad \tilde{u} = u_0 \text{ in } \overline{D}\backslash D_L, \\ &B'_{D_L}(\tilde{u}; v, \tilde{p}) = Q'_{D_L}(\tilde{u}; v) \quad \forall v \in V(D_L) \quad \text{with} \quad \tilde{p} = p_0 \text{ in } \overline{D}\backslash D_L \end{aligned} \tag{27}$$

and $V(D_L)$ is the space of restrictions of functions in V to the domain of influence D_L. The situation is illustrated in Figure 1.

7. As done in step 3, we can derive an estimate of the modeling error of the enhanced solution $\tilde{u}$,

$$\mathcal{E}(\tilde{u}) \approx \mathcal{R}(\tilde{u}; p).$$

Again, the estimation of this term depends on the type of problem (e.g., see [28, 36]). In this work, we introduce an alternative estimator by noting that

$$\mathcal{E}(\tilde{u}) = Q(u) - Q(\tilde{u}) = [Q(u) - Q(u_0)] + [Q(u_0) - Q(\tilde{u})]. \tag{28}$$

The last term in this equation is computable and the first can be estimated by again using Theorem 1:

$$\begin{aligned} \mathcal{E}(\tilde{u}) &\approx \mathcal{R}(u_0; p) + Q(u_0) - Q(\tilde{u}) \\ &= \mathcal{R}(u_0; \tilde{p}) + \mathcal{R}(u_0; p - \tilde{p}) + Q(u_0) - Q(\tilde{u}) \\ &\approx \mathcal{R}(u_0; \tilde{p}) + Q(u_0) - Q(\tilde{u}). \end{aligned} \tag{29}$$

8. If the error estimate is within a user defined error tolerance δ_{TOL}, the analysis stops and provides the analyst with the prediction in the quantity of interest $Q(\tilde{u})$.

9. If the error exceeds the tolerance, we obtain an indication of the error contribution of each subdomain (outside the domain of influence) by computing its contribution to the upper bound of the residual $\mathcal{R}(u_0; p)$ (e.g., see [28, 36, 37]). We then expand the domain of influence D_L by adding the subdomains $\{\Theta_i\}$ whose error contributions, with respect to the maximal contribution, exceed a user defined tolerance γ and return to step 6.

Steps 1–9 characterize a family of Global and Local Goals algorithms for adaptive modeling. Several remarks are in order.

REMARK 1. *While the global versions of the Goals algorithms may involve the solution of surrogate problems of a larger size than the local solution algorithms, they have the advantage of accounting for possible interactions of fine-scale and large-scale behavior. Thus, we expect that they may lead to faster convergence rates than the simpler local methods.*

REMARK 2. *The initial domain of interest, of course, can be a multiconnected domain, depending upon the number and structure of the quantities of interest.*

5. Global estimates. The quality of goal-oriented error estimates, as can be seen in Theorem 1, strongly depends on the accuracy of the coarse solutions u_0 and p_0. We show here how the errors e_0 $(= u - u_0)$ and ε_0 $(= p - p_0)$ are related to the norms of the residuals. Global estimates of modeling error can be obtained for specific forms of $B(\cdot;\cdot)$ and $Q(\cdot)$. In cases in which $B(\cdot;\cdot)$ is a symmetric, positive-definite, bilinear form, global estimates of e_0 and ε_0 can be used to construct upper and lower bounds of errors in quantities of interest [28, 50, 36]. Here we provide a derivation of such bounds for cases in which $B(\cdot;\cdot)$ and $Q(\cdot)$ satisfy a priori conditions.

Let V be a Banach space with norm $\|\cdot\|$. We observe that

$$\begin{aligned}
&B(u+w;v) = B(u;v) + B'(u;w,v) + \Delta_4(u,w,v),\\
&Q'(u+w;v) = Q'(u;v) + \Delta_5(u,w,v),\\
&B'(u+w_2;w_1,v) = B'(u;w_1,v) + \Delta_6(u,w_1,w_2,v),
\end{aligned}$$

where

$$\begin{aligned}
\Delta_4(u,w,v) &= \int_0^1 B''(u+sw;w,w,v)(1-s)ds,\\
\Delta_5(u,w,v) &= \int_0^1 Q''(u+sw;w,v)ds,\\
\Delta_6(u,w_1,w_2,v) &= \int_0^1 B''(u+sw_2;w_1,w_2,v)ds
\end{aligned}$$

for all u, v, w, w_1, and w_2 in V. With this notation, and recalling that $u = u_0 + e_0$ and $p = p_0 + \varepsilon_0$, we have

$$\left.\begin{aligned}
&B(u;v) = B(u_0;v) + B'(u_0;e_0,v) + \Delta_4(u_0,e_0,v)\\
&Q'(u;v) = Q'(u_0;v) + \Delta_5(u_0,e_0,v)\\
&B'(u;v,p) = B'(u_0;v,p_0) + B'(u_0;v,\varepsilon_0) + \Delta_6(u_0,v,e_0,p_0+\varepsilon_0)
\end{aligned}\right\}. \tag{30}$$

Then the primal and dual base models can be written as

$$\boxed{\begin{aligned}
&\text{Find } (e_0,\varepsilon_0) \in V\times V \text{ such that}\\
&\quad B(u_0;v) + B'(u_0;e_0,v) + \Delta_4(u_0,e_0,v) = F(v) \qquad \forall v\in V,\\
&\quad B'(u_0;v,p_0) + B'(u_0;v,\varepsilon_0) + \Delta_6(u_0,v,e_0,p_0+\varepsilon_0)\\
&\qquad\qquad = Q'(u_0;v) + \Delta_5(u_0,e_0,v) \qquad \forall v\in V
\end{aligned}} \tag{31}$$

Thus, the errors e_0 and ε_0 satisfy the following equations:

$$
\begin{aligned}
&B'(u_0;e_0,v)+\Delta_4(u_0,e_0,v)=\mathcal{R}(u_0;v) \qquad \forall v\in V,\\
&B'(u_0;v,\varepsilon_0)+\Delta_6(u_0,v,e_0,p_0+\varepsilon_0)\\
&\qquad\qquad =\bar{\mathcal{R}}(u_0,p_0;v)+\Delta_5(u_0,e_0,v) \qquad \forall v\in V,
\end{aligned}
\tag{32}
$$

where $\mathcal{R}(u_0;v)$ and $\bar{\mathcal{R}}(u_0,p_0;v)$ are the residual functionals in (12) and (16).

If $B(\cdot;\cdot)$ and $Q(\cdot)$ are a bilinear form and a linear form, respectively, then Δ_4, Δ_5, and Δ_6 vanish, $B'=B$, $Q'=Q$, so that the errors are governed by

$$
\begin{aligned}
B(e_0,v)&=\mathcal{R}(u_0;v) \qquad \forall v\in V,\\
B(v,\varepsilon_0)&=\bar{\mathcal{R}}(u_0,p_0;v) \qquad \forall v\in V.
\end{aligned}
$$

Otherwise, we shall assume that constants $C_4=C_4(u_0)$, $C_5=C_5(u_0)$, and $C_6=C_6(u_0,p_0)$ exist such that

$$
\left.
\begin{aligned}
&|\Delta_4(u_0,e_0,v)|\le C_4\|e_0\|^2\|v\|\\
&|\Delta_5(u_0,e_0,v)|\le C_5\|e_0\|\,\|v\|\\
&|\Delta_6(u_0,v,e_0,p_0+\varepsilon_0)|\le C_6\|e_0\|\,\|v\|
\end{aligned}
\right\}
\tag{33}
$$

and that it is possible to find a $u_0\in V$ such that $\infty>\alpha(u_0),\beta(u_0)>0$, where $\alpha(u_0)$ and $\beta(u_0)$ are given by

$$
\begin{aligned}
\alpha(u_0)&=\inf_{w\in V\setminus\{0\}}\sup_{v\in V\setminus\{0\}}\frac{|B'(u_0;w,v)|}{\|w\|\,\|v\|},\\
\beta(u_0)&=\inf_{w\in V\setminus\{0\}}\sup_{v\in V\setminus\{0\}}\frac{|B'(u_0;v,w)|}{\|w\|\,\|v\|}.
\end{aligned}
$$

Then we easily obtain

$$
\begin{aligned}
\alpha(u_0)\|e_0\|&\le\|\mathcal{R}(u_0)\|_*+C_4\|e_0\|^2,\\
\beta(u_0)\|\varepsilon_0\|&\le\|\bar{\mathcal{R}}(u_0,p_0)\|_*+(C_5+C_6)\|e_0\|,
\end{aligned}
$$

where $\|\cdot\|_*$ denotes the norm on the dual space V':

$$
\begin{aligned}
\|\mathcal{R}(u_0)\|_*&=\sup_{v\in V\setminus\{0\}}\frac{|\mathcal{R}(u_0;v)|}{\|v\|},\\
\|\bar{\mathcal{R}}(u_0,p_0)\|_*&=\sup_{v\in V\setminus\{0\}}\frac{|\bar{\mathcal{R}}(u_0,p_0;v)|}{\|v\|}.
\end{aligned}
$$

These results show that for e_0 sufficiently small, when the assumed properties prevail, the modeling errors e_0 and ε_0 are globally bounded by the norms of the residuals.

6. Examples.

6.1. Random heterogeneous materials. We now show an application of the Local Goals algorithms to the analysis of the elastostatics of *random* multiphase composite materials (see also [37]). In section 6.1.1, we first define the model problem and notation. We then introduce a brief overview on the surrogate models and the estimation of modeling errors in section 6.1.2 and present a specific example application to a two-phase composite material in section 6.1.3.

6.1.1. Model problem and notation. We consider a material body, occupying an open and bounded domain $D \subset \mathbb{R}^d$, $d = 1, 2, 3$, with boundary $\partial D = \overline{\Gamma_u \cup \Gamma_t}$, $\Gamma_u \cap \Gamma_t = \varnothing$, $meas(\Gamma_t) > 0$, $meas(\Gamma_u) > 0$, Γ_u and Γ_t being portions of ∂D on which displacements and tractions are to be specified, respectively. The body is in static equilibrium under the action of deterministic applied forces $\boldsymbol{f} \in L^2(D)^d$, a surface traction $\boldsymbol{t} \in L^2(\Gamma_t)^{d-1}$, and a prescribed displacement $\boldsymbol{U} \in L^2(\Gamma_u)^d$ on Γ_u. We assume the body is composed of a multiphase, composite, elastic material with highly oscillatory material properties. The geometrical features and material properties of its constituents are functions of a *random vector* $\boldsymbol{\omega} \in \Omega$, where Ω denotes the collection of all possible realizations. The governing *probability distribution density function* $\rho : \Omega \longrightarrow [0, 1]$ is assumed to be known.

Let $\boldsymbol{u} = \boldsymbol{u}(\boldsymbol{x}, \boldsymbol{\omega})$ denote the random displacement vector field defined on $D \times \Omega$ and let $\boldsymbol{\nabla u}$ denote the spatial gradient. Then the Cauchy stress tensor $\boldsymbol{\sigma}(\boldsymbol{x}, \boldsymbol{\omega})$ satisfies $\boldsymbol{\sigma}(\boldsymbol{x}, \boldsymbol{\omega}) = \boldsymbol{E}(\boldsymbol{x}, \boldsymbol{\omega})\boldsymbol{\nabla u}(\boldsymbol{x}, \boldsymbol{\omega})$, where $\boldsymbol{E}(\boldsymbol{x}, \boldsymbol{\omega})$ is the fourth order tensor of elasticities with components $E_{ijkl}(\boldsymbol{x}, \boldsymbol{\omega}) \in L^\infty(D, L^2(\Omega))^{d \times d}$. The standard symmetry and ellipticity conditions hold for a.e. $\boldsymbol{x} \in D$ and a.s. $\boldsymbol{\omega} \in \Omega$:

$$E_{ijkl}(\boldsymbol{x}, \boldsymbol{\omega}) = E_{jikl}(\boldsymbol{x}, \boldsymbol{\omega}) = E_{ijlk}(\boldsymbol{x}, \boldsymbol{\omega}) = E_{klij}(\boldsymbol{x}, \boldsymbol{\omega}),$$
$$\alpha_0 \xi_{ij} \xi_{ij} \le E_{ijkl}(\boldsymbol{x}, \boldsymbol{\omega}) \xi_{ij} \xi_{kl} \le \alpha_1 \xi_{ij} \xi_{ij}, \qquad \alpha_1 \ge \alpha_0 > 0.$$

We introduce a space of test functions V,

$$V = \left\{ \boldsymbol{v} \in H^1(D, L^2(\Omega)) : \ \boldsymbol{v}(\boldsymbol{x}, \boldsymbol{\omega}) = 0 \ \ \forall \boldsymbol{x} \in \Gamma_u, \ \boldsymbol{\omega} \in \Omega \right\}, \tag{34}$$

where $H^1(D, L^2(\Omega))$ is the Hilbert space,

$$H^1(D, L^2(\Omega)) = \left\{ \boldsymbol{v}(\boldsymbol{x}, \boldsymbol{\omega}) : \ \int_\Omega \int_D \left[\boldsymbol{\nabla v} : \boldsymbol{\nabla v} + \boldsymbol{v} \cdot \boldsymbol{v} \right] \ \rho(\boldsymbol{\omega}) \, \mathrm{d}\boldsymbol{x} \, \mathrm{d}\boldsymbol{\omega} < \infty \right\}.$$

Having established the necessary notations and conventions, we now recall the variational formulation (9) of our linear, stochastic, elastostatics problem, where [37]

$$\begin{aligned} F(\boldsymbol{v}) &= \int_\Omega \int_D (\boldsymbol{f} : \boldsymbol{v}) \ \rho(\boldsymbol{\omega}) \ \mathrm{d}\boldsymbol{x} \, \mathrm{d}\boldsymbol{\omega} + \int_\Omega \int_{\Gamma_t} (\boldsymbol{t} : \boldsymbol{v}) \ \rho(\boldsymbol{\omega}) \ \mathrm{d}s \, \mathrm{d}\boldsymbol{\omega}, \\ B(\boldsymbol{u}, \boldsymbol{v}) &= \int_\Omega \int_D (\boldsymbol{E} \boldsymbol{\nabla u} : \boldsymbol{\nabla v}) \ \rho(\boldsymbol{\omega}) \, \mathrm{d}\boldsymbol{x} \, \mathrm{d}\boldsymbol{\omega}, \end{aligned} \tag{35}$$

where $\boldsymbol{n}$ and $\boldsymbol{s}$ denote the unit normal and position vector on ∂D, respectively. Following the approach in sections 1.2 and 2, we define the target of the analysis as a linear functional $Q(\cdot)$ of the solution $\boldsymbol{u}$.

6.1.2. Surrogate models and modeling errors. To implement versions of the Goals algorithms, we compute a surrogate solution pair $(\boldsymbol{u}_0, \boldsymbol{p}_0)$ of (18), where the bilinear form $B_0(\cdot, \cdot)$ is obtained by first computing the statistical mean $\bar{\boldsymbol{E}}(\boldsymbol{x})$ of $\boldsymbol{E}(\boldsymbol{x}, \boldsymbol{\omega})$ and by then using standard homogenization methods or classical Hashin–Shtrikman bounds [16] to obtain the *deterministic*, *homogeneous*, elasticity tensor $\boldsymbol{E}_0$. Thus,

$$B_0(\boldsymbol{u}_0, \boldsymbol{v}) = \int_\Omega \int_D (\boldsymbol{E}_0 \boldsymbol{\nabla u}_0 : \boldsymbol{\nabla v}) \ \rho(\boldsymbol{\omega}) \, \mathrm{d}\boldsymbol{x} \, \mathrm{d}\boldsymbol{\omega}. \tag{36}$$

By recalling the surrogate primal problem $(18)_1$ and the fact that the applied external forces $\mathbf{f}$ and $\mathbf{t}$ are deterministic, the above choice for the bilinear form $B_0(\cdot, \cdot)$ implies

that $\mathbf{u}_0$ is deterministic. This does not generally extend to the dual solution $\mathbf{p}_0$, which is deterministic only for certain choices of quantities of interest. The incurred modeling errors are denoted as $\boldsymbol{e}_0(\boldsymbol{x},\boldsymbol{\omega}) = \boldsymbol{u}(\boldsymbol{x},\boldsymbol{\omega}) - \boldsymbol{u}_0(\boldsymbol{x})$ and $\boldsymbol{\varepsilon}_0(\boldsymbol{x},\boldsymbol{\omega}) = \boldsymbol{p}(\boldsymbol{x},\boldsymbol{\omega}) - \boldsymbol{p}_0(\boldsymbol{x},\boldsymbol{\omega})$. The corresponding residual functionals (12) and (16) reduce to [37]

$$
\begin{aligned}
\mathcal{R}(\boldsymbol{u}_0,\boldsymbol{v}) &= -\int_\Omega\int_D (\boldsymbol{E}\mathcal{I}_0\boldsymbol{\nabla}\boldsymbol{u}_0 : \boldsymbol{\nabla}\boldsymbol{v})\ \rho(\boldsymbol{\omega})\,\mathrm{d}\boldsymbol{x}\,\mathrm{d}\boldsymbol{\omega},\\
\overline{\mathcal{R}}(\boldsymbol{p}_0,\boldsymbol{z}) &= -\int_\Omega\int_D (\boldsymbol{E}\mathcal{I}_0\boldsymbol{\nabla}\boldsymbol{z} : \boldsymbol{\nabla}\boldsymbol{p}_0)\ \rho(\boldsymbol{\omega})\,\mathrm{d}\boldsymbol{x}\,\mathrm{d}\boldsymbol{\omega},
\end{aligned}
\tag{37}
$$

where $\mathcal{I}_0(\boldsymbol{x},\boldsymbol{\omega}) = \boldsymbol{I} - \boldsymbol{E}^{-1}\boldsymbol{E}_0$ denotes the *deviation tensor*. We can now call upon Corollary 1 to establish the error in the quantity of interest,

$$
\mathcal{E}(\boldsymbol{u}_0) = \mathcal{R}(\boldsymbol{u}_0;\boldsymbol{p}_0) + \frac{1}{2}(\mathcal{R}(\boldsymbol{u}_0,\boldsymbol{\varepsilon}_0) + \overline{\mathcal{R}}(\boldsymbol{p}_0,\boldsymbol{e}_0)) + \Delta_2(\boldsymbol{u}_0,\boldsymbol{p}_0,\boldsymbol{e}_0,\boldsymbol{\varepsilon}_0).
$$

Ignoring the higher order term $\Delta_2(\boldsymbol{u}_0,\boldsymbol{p}_0,\boldsymbol{e}_0,\boldsymbol{\varepsilon}_0)$, we consider the estimate,

$$
\mathcal{E}(\boldsymbol{u}_0) \approx \mathcal{R}(\boldsymbol{u}_0;\boldsymbol{p}_0) + \frac{1}{2}(\mathcal{R}(\boldsymbol{u}_0,\boldsymbol{\varepsilon}_0) + \overline{\mathcal{R}}(\boldsymbol{p}_0,\boldsymbol{e}_0)).
\tag{38}
$$

The last two terms in the RHS of (38) are generally intractable and further steps need to be taken to derive a computable estimate of the error. We here introduce an example of an estimator derived in [37] which is used in the example problem in section 6.1.3:

$$
\boxed{\mathcal{E}(\boldsymbol{u}_0) \approx \eta_{\mathrm{est,upp}} = \mathcal{R}(\boldsymbol{u}_0,\boldsymbol{p}_0) + \frac{1}{4}(\eta^+_{\mathrm{upp}})^2 - \frac{1}{4}(\eta^-_{\mathrm{upp}})^2}
$$

where

$$
\eta^\pm_{\mathrm{upp}} = \sqrt{\int_\Omega\int_D \boldsymbol{E}\mathcal{I}_0\boldsymbol{\nabla}(\boldsymbol{u}_0 \pm \boldsymbol{p}_0) : \mathcal{I}_0\boldsymbol{\nabla}(\boldsymbol{u}_0 \pm \boldsymbol{p}_0)\ \rho(\boldsymbol{\omega})\,\mathrm{d}\boldsymbol{x}\,\mathrm{d}\boldsymbol{\omega}}.
$$

The Goals algorithm provides two approaches to reduce the error $\mathcal{E}(\boldsymbol{u}_0)$ and compute enhanced solutions $(\tilde{\boldsymbol{u}},\tilde{\boldsymbol{p}})$ of $(\boldsymbol{u}_0,\boldsymbol{p}_0)$: the Local or the Global Goals version (see section 4). In either case, we use the estimator $\tilde{\eta}_{\mathrm{est}}$, derived from (29), to assess the error in the quantity of interest of the enhanced solutions,

$$
\mathcal{E}(\tilde{\boldsymbol{u}}) \approx \tilde{\eta}_{\mathrm{est}} = \mathcal{R}(\boldsymbol{u}_0,\tilde{\boldsymbol{p}}) + Q(\boldsymbol{u}_0) - Q(\tilde{\boldsymbol{u}}).
$$

6.1.3. Numerical example. For the base problem, we consider a two-phase composite material consisting of isotropic, linearly elastic constituents. Cylindrical inclusions account for 30 percent of the volume and have been randomly dispersed in the matrix material, as depicted in Figure 2. The structure is subjected to a traction $\boldsymbol{t}$ at the top and has prescribed zero displacements at the base $y = 0$. The Young's modulus and Poisson ratio of the matrix material are deterministic and given by $E_{\mathrm{m}} = 5$ GPa and $\nu_{\mathrm{m}} = 0.345$. The material properties of the inclusions, however, are functions of the random vector $\boldsymbol{\omega} = \{\omega_1,\omega_2\} \in (0,1)\times(0,1)$, such that

$$
\begin{aligned}
E_{\mathrm{incl}}(\boldsymbol{\omega}) &= E_{\mathrm{incl}}(\omega_1) = 110 + 20\times\omega_1, \quad [\mathrm{GPa}]\\
\nu_{\mathrm{incl}}(\boldsymbol{\omega}) &= \nu_{\mathrm{incl}}(\omega_2) = 0.28 + 0.4\times\omega_2.
\end{aligned}
$$

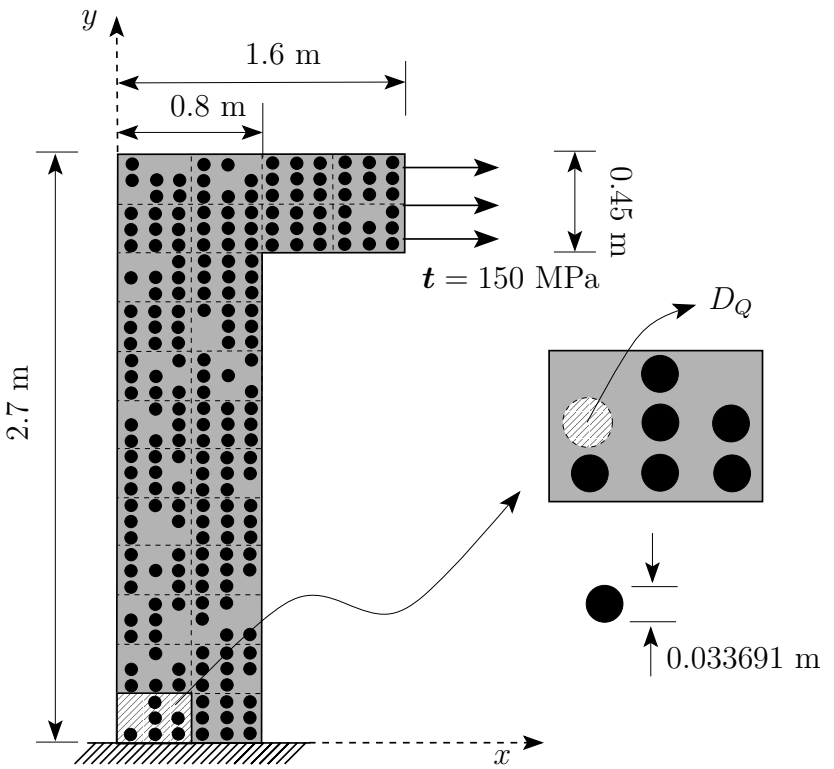


FIG. 2. *A structural component constructed from a two-phase composite material in equilibrium under the action of applied stress* $\boldsymbol{t}$ *and prescribed zero displacements at* $y = 0$.

We assume that ω_1 and ω_2 are statistically independent which implies that $\rho(\boldsymbol{\omega}) = \rho_1(\omega_1)\ \rho_2(\omega_2)$. We additionally assume that $\rho_i(\omega_i)$, $i = 1, 2$, are *truncated Gaussian distribution density functions*,

$$\rho_i(\omega_i) = 2\,\frac{A}{\sqrt{2\pi}}\ e^{-(2\omega_i - 1)^2/2\pi}, \quad i = 1, 2,$$
$$A = 2\left[\operatorname{erf}\left(\frac{1}{\sqrt{2\pi}}\right) - \operatorname{erf}\left(-\frac{1}{\sqrt{2\pi}}\right)\right]^{-1}.$$

In this example, our target $Q(\boldsymbol{u})$ is the statistical average of the average strain ε_{yy} over a small circular area D_Q near the support of the structure, with a radius $r = 0.017$m (see Figure 2).

We establish the surrogate problem (18) by computing the average of the Hashin-Shtrikman upper and lower bounds [16] of the material coefficients in the base model. Using these averaged coefficients, we construct a surrogate $\boldsymbol{E}_0$ whose coefficients are obviously *homogeneous and deterministic*. It is easy to show [38] that with the current choice of $Q(\mathbf{u})$ not only $\boldsymbol{u}_0$ but also $\boldsymbol{p}_0$ is deterministic. In Figure 3, the strain field ε^0_{yy} is shown which is obtained by using overkill finite element approximations of $(\boldsymbol{u}_0, \boldsymbol{p}_0)$ (the finite element models were implemented using the commercial package ProPHLEX [2]).

For comparison, the base solution $\boldsymbol{u}(\boldsymbol{x}, \omega)$ has also been computed by using overkill Monte Carlo and finite element approximations and the statistical average

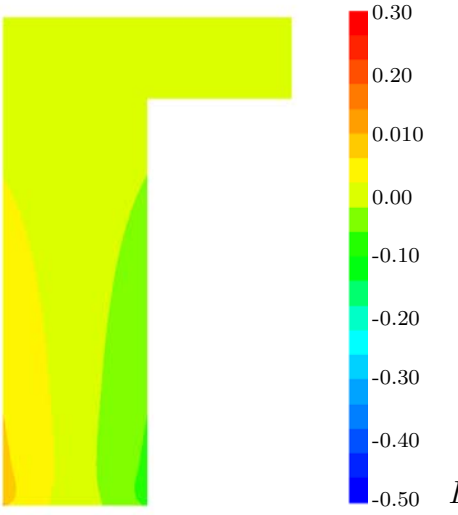


FIG. 3. *Strain field ε^0_{yy} for the deterministic surrogate problem.*

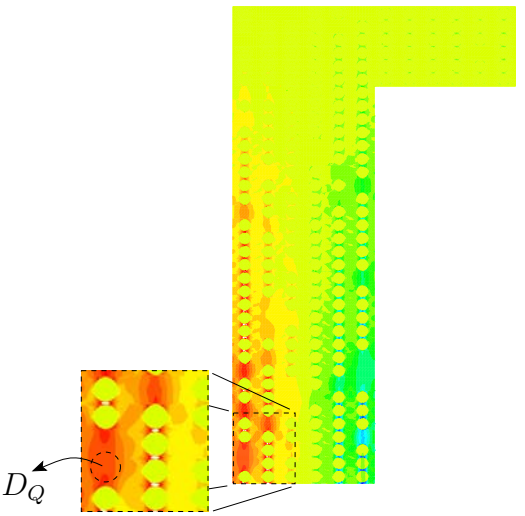


FIG. 4. *Mean value of the strain field ε_{yy} for the base problem.*

TABLE 1
Relative error and effectivity index of the estimator $\eta_{est,upp}$.

$\mathcal{E}(\boldsymbol{u}_0)/Q(\boldsymbol{u})$	$\eta_{\text{est,upp}}/\mathcal{E}(\boldsymbol{u}_0)$
0.73	1.22

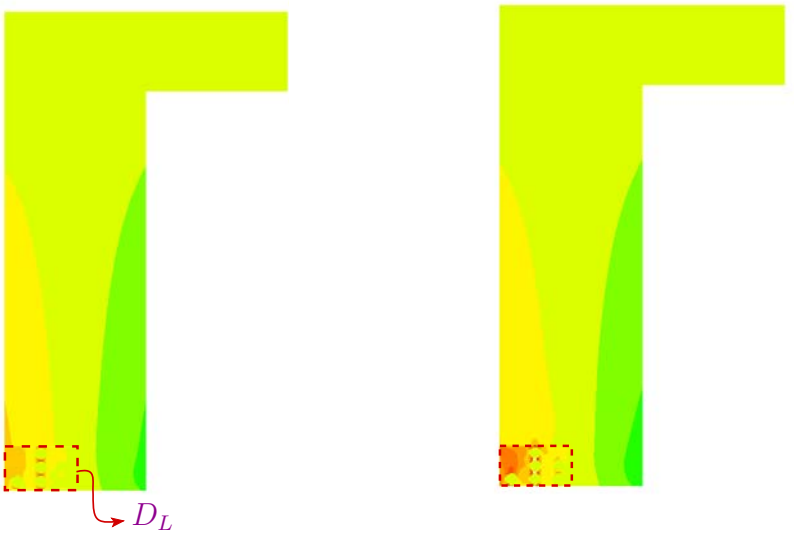
TABLE 2
Relative error and estimator effectivity for the locally enhanced problems.

	Local Goals		Global Goals	
Step no.	$\mathcal{E}(\tilde{\boldsymbol{u}})/Q(\boldsymbol{u})$	$\tilde{\eta}_{\text{est}}/\mathcal{E}(\tilde{\boldsymbol{u}})$	$\mathcal{E}(\tilde{\boldsymbol{u}})/Q(\boldsymbol{u})$	$\tilde{\eta}_{\text{est}}/\mathcal{E}(\tilde{\boldsymbol{u}})$
1	0.61	1.07	0.20	1.31
2	0.61	1.10	0.15	1.49
3	0.56	1.08	0.13	1.40

of the resulting strain field $\varepsilon_{yy}(\boldsymbol{x},\omega)$ is shown in Figure 4, where the area of interest around D_Q has been magnified. In Table 1, the relative error in the quantity of interest and the effectivity indices of the estimate $\eta_{\text{est,upp}}$ is given. Even though the relative error is large, the estimator $\eta_{\text{est,upp}}$ provides a fairly accurate assessment of the error with an effectivity index of 1.22.

To reduce the error, we perform two separate series of local enhancements. In one series, we employ the Local Goals algorithm, while the Global Goals is used in a second series. In Table 2, the relative errors and the effectivity indices of the error estimator $\tilde{\eta}_{\text{est}}$ are listed for three steps of both approaches. Also, in Figures 5 and 6, the statistical averages of the enhanced strain fields $\tilde{\varepsilon}_{yy}$ that are obtained after the first and third step, respectively, are shown. In these figures, the boundary of the domain of influence ∂D_L is highlighted red.

As expected (see Remark 1), the results in Table 2 reveal that, although the Global Goals algorithm requires solving a larger problem, the rate of decrease in the error in the target quantity is more pronounced with this approach. In both approaches, the



(a) Local Goals (b) Global Goals

FIG. 5. *Mean values of the strain field $\tilde{\varepsilon}_{yy}$ after the first step of local enhancement using the Local and Global Goals algorithms.*

estimator $\tilde{\eta}_{\text{est}}$ exhibits good accuracy with effectivity indices close to unity. For the Global Goals version, the effectivity is slightly worse, but this is due to the fact that the error has been reduced to much smaller values. In these applications, we have observed that the estimates become less accurate as the actual error decreases.

6.2. Nanoindentation application. In this section, we consider the problem of determining static equilibrium configurations of a regular lattice of N atoms. The base problem is derived by minimization of the potential energy of the system involving all atoms in the lattice. However, for many applications, the base problem is intractable due to the very large number of atoms N. We construct here surrogate problems using the popular QCM [44, 45, 46] which allows for the reduction of the number of active atoms as needed.

6.2.1. The base and surrogate problems. Let $\mathcal{L}$ be a regular lattice of N atoms in $\mathbb{R}^d$, $d = 2$ or 3. The positions of the atoms in the reference configuration are given by the vectors $\hat{\boldsymbol{x}}_i \in \mathbb{R}^d$, $i = 1, \ldots, N$, and in the equilibrium configuration by $\boldsymbol{x}_i = \hat{\boldsymbol{x}}_i + \boldsymbol{u}_i$, $i = 1, \ldots, N$, where $\boldsymbol{u}_i$ is the displacement of atom i. We assume that the lattice in the reference configuration covers the region $\bar{\Omega}$, where Ω is an open bounded set of $\mathbb{R}^d$ with boundary $\partial\Omega$. For a more detailed analysis, see [34]. We introduce the finite-dimensional vector space $V = (\mathbb{R}^d)^N$, and we use the notation $\boldsymbol{u} = (\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_N) \in V$ to refer to the displacements of the collection of N atoms.

(a) Local Goals (b) Global Goals

FIG. 6. *Mean values of the strain field $\tilde{\varepsilon}_{yy}$ after the third step of local enhancement using the Local and Global Goals algorithms.*

The total potential energy of the system is assumed to be the sums

$$E(\boldsymbol{u}) = -\sum_{i=1}^{N} \boldsymbol{f}_i \cdot \boldsymbol{u}_i + \sum_{k=1}^{N} E_k(\boldsymbol{u}), \tag{39}$$

where $\boldsymbol{f}_i$ is the external load applied to an interior atom i and $E_k(\boldsymbol{u})$ is the energy of atom k determined from interatomic potentials.

The goal of molecular statics is to find the equilibrium state $\boldsymbol{u} \in V$ that minimizes the total potential energy of the system, i.e., $\boldsymbol{u} = \operatorname{argmin}_{\boldsymbol{v}\in V} E(\boldsymbol{v})$. This minimization problem can be recast into the variational problem:

$$\textit{Find } \boldsymbol{u} \in V \textit{ such that} \quad B(\boldsymbol{u};\boldsymbol{v}) = F(\boldsymbol{v}) \quad \forall \boldsymbol{v} \in V, \tag{40}$$

where, for any $\boldsymbol{u} \in V$ and $\boldsymbol{v} \in V$,

$$\begin{aligned} B(\boldsymbol{u};\boldsymbol{v}) &= \sum_{i=1}^{N} \left[\sum_{k=1}^{N} \frac{\partial E_k}{\partial \boldsymbol{u}_i}(\boldsymbol{u}) \right] \cdot \boldsymbol{v}_i, \\ F(\boldsymbol{v}) &= \sum_{i=1}^{N} \boldsymbol{f}_i \cdot \boldsymbol{v}_i. \end{aligned} \tag{41}$$

Here $\partial/\partial \boldsymbol{u}_i$ is the gradient vector with respect to each component $u_{\alpha,i}$ of the displacement vector $\boldsymbol{u}_i$; i.e., $\partial/\partial \boldsymbol{u}_i = (\partial/\partial u_{1,i}, \ldots, \partial/\partial u_{d,i})$.

We briefly recall the main features of the quasicontinuum method (QCM). For more details, see [44, 40, 21]. The first step consists of choosing a set of R representative atoms, the so-called *repatoms*, and of approximating $\boldsymbol{u} \in V$ by a reduced

vector $\boldsymbol{u}_0 \in W = (\mathbb{R}^d)^R$, $R \ll N$, with $\boldsymbol{u}_0 = (\boldsymbol{u}_{0,1}, \boldsymbol{u}_{0,2}, \ldots, \boldsymbol{u}_{0,R}) \in W$. The displacements $\boldsymbol{u}_0$ represent the active degrees of freedom of the system and the repatoms are conveniently identified with the nodes of a finite element partition $\mathcal{P}_h$ of Ω. The displacements of the $(N-R)$ slave atoms are then interpolated from $\boldsymbol{u}_0$ by piecewise linear polynomials using the finite element mesh.

The second step of the QCM is devoted to the approximation of the total energy $E(\Pi \boldsymbol{u}_0)$ (recall section 3.1) by summing only over the repatoms such that

$$E(\Pi \boldsymbol{u}_0) \approx E_0(\boldsymbol{u}_0) = -\sum_{r=1}^{R} n_r \, \boldsymbol{f}_{0,r} \cdot \boldsymbol{u}_{0,r} + \sum_{r=1}^{R} n_r E_r(\boldsymbol{u}_0), \tag{42}$$

where n_r is an appropriate weight function associated with repatom r so as to account for all atoms in the lattice, i.e., $\sum_r n_r = N$, and $\boldsymbol{f}_{0,r}$ is the averaged external force acting on repatom r. For further details, see [21].

The minimization of the energy $E_0(\boldsymbol{u}_0)$ yields the surrogate problem

$$\textit{Find } \boldsymbol{u}_0 \in W \textit{ such that} \quad B_0(\boldsymbol{u}_0; \boldsymbol{v}) = F_0(\boldsymbol{v}) \quad \forall \boldsymbol{v} \in W, \tag{43}$$

where the semilinear form $B_0(\cdot;\cdot)$ and linear form $F_0(\cdot)$ are defined as

$$\begin{aligned} B_0(\boldsymbol{u}; \boldsymbol{v}) &= \sum_{i=1}^{R} \left[\sum_{r=1}^{R} n_r \frac{\partial E_r}{\partial \boldsymbol{u}_i}(\boldsymbol{u}) \right] \cdot \boldsymbol{v}_i, \\ F_0(\boldsymbol{v}) &= \sum_{i=1}^{R} n_i \, \boldsymbol{f}_{0,i} \cdot \boldsymbol{v}_i. \end{aligned} \tag{44}$$

6.2.2. Error estimation and adaptivity. In the nanoindentation problem that will be presented in the next section, the quantity of interest is the force that the crystal exerts on the indenter. Let the atoms, in contact with the lower surface of the indenter, be numbered from 1 to M. The quantity of interest is

$$Q(\boldsymbol{u}) = -\sum_{i=1}^{M} f_{y,i} = -\sum_{i=1}^{M} \frac{\partial E_i}{\partial u_{y,i}}(\boldsymbol{u}), \tag{45}$$

where $f_{y,i}$ and $u_{y,i}$ are the force and the displacement in the y-direction with respect to atom i. We assume that the meshes are such that the M atoms under the indenter are representative atoms, and all the forces are computed by the nonlocal approach (see [21]). The objective is then to estimate the error quantity

$$\mathcal{E} = Q(\boldsymbol{u}) - Q(\Pi \boldsymbol{u}_0). \tag{46}$$

The dual problem associated with the base problem (40) reads

$$\textit{Find } \boldsymbol{p} \in V \textit{ such that} \quad B'(\boldsymbol{u}; \boldsymbol{v}, \boldsymbol{p}) = Q'(\boldsymbol{u}; \boldsymbol{v}) \quad \forall \boldsymbol{v} \in V, \tag{47}$$

where the derivatives are given in the molecular statics case by

$$\begin{aligned} B'(\boldsymbol{u}; \boldsymbol{v}, \boldsymbol{p}) &= \sum_{j=1}^{N} \sum_{i=1}^{N} \boldsymbol{v}_j \cdot \left[\sum_{k=1}^{N} \frac{\partial^2 E_k}{\partial \boldsymbol{u}_j \partial \boldsymbol{u}_i}(\boldsymbol{u}) \right] \cdot \boldsymbol{p}_i, \\ Q'(\boldsymbol{u}; \boldsymbol{v}) &= -\sum_{j=1}^{N} \boldsymbol{v}_j \cdot \left[\sum_{i=1}^{M} \frac{\partial^2 E_i}{\partial \boldsymbol{u}_j \partial u_{y,i}}(\boldsymbol{u}) \right]. \end{aligned} \tag{48}$$

The surrogate dual problem is then

$$\text{Find } \boldsymbol{p}_0 \in W \text{ such that } \quad B_0'(\boldsymbol{u}_0; \boldsymbol{v}, \boldsymbol{p}_0) = Q_0'(\boldsymbol{u}_0; \boldsymbol{v}) \quad \forall \boldsymbol{v} \in W, \tag{49}$$

where by Q_0 we mean $Q_0(\boldsymbol{v}) = Q(\Pi \boldsymbol{v})$. The errors $\boldsymbol{e}_0 \in V$ and $\boldsymbol{\varepsilon}_0 \in V$ in $\Pi \boldsymbol{u}_0$ and $\Pi \boldsymbol{p}_0$, respectively, are defined as $\boldsymbol{e}_0 = \boldsymbol{u} - \Pi \boldsymbol{u}_0$ and $\boldsymbol{\varepsilon}_0 = \boldsymbol{p} - \Pi \boldsymbol{p}_0$. Then, the error in $Q(\boldsymbol{u})$ produced by $\Pi \boldsymbol{u}_0$ is given by, using Theorem 1,

$$\mathcal{E} = Q(\boldsymbol{u}) - Q(\Pi \boldsymbol{u}_0) = \mathcal{R}(\Pi \boldsymbol{u}_0; \boldsymbol{p}) + \Delta_1(\Pi \boldsymbol{u}_0, \Pi \boldsymbol{p}_0, \boldsymbol{e}_0, \boldsymbol{\varepsilon}_0), \tag{50}$$

where $\mathcal{R}(\Pi \boldsymbol{u}_0; \boldsymbol{v})$ is the residual functional, with respect to the base model problem

$$\mathcal{R}(\Pi \boldsymbol{u}_0; \boldsymbol{v}) = F(\boldsymbol{v}) - B(\Pi \boldsymbol{u}_0; \boldsymbol{v}), \quad \boldsymbol{v} \in V. \tag{51}$$

For details on the error estimator, see [34].

The adaptivity algorithm then proceeds as follows:

1. Initialize the load step s = 0. Input user-tolerance δ_{tol}.
2. s = s + 1.
3. Solve the primal and dual problems.
4. Compute the error estimates.
5. Check: $|\eta| > \delta_{\text{tol}} |Q(\boldsymbol{u}_0)|$. If false: go to step 2. If true: mark those elements with $|\eta_K(\Pi \boldsymbol{u}_0, \tilde{\boldsymbol{p}})| > \alpha \max_K |\eta_K(\Pi \boldsymbol{u}_0, \tilde{\boldsymbol{p}})|$, where α is a user-supplied number between 0 and 1.
6. Refine flagged elements and go to step 3.

6.2.3. A numerical example. The performance of the error estimator and adaptive strategy is demonstrated for a nanoindentation problem proposed by Tadmor et al. [33, 40, 45]. The simulation is actually provided as a model example accompanying the open source software package [22]. In this example, a thin film of aluminum crystal is indented by a rigid rectangular indenter, 9.31 Å wide and infinite in the out-of-plane direction, as depicted in Figure 7. The dimensions for the block of crystal are 2000×1000 (in Angströms) in the [111] and [$\bar{1}$10] directions of the crystal. The crystal rests on a rigid support so that homogeneous boundary conditions $\boldsymbol{u}_i = \boldsymbol{0}$ are prescribed for those atoms i located at $y = 0$. The indenter is moved downward in a succession of 30 load-step increments $\delta l = 0.2$. The boundary conditions for the atoms i just below the indenter are given by $\boldsymbol{u}_i = (0, -m\,\delta l, 0)$, $m = 1, \ldots, 30$. Quasistatic steps are then considered to solve for the displacements.

The site energies $E_k(\boldsymbol{u})$ of the aluminum crystal are computed here using the embedded atom method (EAM) (see, e.g., [9, 14]). Briefly, the semiempirical potential energy for atom k is given by the sum of an electron-density dependent embedding energy and pairwise interatomic potentials. The QCM also employs a cutoff function to approximate the interatomic potentials as the potential function decays rapidly with respect to the interatomic distance.

The interatomic distances in the undeformed configuration of the crystal are 2 Å in the x- and y-directions. One $(\bar{1}, \bar{1}, 0)$ layer of the film contains about 500,000 atoms. Rather than solving for the solution $\boldsymbol{u}$ of the full base problem (40), we have opted to consider as our reference solution an "overkill" solution of the surrogate problem (43). This solution is hereafter referred to as the base model solution as it involves a sufficiently high number of degrees of freedom so that it is a very accurate approximation of $\boldsymbol{u}$.

The base model solution is compared to the solution $\boldsymbol{u}_0$ obtained by the QCM on a much coarser discretization. In the quasicontinuum (QC) solution, the refinement

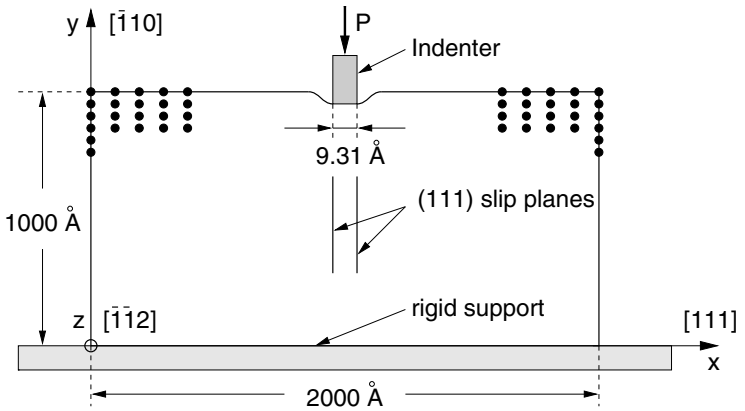


FIG. 7. *Nanoindentation of an aluminum crystal (the schematic representation is inspired by Figure* 1 *in* [45]*).*

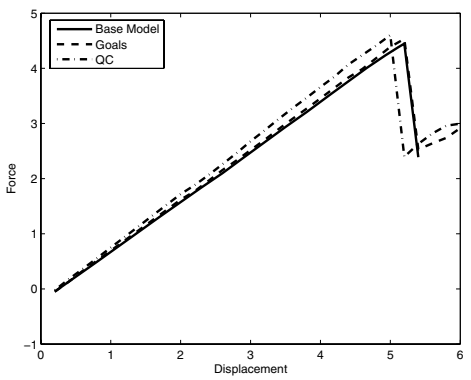


FIG. 8. *Force-displacement curve comparing the evolution of the base model, QC, and Goals solutions.*

parameter was set according to the authors' recommendations [22] to .075 while the Goals solution was set to a 5 percent error tolerance in the quantity of interest. Next, the force under the indenter is shown in Figure 8 for the base model solution, the QC solution, and the Goals algorithm. This force represents a quantity of physical interest as it can be experimentally measured and clearly indicates the nucleation of the dislocation. The QCM produces a stiffer material so that the critical force for dislocation nucleation occurs one load-step earlier than the base model solution. On the other hand, the Goals solution produces a remarkably accurate representation of the force-displacement curve when compared to the base model. Shown in Figure 9 are the relative errors and effectivity indices for each load-step, demonstrating the effectiveness of the error estimator used. Note that the effectivity index at loadstep 2 is far from unity due to the fact that the error is very near zero. As can be seen, the error in the solution is controlled to within the preset tolerance of 5 percent with the exception of the third and final load-steps. Although the error estimator overestimates the exact error for most of the load-steps in this example, this behavior may not be true for other cases. It is not proved that this is a guaranteed upper bound.

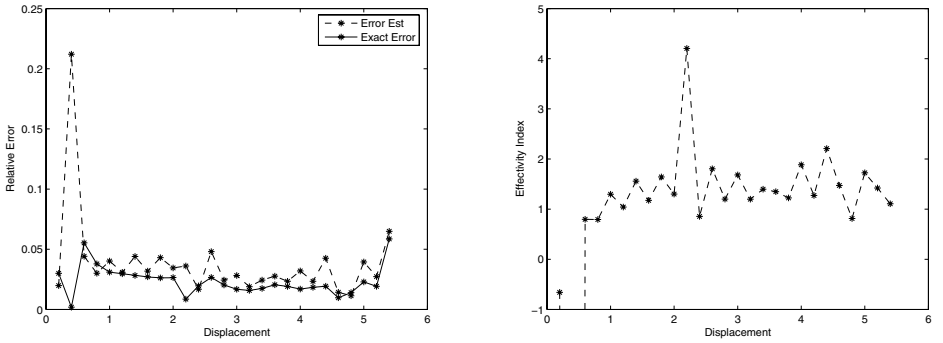


FIG. 9. *Exact and estimated relative errors for the Goals solution (left). Effectivity indices (right).*

6.3. Molecular dynamics application. We present in this section an application to modeling the dynamics of atoms whose motion is determined by molecular dynamics (MD) principles. We consider here surrogate models generated by the bridging-scale method (BSM) of Wagner and Liu [51] and the pseudospectral multiscale method (PMM) of Tang, Hou, and Liu [48]. We will show only preliminary results, but future work will include the development of automatic model adaptivity algorithms and the investigation of other surrogate models such as that put forth by Xiao and Belytschko [52].

6.3.1. The base model. Let $\Omega \subset \mathbb{Z}^d$ be an open bounded set, $d = 1$, 2, or 3. Let $\mathcal{L}$ denote the lattice of n atoms covering $\overline{\Omega}$. The initial positions of the atoms are given by the vectors $\boldsymbol{x}_i \in \mathbb{R}^d$ and the respective displacements by $\boldsymbol{u}_i \in \mathbb{R}^d$, $i = 1, \ldots, n$. The notation $\boldsymbol{u} = (\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_n)$ collectively represents the displacement of the n atoms. We denote by $\boldsymbol{M}$ the mass matrix of the system $M_{ij} = m_{(i)}\delta_{ij}$, where m_i is the mass of atom i. Additionally, the interatomic potential energy (assumed to be given) is represented by $E(\boldsymbol{u})$ and the interatomic forces are computed as $\boldsymbol{f}(\boldsymbol{u}) = -\partial_{\boldsymbol{u}} E(\boldsymbol{u})$.

The MD problem is then characterized by the following system of ordinary differential equations for the displacements $\boldsymbol{u} = \boldsymbol{u}(t)$:

$$\boldsymbol{M}\ddot{\boldsymbol{u}} = \boldsymbol{f}(\boldsymbol{u}), \quad \boldsymbol{u}(0) = \boldsymbol{U}_0, \quad \dot{\boldsymbol{u}}(0) = \boldsymbol{V}_0 \tag{52}$$

subjected to appropriate boundary conditions. Note that external forces on the atoms have been neglected for simplicity. The superimposed dot implies differentiation with respect to time. Assuming $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{V} = C^2\left((0,T); \left(\mathbb{R}^d\right)^n\right)$, we may recast the above problem into the weak form (1) with

$$\begin{aligned} B(\boldsymbol{u};\boldsymbol{v}) &= \int_0^T \boldsymbol{v}^T \left(\boldsymbol{M}\ddot{\boldsymbol{u}} - \boldsymbol{f}(\boldsymbol{u})\right) dt + \boldsymbol{v}^T(0)\boldsymbol{M}\dot{\boldsymbol{u}}(0) - \dot{\boldsymbol{v}}^T(0)\boldsymbol{M}\boldsymbol{u}(0), \\ F(\boldsymbol{v}) &= \boldsymbol{v}^T(0)\boldsymbol{M}\boldsymbol{V}_0 - \dot{\boldsymbol{v}}^T(0)\boldsymbol{M}\boldsymbol{U}_0. \end{aligned} \tag{53}$$

Note that the initial conditions in the above formulation are weakly imposed. Similarly, it is not difficult to derive the first derivative of the semilinear form $B(\cdot;\cdot)$. Indeed, we have for $\boldsymbol{p}, \boldsymbol{v} \in \mathbb{V}$,

$$\begin{aligned} B'(\boldsymbol{u};\boldsymbol{v},\boldsymbol{p}) = \int_0^T \left(\boldsymbol{M}\ddot{\boldsymbol{p}} - (\boldsymbol{f}'(\boldsymbol{u}))^T\boldsymbol{p}\right)^T \boldsymbol{v}\, dt \\ + (\boldsymbol{M}\boldsymbol{p}(T))^T\dot{\boldsymbol{v}}(T) - (\boldsymbol{M}\dot{\boldsymbol{p}}(T))^T\boldsymbol{v}(T), \end{aligned} \tag{54}$$

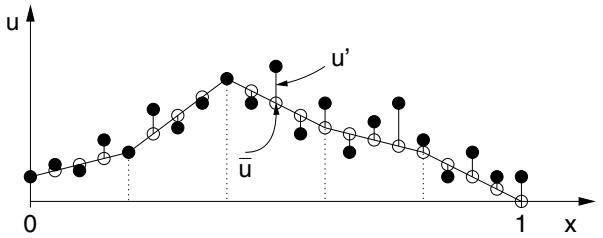


FIG. 10. *Illustration of the scale decomposition for BSM.*

where $\boldsymbol{f}'(\boldsymbol{u}) = \partial_{\boldsymbol{u}} \boldsymbol{f}(\boldsymbol{u})$ is the Hessian of $E(\boldsymbol{u})$ evaluated at $\boldsymbol{u}$.

6.3.2. The surrogate models: BSM and PMM. We consider as surrogate models those determined by the BSM [47, 51] and the PMM [48]. These methods are similar and their overall objectives are the following:

1. Construct a coarse-scale model from the MD model.
2. Solve the MD model only over a subdomain $\Omega_{MD} \subset \Omega$ and solve the coarse-scale model in the complementary subdomain $\Omega_C = \Omega \setminus \Omega_{MD}$. One difficulty is then to develop proper interface conditions to reduce/eliminate spurious wave reflection at the interface of Ω_{MD} and Ω_C.

The starting point of the BSM and the PMM is based on the classical decomposition of scales (see, e.g., [17]):

$$\boldsymbol{u} = \bar{\boldsymbol{u}} + \boldsymbol{u}', \quad \bar{\boldsymbol{u}} = \boldsymbol{P}\boldsymbol{u}, \quad \boldsymbol{u}' = \boldsymbol{Q}\boldsymbol{u} = (\boldsymbol{I} - \boldsymbol{P})\boldsymbol{u}, \tag{55}$$

where $\bar{\boldsymbol{u}}$ and $\boldsymbol{u}'$ define the coarse-scale displacements and the fine-scale fluctuations (see Figure 10), respectively, $\boldsymbol{P}$ is a general projection operator, and $\boldsymbol{I}$ is the identity matrix. One main difference between the BSM and the PMM is that $\bar{\boldsymbol{u}}$ is obtained in terms of piecewise linear functions in the case of the BSM and of a spectral representation in the case of the PMM.

Considering the subdomains Ω_{MD} and Ω_C, the solution $\boldsymbol{u}$ can be further decomposed into the following form:

$$\boldsymbol{u} = \begin{bmatrix} \boldsymbol{u}_{MD} \\ \boldsymbol{u}_C \end{bmatrix} = \begin{bmatrix} \bar{\boldsymbol{u}}_{MD} \\ \bar{\boldsymbol{u}}_C \end{bmatrix} + \begin{bmatrix} \boldsymbol{u}'_{MD} \\ \boldsymbol{u}'_C \end{bmatrix} \qquad \begin{matrix} \text{in } \Omega_{MD}, \\ \text{in } \Omega_C, \end{matrix} \tag{56}$$

where $\boldsymbol{u}_{MD}$ and $\boldsymbol{u}_C$ are the parts of the solution in Ω_{MD} and in Ω_C, respectively. In essence, the BSM and the PMM consist in approximating the large scales $\bar{\boldsymbol{u}}_{MD} + \bar{\boldsymbol{u}}_C$ by constructing a coarse-scale model over the whole domain Ω (see [51, 47, 48] for details) and by using $\bar{\boldsymbol{u}}_C$ to define the boundary conditions for the full MD model in Ω_{MD}. However, it is well known that such a coarse model may yield spurious wave reflection at the interface of the two domains. In order to alleviate this issue, the BSM and the PMM propose to calculate the fine-scale components in Ω_C corresponding to those atoms that are within the cutoff radius of atoms in Ω_{MD}. Therefore, the vector $\boldsymbol{u}'_C$ is approximated by a vector, say, $\boldsymbol{u}'_G$, that has zero entries, except at the position of these atoms "close" to the interface boundary. It follows that the surrogate models using the BSM or the PMM produce solutions in the form

$$\tilde{\boldsymbol{u}} = \begin{bmatrix} \boldsymbol{u}_{MD}, \\ \bar{\boldsymbol{u}}_C, \end{bmatrix} \qquad \begin{matrix} \text{in } \Omega_{MD}, \\ \text{in } \Omega_C. \end{matrix} \tag{57}$$

Once again, the readers are referred to [51, 47, 48] for the full description of the methods. Our main interest in the present work is to provide error estimates with respect to quantities of interest for the surrogate solutions $\tilde{\boldsymbol{u}}$.

6.3.3. Error estimates. The residual functional with respect to the MD problem can be written as

$$\mathcal{R}(\tilde{\boldsymbol{u}}, \boldsymbol{v}) = \int_0^T \boldsymbol{v}^T (\underbrace{\boldsymbol{M}\ddot{\tilde{\boldsymbol{u}}} - \boldsymbol{f}(\tilde{\boldsymbol{u}})}_{\boldsymbol{r}(\tilde{\boldsymbol{u}})}) dt = \int_0^T \sum_{i=1}^n \boldsymbol{v}_i \cdot \boldsymbol{r}_i(\tilde{\boldsymbol{u}}) dt. \tag{58}$$

The local residual $\boldsymbol{r}_i$ simply indicates how the second Newton law fails to be satisfied at atom i. Given a quantity of interest $Q(\boldsymbol{u})$, the modeling error in $Q(\tilde{u})$ can be approximated by

$$Q(\boldsymbol{u}) - Q(\tilde{\boldsymbol{u}}) \approx \mathcal{R}(\tilde{\boldsymbol{u}}, \boldsymbol{p}) = \int_0^T \sum_{i=1}^n \boldsymbol{p}_i \cdot \boldsymbol{r}_i(\tilde{u}) dt, \tag{59}$$

where $\boldsymbol{p}$ is the dual solution associated with $Q(\boldsymbol{u})$. The computational domain is divided here into a partition $\{\Theta\}_{j=0}^k$, as indicated in section 4. In the case of the BSM, the subdomains are conveniently constructed from the coarse lattice used in the solution of the coarse-scale model. We note that the error quantity can then be decomposed into local contributions either in time, η_t, or in space, η_j, as

$$\eta_t = \sum_{i=1}^n \boldsymbol{p}_i \cdot \boldsymbol{r}_i(\tilde{\boldsymbol{u}}), \qquad \eta_j = \int_0^T \sum_{i \in \Theta_j} \boldsymbol{p}_i \cdot \boldsymbol{r}_i(\tilde{\boldsymbol{u}}) dt, \tag{60}$$

and that the error can be rewritten in terms of these local contributions as

$$Q(\boldsymbol{u}) - Q(\tilde{\boldsymbol{u}}) \approx \mathcal{R}(\tilde{\boldsymbol{u}}, \boldsymbol{p}) = \int_0^T \eta_t dt = \sum_{j=0}^k \eta_j. \tag{61}$$

We will show in the following example the distribution of the contributions η_i and the evolution of the contributions η_t.

6.3.4. Numerical example. The performance of the error estimator and of the Local Goals algorithm is demonstrated here for the one-dimensional model problem studied in [47]. In this problem, $\Omega = [-2, 2]$, $\Omega_{MD} = [-0.35, 0.35]$, with atomic spacing $h_a = 0.005$. The atoms in domains Ω_{MD} and Ω_C are represented in black and red, respectively, in Figure 11 and subsequent figures. The initial conditions are given by

$$V_0(x_i) = 0, \qquad x_i \in \Omega, \tag{62}$$

and

$$U_0(x_i) = \begin{cases} 0.005\,(1 + 0.1\cos(80\pi x_i))\,\dfrac{e^{-100x_i^2} - e^{-6.25}}{1 - e^{-6.25}}, & |x_i| \le 0.25, \\ 0 & \text{otherwise.} \end{cases} \tag{63}$$

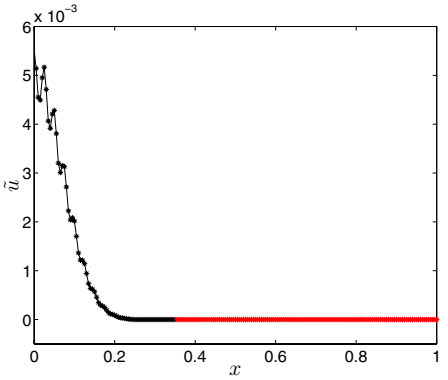


FIG. 11. *Initial displacement $U_0(x_i)$. The points in black represent atoms in Ω_{MD} while points in red denote atoms in Ω_C.*

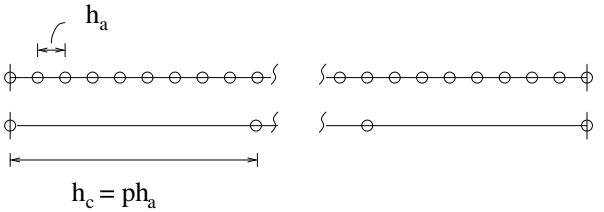


FIG. 12. *Schematic illustrating the construction of the coarse lattice on which the coarse-scale displacements $\bar{\boldsymbol{u}}$ are computed (BSM).*

Furthermore, $m_i = 1$, $i = 1, \ldots, n$, and the interatomic forces are given by linear springs:

$$\boldsymbol{f}(\boldsymbol{u}) = \boldsymbol{K}\boldsymbol{u}, \qquad \boldsymbol{K} = \begin{bmatrix} -2 & 1 & & \\ 1 & -2 & \ddots & \\ & \ddots & \ddots & 1 \\ & & 1 & -2 \end{bmatrix}. \tag{64}$$

In the case of the BSM, the coarse lattice is constructed by placing a coarse grid point at every p atom in Ω so that the coarse grid spacing $h_c = ph_a$; see Figure 12. The time integration scheme is identical to that used by Wagner and Liu [51]. In this example, we suppose that we are interested in the locally averaged displacement

$$Q(\boldsymbol{u}) = \frac{1}{M} \sum_{i \in \mathcal{M}} u_i, \qquad \mathcal{M} = \{i : |x_i| \leq 0.0025\}, \tag{65}$$

for which the strong form of the dual problem is

$$\boldsymbol{M}\ddot{\boldsymbol{p}} - \left(\boldsymbol{f}'(\boldsymbol{u})\right)^T \boldsymbol{p} = \boldsymbol{0}, \qquad -\boldsymbol{M}\dot{\boldsymbol{p}}(T) = \boldsymbol{q}, \qquad \boldsymbol{p}(T) = \boldsymbol{0}, \tag{66}$$

where M is the number of atoms considered in set $\mathcal{M}$ and $\boldsymbol{q}$ is the vector defined such that $Q(\boldsymbol{u}) = \boldsymbol{q}^T\boldsymbol{u}$. The dual problem is solved by integrating (66) backward in time using a numerical scheme similar to that used in the forward time integration of the primal problem.

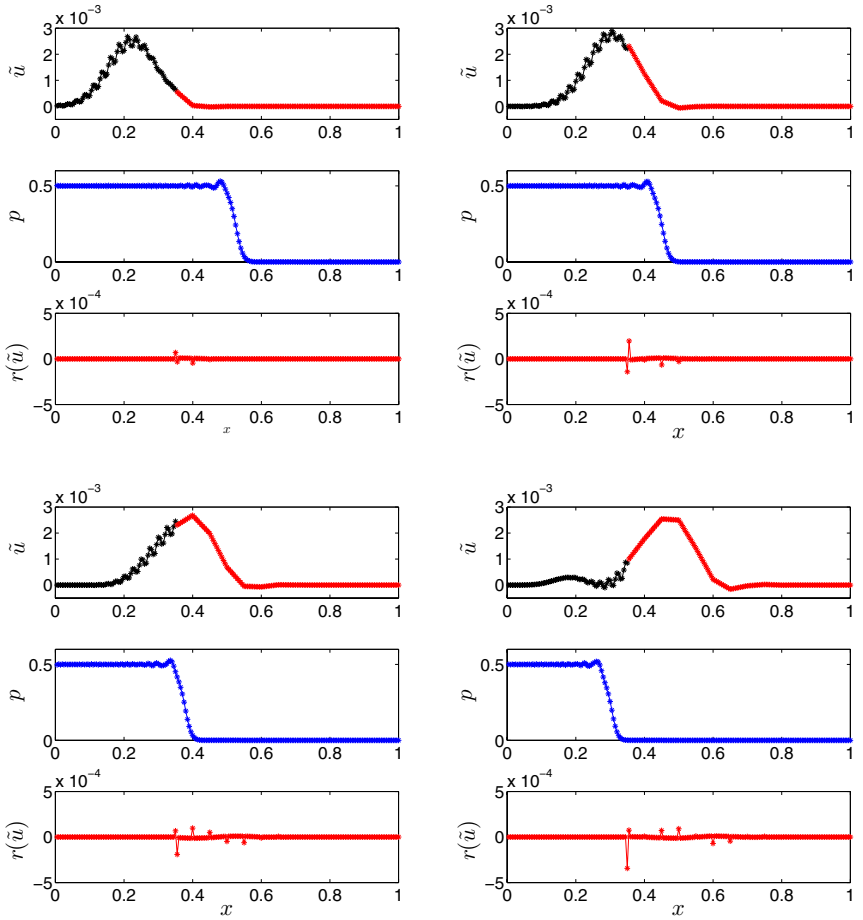


FIG. 13. *Time snapshots of the primal, dual, and force residual at* $t = 45, 60, 75, 90$ *(BSM).*

The first set of experiments deals with the BSM. The surrogate solution $\tilde{\boldsymbol{u}}$, the dual solution $\boldsymbol{p}$, and the local residuals $\boldsymbol{r}(\tilde{\boldsymbol{u}})$ are shown in Figure 13 for four specific times. Observe how the dual solution propagates in time in the opposite direction to the primal solution. The dual solution indicates how the local residuals $\boldsymbol{r}(\tilde{\boldsymbol{u}})$ influence the error in the quantity of interest. In particular, it can be seen that, as the front moves left to right, the dual solution reduces to zero. This simply means that the sources of error introduced in those spatial regions in which the dual solution is zero do not have enough time to propagate and pollute the quantity of interest. In the meantime, the local residuals become significant when the fine-scale noise starts to move across the interface boundary. The combination of both effects implies that the major contributions to the error in the quantity of interest are generated during the time interval $0.35 \leq t \leq 0.90$ and in the subregion $0.35 \leq x \leq 0.5$ as shown in Figure 14.

Since the sources of error, according to Figure 14, are essentially created in the space intervals $[0.35, 0.5]$ and $[-0.5, -0.35]$ (by symmetry), we choose to enlarge Ω_{MD} to $[-0.5, 0.5]$ in a one adaptive step of the Goals algorithm. Figure 15 displays the new

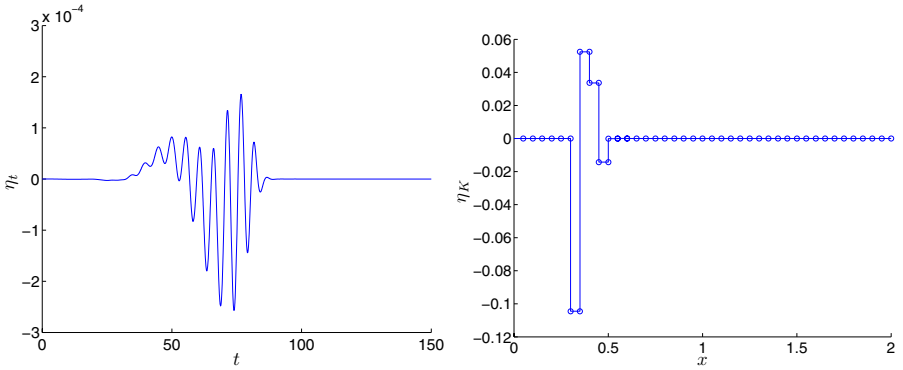


FIG. 14. *The temporal contribution to the error (left) and the spatial contribution (right) (BSM).*

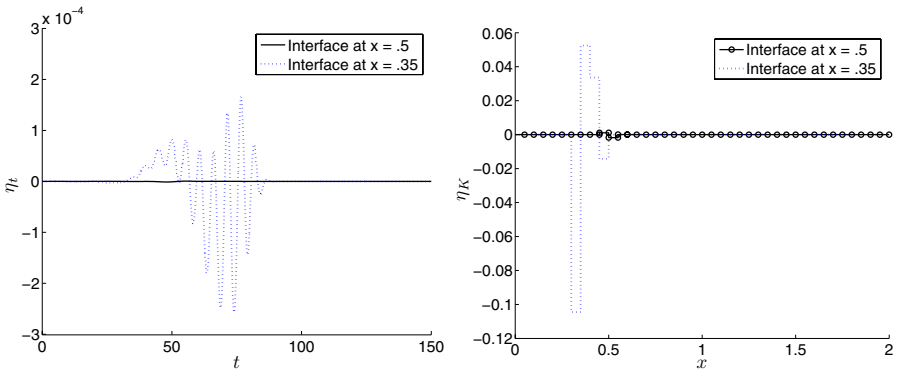


FIG. 15. *The temporal contribution to the error (left) and the spatial contribution (right) following model adaptivity (BSM).*

error contributions along the preadapted solution errors for comparison. As clearly shown, the modeling error has been effectively eliminated. Note that the dual solution $\boldsymbol{p}$ in the present numerical experiment was computed using the fine-scale model. A major task will be to construct adequate and reliable surrogate models for the dual problem.

We repeat the same experiments in the case of the PMM. Here we show only the results of the adaptive algorithm by considering $\Omega_{MD} = [-0.5, 0.5]$, $[-0.4, 0.4]$, and $[-0.45, 0.45]$. Figure 16 shows how contribution to the errors is reduced by successive refinement of the model problem.

7. Concluding comments. That mathematical models of physical events are almost always imperfect abstractions of nature is a truism universally understood by all students of science and engineering. The possibility of quantifying in some way the level of imperfection is therefore an intriguing proposition. Our approach is to measure this imperfection as estimates of error in specific quantities that we single out as important features of the behavior of the system under consideration. But even then we do not escape the necessity of using another imperfect model, the base model of the event. Nevertheless, we believe that the relatively straightforward machinery we describe for comparing models by estimating modeling error can be very valuable, particularly when two or more models of events that occur at different scales are

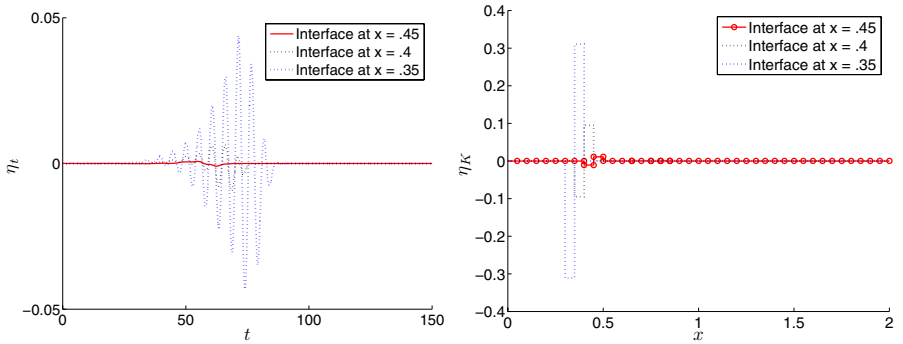


FIG. 16. *The temporal contribution to the error (left) and the spatial contribution (right) following model adaptivity (PMM).*

considered. If this process can be tied to physical measurements as well, it may be possible to develop powerful techniques for simulating events at multiple scales with a level of reliability beyond that thought possible in recent times.

There are many open issues that remain to be resolved. Given a list of quantities of interest for a base model at the finest scale, a systematic approach for developing the optimal ensemble-averaged or homogenized characterizations of these quantities at coarser scales is needed. We have offered some examples in this work, but further work on how these averaging techniques affect accuracy of estimates and convergence rates of the adaptive process is needed.

Adaptive modeling techniques are in early stages of development. The Global and Local Goals algorithms presented here represent only two of many possible approaches. Other techniques could be inspired by many of the multigrid methods in use in linear and nonlinear solvers. The role of various hand-shake methods for interfacing models of different scales as a basis for adaptive modeling is also a topic worthy of further study.

There is also the issue of modifying the definition of a quantity of interest as one progresses to coarser-scale models. In a two-scale situation, the coarse-scale model is a surrogate: an artifact of a computational process. In more general situations, it can be conceived that coarser-scale models may possess features of interest themselves, not always tied to fine-scale quantities of interest. The procedures we develop here should be applicable to such situations with straightforward modifications. Of overriding importance is the goal of estimating the relative error in quantities delivered by two or more models.

REFERENCES

[1] M. AINSWORTH AND J. T. ODEN, *A Posteriori Error Estimation in Finite Element Analysis*, John Wiley & Sons, New York, 2000.

[2] ALTAIR ENGINEERING, INC., *ProPHLEX User Manual Version* 3.0, Austin, TX, 2000.

[3] W. BANGERTH AND R. RANNACHER, *Adaptive Finite Element Methods for Differential Equations*, Birkhäuser Verlag, Basel, 2003.

[4] R. BECKER AND R. RANNACHER, *An optimal control approach to a posteriori error estimation in finite element methods*, Acta Numer., 10 (2001), pp. 1–102.

[5] T. BELYTSCHKO AND S. P. XIAO, *Coupling methods for continuum model with molecular model*, International Journal for Multiscale Computational Engineering, 1 (2003), pp. 115–126.

[6] X. Blanc, C. L. Bris, and P. L. Lions, *From molecular models to continuum mechanics*, Arch. Ration. Mech. Anal., 164 (2002), pp. 341–381.
[7] J. Q. Broughton, F. F. Abraham, N. Bernstein, and E. Kaxiras, *Concurrent coupling of length scales: Methodology and application*, Phys. Rev. B, 60 (1999), pp. 2391–2403.
[8] W. A. Curtin and R. E. Miller, *Atomistic/continuum coupling in computational materials science*, Modelling Simul. Mater. Sci. Eng., 11 (2003), pp. R33–R68.
[9] M. S. Daw and M. I. Baskes, *Embedded-atom method: Derivation and applications to impurities, surfaces, and other defects in metals*, Phys. Rev. B, 29 (1984), pp. 6443–6453.
[10] W. E and B. Engquist, *The heterogeneous multiscale methods*, Comm. Math. Sci., 1 (2003), pp. 87–132.
[11] W. E, B. Engquist, X. Li, W. Ren, and E. Vanden-Eijnden, *The Heterogeneous Multiscale Method: A Review*, preprint, Princeton University, Princeton, NJ, 2004, available at http://www.math.princeton.edu/multiscale/.
[12] W. E, X. Li, and E. Vanden-Eijnden, *Some Recent Progress in Multiscale Modeling*, preprint, Princeton University, Princeton, NJ, 2004, available at http://www.math.princeton.edu/multiscale/.
[13] W. E, P. B. Ming, and P. W. Zhang, *Analysis of the heterogeneous multiscale method for elliptic homogenization problems*, J. Amer. Math. Soc., 18 (2005), pp. 121–156.
[14] S. M. Foiles, M. I. Baskes, and M. S. Daw, *Embedded-atom-method functions for fcc metals Cu, Ag, Au, Ni, Pd, Pt, and their alloys*, Phys. Rev. B, 33 (1986), pp. 7983–7991.
[15] G. Friesecke and R. D. James, *A scheme for passage from atomic to continuum theory for thin films, nanotubes, and nanorods*, J. Mech. Phys. Solids, 48 (2000), pp. 1519–1540.
[16] Z. Hashin, *Analysis of composite materials, a survey*, J. Appl. Mech., 50 (1983), pp. 481–505.
[17] T. J. R. Hughes, G. R. Feijóo, L. Mazzei, and J.-B. Quincy, *The variational multiscale method—a paradigm for computational mechanics*, Comput. Methods Appl. Mech. Engrg., 166 (1998), pp. 3–24.
[18] S. Kohlhoff, P. Gumbsch, and H. F. Fischmeister, *Crack propagation in b.c.c. crystals studied with a combined finite-element and atomistic model*, Phil. Mag. A, 64 (1991), pp. 851–878.
[19] W. K. Liu, E. G. Karpov, S. Zhang, and H. S. Park, *An introduction to computational nanomechanics and materials*, Comput. Methods Appl. Mech. Engrg., 193 (2004), pp. 1529–1578.
[20] R. Miller, M. Ortiz, R. Phillips, V. Shenoy, and E. B. Tadmor, *Quasicontinuum models of fracture and plasticity*, Eng. Fracture Mech., 61 (1998), pp. 427–444.
[21] R. E. Miller and E. B. Tadmor, *The quasicontinuum method: Overview, applications, and current directions*, J. Computer-Aided Design, 9 (2002), pp. 203–239.
[22] R. E. Miller and E. B. Tadmor, *QC Tutorial Guide Version* 1.1, 2004, available at http://www.qcmethod.com.
[23] R. E. Miller, E. B. Tadmor, R. Phillips, and M. Ortiz, *Quasicontinuum simulation of fracture at the atomic scale*, Modeling Simul. Mater. Sci. Eng., 6 (1998), pp. 607–638.
[24] J. T. Oden and S. Prudhomme, *Goal-oriented error estimation and adaptivity for the finite element method*, Comput. Math. Appl., 41 (2001), pp. 735–756.
[25] J. T. Oden and S. Prudhomme, *Estimation of modeling error in computational mechanics*, J. Comput. Phys., 182 (2002), pp. 496–515.
[26] J. T. Oden, S. Prudhomme, and P. Bauman, *On the extension of goal-oriented error estimation and hierarchical modeling to discrete lattice models*, Comput. Methods Appl. Mech. Engrg., 194 (2005), pp. 3668–3688.
[27] J. T. Oden, S. Prudhomme, T. Westermann, J. Bass, and M. E. Botkin, *Error estimation of eigenfrequencies for elasticity and shell problems*, Math. Models Methods Appl. Sci., 13 (2003), pp. 323–344.
[28] J. T. Oden and K. Vemaganti, *Estimation of local modeling error and goal-oriented modeling of heterogeneous materials; Part* I. *Error estimates and adaptive algorithms*, J. Comput. Phys., 164 (2000), pp. 22–47.
[29] J. T. Oden, K. Vemaganti, and N. Moës, *Hierarchical modeling of heterogeneous solids*, Comput. Methods Appl. Mech. Engrg., 172 (1999), pp. 3–25.
[30] J. T. Oden and T. I. Zohdi, *Analysis and adaptive modeling of highly heterogeneous elastic structures*, Comput. Methods Appl. Mech. Engrg., 148 (1997), pp. 367–391.
[31] H. S. Park, E. G. Karpov, P. A. Klein, and W. K. Liu, *Three-dimensional bridging scale analysis of dynamic fracture*, J. Comput. Phys., 207 (2005), pp. 588–609.
[32] H. S. Park, E. G. Karpov, W. K. Liu, and P. A. Klein, *The bridging scale for two dimensional atomistic/continuum coupling*, Philosophical Magazine, 85 (2005), pp. 79–113.

[33] R. Phillips, D. Rodney, V. Shenoy, E. B. Tadmor, and M. Ortiz, *Hierarchical models of plasticity: Dislocation nucleation and interaction*, Modeling Simul. Mater. Sci. Eng., 7 (1999), pp. 769–780.

[34] S. Prudhomme, P. Bauman, and J. T. Oden, *Error Control for Molecular Statics Problems*, ICES Report 05-37, The University of Texas at Austin, Austin, TX, 2005.

[35] D. Qian, G. J. Wagner, and W. K. Liu, *A multiscale projection method for the analysis of carbon nanotubes*, Comput. Methods Appl. Mech. Engrg., 193 (2004), pp. 1603–1632.

[36] A. Romkes and J. T. Oden, *Adaptive modeling of wave propagation in heterogeneous elastic solids*, Comput. Methods Appl. Mech. Engrg., 193 (2004), pp. 539–559.

[37] A. Romkes, J. T. Oden, and K. Vemaganti, *Multi-scale goal-oriented adaptive modeling of random heterogeneous materials*, J. Mech. Mater., 38 (2006), pp. 859–872.

[38] A. Romkes, K. Vemaganti, and J. T. Oden, *The Extension of the GOALS Algorithm to the Analysis of Elastostatics Problems of Random Heterogeneous Materials*, ICES Report 04-45, The University of Texas at Austin, Austin, TX, 2004.

[39] V. B. Shenoy, R. Miller, E. B. Tadmor, R. Phillips, and M. Ortiz, *Quasicontinuum models of interfacial structure and deformation*, Phys. Rev. Lett., 80 (1998), pp. 742–745.

[40] V. B. Shenoy, R. Miller, E. B. Tadmor, D. Rodney, R. Phillips, and M. Ortiz, *An adaptive finite element approach to atomic-scale mechanics—the quasicontinuum method*, J. Mech. Phys. Solids, 47 (1999), pp. 611–642.

[41] L. E. Shilkrot, R. E. Miller, and W. A. Curtin, *Coupled atomistic and discrete dislocation plasticity*, Phys. Rev. Lett., 89 (2002), 025501.

[42] L. E. Shilkrot, R. E. Miller, and W. A. Curtin, *Multiscale plasticity modeling: Coupled atomistics and discrete dislocation mechanics*, J. Mech. Phys. Solids, 52 (2004), pp. 755–787.

[43] G. S. Smith, E. B. Tadmor, N. Bernstein, and E. Kaxiras, *Multiscale simulations of silicon nanoindentation*, Acta Mater., 49 (2001), pp. 4089–4101.

[44] E. B. Tadmor, *The Quasicontinuum Method*, Ph.D. thesis, Brown University, Providence, RI, 1996.

[45] E. B. Tadmor, R. Miller, R. Phillips, and M. Ortiz, *Nanoindentation and incipient plasticity*, J. Mater. Res., 14 (1999), pp. 2233–2250.

[46] E. B. Tadmor, M. Ortiz, and R. Phillips, *Quasicontinuum analysis of defects in solids*, Phil. Mag. A, 73 (1996), pp. 1529–1563.

[47] S. Tang, T. Y. Hou, and W. K. L. Liu, *A mathematical framework of the bridging scale method*, Internat. J. Numer. Methods Engrg., 65 (2005), pp. 1688–1713.

[48] S. Tang, T. Y. Hou, and W. K. L. Liu, *Pseudo-spectral multiscale method: Interfacial conditions and coarse grid equations*, J. Comput. Phys., 213 (2006), pp. 57–85.

[49] E. van der Giessen and A. Needleman, *Discrete dislocation plasticity: A simple planar model*, Modeling Simul. Mater. Sci. Eng., 3 (1995), pp. 689–735.

[50] K. Vemaganti and J. T. Oden, *Estimation of local modeling error and goal-oriented modeling of heterogeneous materials; part* II: *A computational environment for adaptive modeling of heterogeneous elastic solids*, Comput. Methods Appl. Mech. Engrg., 190 (2001), pp. 6089–6124.

[51] G. J. Wagner and W. K. Liu, *Coupling of atomistic and continuum simulations using a bridging scale decomposition*, J. Comput. Phys., 190 (2003), pp. 249–274.

[52] S. P. Xiao and T. Belytschko, *A bridging domain method for coupling continua with molecular dynamics*, Comput. Methods Appl. Mech. Engrg., 193 (2004), pp. 1645–1669.

[53] T. I. Zohdi, J. T. Oden, and G. J. Rodin, *Hierarchical modeling of heterogeneous bodies*, Comput. Methods Appl. Mech. Engrg., 138 (1996), pp. 273–298.