AIAA JOURNAL
Vol. 34, No. 5, May 1996

Newton–Krylov Methods for Low-Mach-Number Compressible Combustion

D. A. Knoll* and P. R. McHugh†
Idaho National Engineering Laboratory, Idaho Falls, Idaho 83415-3808
and
D. E. Keyes‡
Old Dominion University, Norfolk, Virginia 23529-0162

Fully coupled numerical techniques are used to compute steady-state solutions to a combusting, low-Mach-number compressible flow through a channel. The nonlinear governing equations are discretized on a staggered mesh via integration over discrete finite volumes. The resulting nonlinear algebraic equations are linearized with Newton's method and solved with a preconditioned Krylov algorithm. The selected Krylov solver is the generalized minimum residual algorithm. A matrix-free Newton–Krylov method and a modified Newton–Krylov method are employed as a means of reducing the required number of expensive Jacobian evaluations. The matrix-free implementation is shown to be superior to the modified Newton–Krylov method when starting from a poor initial guess. The technique of mesh sequencing is shown to provide significant CPU savings for fine grid calculations. Additionally, the domain-based multiplicative Schwarz preconditioning strategy was found to be more effective than incomplete lower-upper factorization type preconditioning at lower Mach numbers.

Nomenclature

A	= fuel species
$a_{i,r}$	= stoichiometric coefficients for reactants
B	= oxidizer species
$b_{i,r}$	= stoichiometric coefficients for products
C	= product species
c_p	= dimensionless specific heat capacity at constant pressure, $\bar{c}_p/\bar{c}_{p_0}$
c_v	= dimensionless specific heat capacity at constant volume, $\bar{c}_v/\bar{c}_{v_0}$
D	= dimensionless mass diffusivity, $\bar{D}/\bar{D}_0$
e	= dimensionless internal energy
$\boldsymbol{F}$	= vector of discrete governing equations
f	= individual discrete governing equation
h	= dimensionless enthalpy
i	= species number
J	= Jacobian matrix
kb_r	= backward reaction rate for rth reaction
kf_r	= forward reaction rate for rth reaction
Ma	= Mach number, $\bar{u}_0/\sqrt{[\gamma_0(\bar{R}_g/\overline{Mw_0})\bar{T}_0]}$
Mw	= dimensionless molecular weight, $\overline{Mw}/\overline{Mw_0}$
m	= Jacobian/preconditioner formation interval
N	= system dimension
NR	= number of reactions
NS	= number of species
Pe	= Peclet number, $\bar{\rho}_0\bar{u}_0\bar{c}_{p_0}\bar{L}/\bar{\kappa}_0$
Pr	= Prandtl number, $\bar{\mu}_0\bar{c}_{p_0}/\bar{\kappa}_0$
p	= dimensionless pressure, $\bar{p}/(\bar{\rho}_0\bar{u}_0^2)$
q_r	= dimensionless energy release for rth reaction, $\bar{q}_r/(\overline{Mw_0}\bar{c}_{v_0}\bar{T}_0)$
Re	= Reynolds number, $\bar{\rho}_0\bar{u}_0\bar{L}/\bar{\mu}_0$
$\bar{R}_g$	= universal gas constant
r	= reaction number
Sc	= Schmidt number, $\bar{\mu}_0/\bar{\rho}_0\bar{D}_0$
s	= damping coefficient
T	= dimensionless temperature, $\bar{T}/\bar{T}_0$
u	= dimensionless velocity in x direction, $\bar{u}/\bar{u}_0$
$\boldsymbol{V}$	= diagonal matrix whose entries are cell volumes
v	= dimensionless velocity in y direction, $\bar{v}/\bar{v}_0$
$\boldsymbol{v}$	= general Krylov vector
$\boldsymbol{x}$	= state vector
x, y	= dimensionless Cartesian coordinates, $\bar{x}/\bar{L}$, $\bar{y}/\bar{L}$
Y_i	= mass fraction for ith species, ρ_i/ρ
α_r, β_r	= Arrhenius reaction rate coefficients
γ	= ratio of specific heat capacities, $\bar{c}_{p_0}/\bar{c}_{v_0}$
Δt	= time step
$\delta\boldsymbol{x}$	= Newton's method vector update
ε	= perturbation constant
η	= time step acceleration parameter
κ	= dimensionless thermal conductivity, $\bar{\kappa}/\bar{\kappa}_0$
μ	= dimensionless viscosity coefficient, $\bar{\mu}/\bar{\mu}_0$
ρ	= dimensionless density, $\bar{\rho}/\bar{\rho}_0$
$\dot{\omega}_r$	= dimensionless progress rate for rth reaction, $\bar{\dot{\omega}}_r(\bar{L}\overline{Mw_0})/(\bar{\rho}_0\bar{u}_0)$

Subscripts

i	= species number
r	= reaction number
0	= reference quantity

Superscript

n	= Newton iteration number

Operators

$\bar{}$	= dimensional quantity
T	= transpose
$\lVert\ \rVert_\infty$	= L_∞ norm

Received May 1, 1995; presented as Paper 95-1672 at the AIAA 12th Computational Fluid Dynamics Conference, San Diego, CA, June 19–22, 1995; revision received Dec. 5, 1995; accepted for publication Dec. 17, 1995. Copyright © 1996 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

*Advisory Engineer, Engineering Analysis Department, P.O. Box 1625. Member AIAA.

†Staff Engineer, Engineering Analysis Department, P.O. Box 1625. Member AIAA.

‡Professor, Department of Computer Science; also Senior Research Associate, Institute for Computer Applications in Science and Engineering, NASA Langley Research Center, Hampton, VA 23681-0001. Member AIAA.

Introduction

THE conservation equations describing low-Mach-number compressible flow with combustion contain a wide disparity in time scales and severe nonlinearities. These difficulties make the efficient solution of steady-state problems challenging. To meet this challenge, fully implicit Newton methods have been studied for such problems. A number of Newton methods have been previously studied[1-3] using a variety of linear solution techniques. The most recent of these works[3] used preconditioned Krylov projection methods for the linear solve, but none of these efforts have employed a matrix-free implementation[4,5] of a Newton–Krylov (NK) method.

In many Krylov algorithms the Jacobian (from Newton's method) is required only in the form of matrix-vector products, which can be approximated as follows[4,5]:

$$Jv \approx \frac{F(x + \varepsilon v) - F(x)}{\varepsilon} \tag{1}$$

Recent work using NK algorithms for solving the Navier–Stokes equations has successfully implemented this so-called matrix-free approximation.[6,7] This approximation has also been successfully implemented for the highly nonlinear system of convection-diffusion-reaction equations describing the boundary-layer plasma of magnetic confinement fusion devices.[8]

The matrix-free expression, Eq. (1), approximates the action of the Jacobian without explicitly computing and storing the full Jacobian matrix. This feature is significant, since for combustion-type problems the operations required to form the Jacobian matrix can dominate the overall CPU time. Consequently, use of the matrix-free implementation given in Eq. (1) can potentially lower computation time by avoiding expensive Jacobian evaluations. This implementation enables one to capture the effects of the current Jacobian when performing matrix-vector products, thus retaining the rapid convergence characteristics of Newton's method with infrequent explicit Jacobian evaluations. Typically, however, the Jacobian is needed periodically during the outer Newton iteration to generate an effective preconditioner for the inner Krylov iteration. In this situation, the primary advantage of the matrix-free implementation lies in the ability to amortize the cost of these periodic Jacobian and preconditioner evaluations over many Newton steps without sacrificing the strong convergence characteristics of Newton's method. The main effect, then, is a lagged preconditioner, which may negatively impact the performance of the Krylov algorithm if the preconditioner becomes inadequate. It should be reemphasized that within this matrix-free implementation we do form the Jacobian matrix, but with a reduced frequency. Thus, the Jacobian is available for sensitivity analysis.

An alternative approach for reducing the number of expensive Jacobian evaluations is the use of a modified Newton–Krylov (MNK) algorithm, in which the Jacobian is frozen for a specified number of Newton iterations. In this approach, the Jacobian-vector products within the Krylov iteration are computed in the normal fashion using the frozen Jacobian matrix. Consequently, Newton-like convergence is sacrificed to reduce the CPU cost associated with the Jacobian evaluation. The severity of the sacrifice in nonlinear convergence is then dependent upon the significance of the ignored changes in the Jacobian matrix. The performance of the Krylov algorithm is not impacted by this second approach since the Jacobian matrix and preconditioner are consistent in that they originate from the same Newton step.

In summary, both approaches aim to reduce the required number of expensive Jacobian evaluations. The matrix-free Newton–Krylov (MFNK) algorithm accomplishes this goal via the use of Eq. (1) and infrequent Jacobian evaluations, which are used only to refresh the preconditioner. The negative aspects of this implementation include an increase in the cost of Krylov iterations and the possibility of more Krylov iterations due to the lagged preconditioner. The MNK algorithm accomplishes the aforementioned goal by simply freezing both the Jacobian and the preconditioner for a specified number of Newton steps. The negative aspect of this implementation is the likelihood of a degradation in nonlinear convergence (i.e., more Newton iterations). The effectiveness of the MFNK and the MNK implementations are compared in the computational results presented later.

Critical to the success of the MFNK implementation are relatively low Krylov (linear) iteration counts. This restriction arises because Eq. (1) replaces matrix-vector operations within the Krylov iteration. Because this approximation requires a new residual evaluation, which is an expensive operation, for each matrix-vector product, low inner Krylov iteration counts are essential if the benefits of the matrix-free implementation are to be realized. Two valuable tools for obtaining low inner iteration counts are effective preconditioning strategies and pseudotransient continuation techniques. In this work, both of these techniques are used with the MFNK implementation to reduce computation times.

In this paper we study the application of NK methods to low-Mach-number combustion problems. It is well known that many standard computational fluid dynamics (CFD) algorithms suffer convergence degradation for low-Mach-number flow because of a large spread in time scales.[9-11] It is for this reason that a compressible formulation is employed here instead of a zero-Mach-number approximation. We will show that, with a good choice of preconditioner, an NK algorithm can overcome the severe Courant–Friedrichs–Lewy (CFL) constraints inherent in other compressible flow solution techniques applied to low-Mach-number flows. A dimensionless model problem describing a laminar diffusion flame with three chemical species and one kinetic reaction is presented and solved. We use primitive variables and first-order accurate finite volume discretization on a staggered grid. Solutions to this model problem demonstrate the benefits of a MFNK implementation compared with a MNK method. The potential CPU gains in using a mesh sequencing algorithm are also shown, as is the ability to make a transition to an infinite CFL when using a pseudotransient NK algorithm. Finally, a preliminary comparison of domain-based preconditioning with more standard incomplete lower-upper (ILU) type preconditioning is made.

Chemically Reacting Flow Equations

The nondimensional conservation equations for chemically reacting, compressible, laminar flow with variable transport coefficients in two-dimensional Cartesian coordinates can be expressed by the following:

Continuity for species i

$$\frac{\partial \rho_i}{\partial t} + \frac{\partial}{\partial x}\left[\rho_i u - \frac{\rho D}{ReSc}\frac{\partial Y_i}{\partial x}\right] + \frac{\partial}{\partial y}\left[\rho_i v - \frac{\rho D}{ReSc}\frac{\partial Y_i}{\partial y}\right]$$

$$= Mw_i \sum_{r=1}^{NR}(a_{i,r} - b_{i,r})\dot{\omega}_r \tag{2}$$

x momentum

$$\frac{\partial \rho u}{\partial t} + \frac{\partial \rho u^2}{\partial x} + \frac{\partial \rho uv}{\partial y} = -\frac{\partial p}{\partial x} + \frac{1}{Re}\frac{\partial}{\partial x}\left[2\mu\frac{\partial u}{\partial x} - \frac{2}{3}\mu\right.$$

$$\left.\times\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right)\right] + \frac{1}{Re}\frac{\partial}{\partial y}\left(\mu\frac{\partial u}{\partial y} + \mu\frac{\partial v}{\partial x}\right) \tag{3}$$

y momentum

$$\frac{\partial \rho v}{\partial t} + \frac{\partial \rho uv}{\partial x} + \frac{\partial \rho v^2}{\partial y} = -\frac{\partial p}{\partial y} + \frac{1}{Re}\frac{\partial}{\partial x}\left(\mu\frac{\partial u}{\partial y} + \mu\frac{\partial v}{\partial x}\right)$$

$$+\frac{1}{Re}\frac{\partial}{\partial y}\left[2\mu\frac{\partial v}{\partial y} - \frac{2}{3}\mu\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right)\right] \tag{4}$$

Thermal energy

$$\frac{\partial \rho e}{\partial t} + \frac{\partial}{\partial x}\left[\rho u e - \frac{\gamma\kappa}{Pe}\frac{\partial T}{\partial x} - \frac{\gamma D\rho}{ReSc}\sum_{i=1}^{NS} h_i \frac{\partial Y_i}{\partial x}\right]$$

$$+\frac{\partial}{\partial y}\left[\rho v e - \frac{\gamma\kappa}{Pe}\frac{\partial T}{\partial y} - \frac{\gamma D\rho}{ReSc}\sum_{i=1}^{NS} h_i \frac{\partial Y_i}{\partial y}\right]$$

$$= -\gamma(\gamma - 1)Ma^2 p\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right) + \sum_{r=1}^{NR} q_r \dot{\omega}_r \tag{5}$$

Equation of state

$$p = \frac{T}{\gamma Ma^2}\sum_{i=1}^{NS}\frac{\rho_i}{Mw_i} \tag{6}$$

In these equations, the Stokes condition[12] was assumed, whereas the effects of viscous dissipation and buoyancy were neglected. The species diffusion flux vectors in Eq. (2) are determined assuming an equivalent mass diffusivity for each species, whereas the heat flux vector in Eq. (5) contains contributions from both heat conduction and species diffusion. The fluid density appearing in Eqs. (2–6) is computed from individual species densities according to

$$\rho = \sum_{i=1}^{NS}\rho_i \tag{7}$$

Additionally, the assumptions of a calorically perfect gas enable the following relations to be used:

$$e = c_v T = \frac{T}{\rho}\sum_{i=1}^{NS}\rho_i c_{v_i} \tag{8}$$

$$h = c_p T = \frac{T}{\rho}\sum_{i=1}^{NS}\rho_i c_{p_i} \tag{9}$$

The progress rates appearing in Eqs. (2) and (5) are defined by

$$\dot\omega_r = kf_r\prod_{i=1}^{NS}\left(\frac{\rho_i}{Mw_i}\right)^{a_{i,r}} - kb_r\prod_{i=1}^{NS}\left(\frac{\rho_i}{Mw_i}\right)^{b_{i,r}} \tag{10}$$

For low-Mach-number combustion, the equations outlined earlier exhibit disparate time scales arising from the bulk velocity, sound waves, species diffusion, and kinetic chemistry. The fully implicit solution approach discussed next is used to overcome this difficulty.

Newton–Krylov Methods

The Newton–Raphson method is a powerful technique for solving systems of nonlinear equations of the form

$$\boldsymbol{F}(\boldsymbol{x}) = [f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_N(\boldsymbol{x})]^T = 0 \tag{11}$$

where the vector of state variables $\boldsymbol{x}$ can be expressed as

$$\boldsymbol{x} = [x_1, x_2, \ldots, x_N]^T \tag{12}$$

Application of Newton's method requires the solution of the following linear system on each iteration:

$$\boldsymbol{J}^n\delta\boldsymbol{x}^n = -\boldsymbol{F}(\boldsymbol{x}^n) \tag{13}$$

The (row, column) elements of the Jacobian matrix are defined by

$$J^n_{\text{row,column}} = \frac{\partial f_{\text{row}}}{\partial x^n_{\text{column}}} \tag{14}$$

In this study, these Jacobian terms are computed numerically using finite difference approximations. After solving Eq. (13), the new solution approximation is obtained from

$$\boldsymbol{x}^{n+1} = \boldsymbol{x}^n + s\delta\boldsymbol{x}^n \tag{15}$$

The scalar s is used to damp the update. This iteration is continued until the following convergence criteria are satisfied:

$$\max_i\left[\frac{|\delta x_i^n|}{\max\{|x_i^n|, 1\}}\right] \quad\text{and}\quad \|\boldsymbol{F}(\boldsymbol{x})\|_\infty < 1\times10^{-5} \tag{16}$$

Of interest in this study are steady-state solutions to the governing equations described previously. However, because of the extreme nonlinearities in the chemical source/sink terms and the detailed structure inherent in many combustion solutions, pseudotransient relaxation is often used to increase the radius of convergence of Newton's method. This technique is implemented by replacing Eq. (13) with

$$[(V/\Delta t^n) + \boldsymbol{J}^n]\delta\boldsymbol{x}^n = -\boldsymbol{F}(\boldsymbol{x}^n) \tag{17}$$

thus providing some level of underrelaxation when a steady-state solution is the goal. There are many options for letting Δt^n vary throughout the nonlinear iteration. A commonly used algorithm is switched evolution relaxation (SER).[13] This algorithm relates the growth in time step size to convergence of the residual by

$$\Delta t^n = \Delta t^0\eta\frac{\|F(x^0)\|_\infty}{\|F(x^{n-1})\|_\infty} \tag{18}$$

where η is a tuning parameter. For most problems there will be a limit on the maximum size of the starting time step Δt^0 and a limit on how large one allows Δt^n to grow. The limit on Δt^0 is typically determined by the quality of the initial guess and the nonlinearity of the problem. The nonlinearity is strongly influenced by the exponential dependence of the chemical reactions in chemically reacting flow. The limit on the maximum size of Δt^n (or $CFL_{\max}$) is often determined by the effectiveness of the iterative technique used to solve Eq. (17).

In this study, preconditioned Krylov projection methods are used to solve the linear problem defined by Eq. (17), using an inexact Newton convergence criterion proposed by Averick and Ortega[14] and Dembo et al.[15] This inner iteration convergence criterion is given by

$$\frac{\|\boldsymbol{J}^n\delta\boldsymbol{x}^n + \boldsymbol{F}(\boldsymbol{x}^n)\|}{\|\boldsymbol{F}(\boldsymbol{x}^n)\|} < 1\times10^{-2} \tag{19}$$

Note that if the MFNK implementation is employed, Eq. (1) is used to compute $\boldsymbol{J}^n\delta\boldsymbol{x}^n$ in Eq. (19). Previous research[6] demonstrates that the generalized minimum residual (GMRES) algorithm[16] tends to perform better than other popular Krylov methods when using this matrix-free approximation. Consequently, GMRES(40) is used in all of the calculations described here, where the maximum dimension of the Krylov subspace is fixed at 40. Right preconditioning is employed to improve the performance of the GMRES algorithm. With right preconditioning, Eq. (1) takes the form

$$\boldsymbol{J}(\boldsymbol{P})^{-1}\boldsymbol{v} \approx \frac{\boldsymbol{F}[\boldsymbol{x} + \varepsilon(\boldsymbol{P})^{-1}\boldsymbol{v}] - \boldsymbol{F}(\boldsymbol{x})}{\varepsilon} \tag{20}$$

where $\boldsymbol{P}$ is the preconditioning matrix, which approximates $\boldsymbol{J}$, but is relatively easy to invert. Thus, to obtain $\boldsymbol{P}$ we often must first compute the Jacobian matrix. However, as will be shown, the same preconditioner often can be used for many Newton steps, thereby enabling significant CPU savings. The perturbation constant ε is chosen as follows:

$$\varepsilon = \frac{1}{N\|\boldsymbol{v}\|_2}\sum_{i=1}^{N}\varepsilon_i, \qquad \varepsilon_i = a|x_i| + a \tag{21}$$

where x_i is the ith component of the state vector of dimension N, and a is a constant whose magnitude is 1×10^{-6} in this study.

We restrict our preconditioning choices to those of the ILU factorization type, as well as the domain-based multiplicative Schwarz method using LINPACK banded Gaussian elimination for subdomain solves.[7,8,17–20] In the case of ILU-type preconditioning, a modified level of fill-in idea is used to determine nonzero locations in the LU factors.[18] The domain-based preconditioners offer significant improvements in parallelism compared with the more standard ILU preconditioners.[21] Also, a domain-based preconditioner, with full solves on the subdomains, may be more effective for more ill-conditioned problems. This second property is specifically exploited in this study. A detailed study comparing additive and multiplicative Schwarz methods, approximate subdomain solvers, blocking strategies, and the effects of domain overlap are deferred to another paper.[22]

In applications of Newton's method to the Euler and Navier–Stokes equations, the Jacobian elements are typically derived analytically (see, e.g., Ref. 23). However, this is not an attractive option when considering multispecies, reacting flow simulations. Thus, faced with expensive numerical Jacobian evaluations, the MFNK and the MNK algorithms offer less expensive alternatives. For example, periodic formation of the Jacobian and preconditioning matrices and use of the matrix-free approximation within the Krylov algorithm may enable significant CPU time reduction without sacrificing the strong convergence of Newton's method.

Model Problem and Computational Results

The selected model problem is a nonpremixed, laminar diffusion flame with three chemical species and one reaction. This problem is a modification of the test problem contained in the APACHE code user's manual.[24] The single chemical reaction is expressed as

$$A + 2B \xrightarrow{kf} C \tag{22}$$

where $kb = 0$ and kf is of Arrhenius form, i.e.,

$$kf = \alpha \exp[-\beta/T] \tag{23}$$

The reaction rate in Eq. (23) is defined by $\alpha = 4.0e5$ and $\beta = 7$, whereas the energy release in Eq. (5) is defined by $q = 15$. Additionally, the following parameter definitions are assumed: $Mw_A = 2$, $Mw_B = 2$, $Mw_C = 6$, $c_{v_A} = 1.25$, $c_{v_B} = 1.25$, $c_{v_C} = 0.5$, $c_{p_A} = 1.75$, $c_{p_B} = 1.75$, $c_{p_C} = 0.665$, and $R_g = 1$.

The model problem geometry is shown in Fig. 1 with $L = 25$ and $W = 4.5$. The following boundary conditions are enforced:

Inlet ($x = 0$)

$$\rho_C = 0.0, \qquad T = 1.0 \tag{24}$$

$$\rho_A = \begin{cases} 1.0, & 1.5 < y < 3.0 \\ 0.0, & \text{elsewhere} \end{cases} \tag{25}$$

$$\rho_B = \begin{cases} 0.0, & 1.5 < y < 3.0 \\ 1.0, & \text{elsewhere} \end{cases} \tag{26}$$

$$u = \begin{cases} 1.0, & 1.5 < y < 3.0 \\ 1.5, & \text{elsewhere} \end{cases} \tag{27}$$

Exit ($x = 25$)

$$\frac{\partial u}{\partial x} = 0, \qquad \frac{\partial \rho_i}{\partial x} = 0, \qquad \frac{\partial T}{\partial x} = 0 \tag{28}$$

The wall boundary conditions at $y = 0$ and 4.5 are assumed to be rigid (no slip), adiabatic, and noncatalytic. Values assumed for the nondimensional constants are $Ma = 0.14$, $Re = 30$, $Pr = 0.7$, $\gamma = 1.4$, and $Sc = 0.6$; whereas the temperature-dependent transport coefficients are computed from $D = \mu = \kappa = \sqrt{T}$. Note that the computational cells are naturally ordered, starting from the lower right corner in Fig. 1, sweeping across the channel first and then back towards the inlet. This cell ordering strategy performed best with ILU preconditioning, as compared with other row/column type ordering choices.

Before analyzing algorithm performance, the solution structure of the model problem is illustrated using a uniform 180 × 60 grid. Figure 2 shows contour plots for temperature, Mach number, and pressure, respectively. Notice from Fig. 2a the sharp rise in temperature in the region where the reaction is taking place. The tip of the flame is located at roughly $x = 19.0$ with a peak dimensionless temperature of 5.0. Because of the low Mach number (Fig. 2b), the pressure field is fairly constant as shown in Fig. 2c. Figure 3 shows mass fraction contour plots for the three chemical species. Figure 3a shows that fuel A is almost completely consumed by $x = 12.0$. Similarly, the mass fraction of species B is reduced considerably near the channel outlet as shown in Fig. 3b. In contrast, Fig. 3c shows that although species C mass fraction is zero at the inlet, it is the dominant species at the channel outlet.

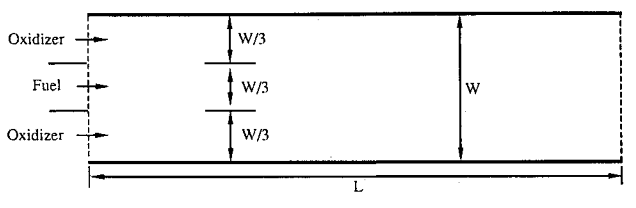

Fig. 1 Schematic of diffusion flame model problem geometry.

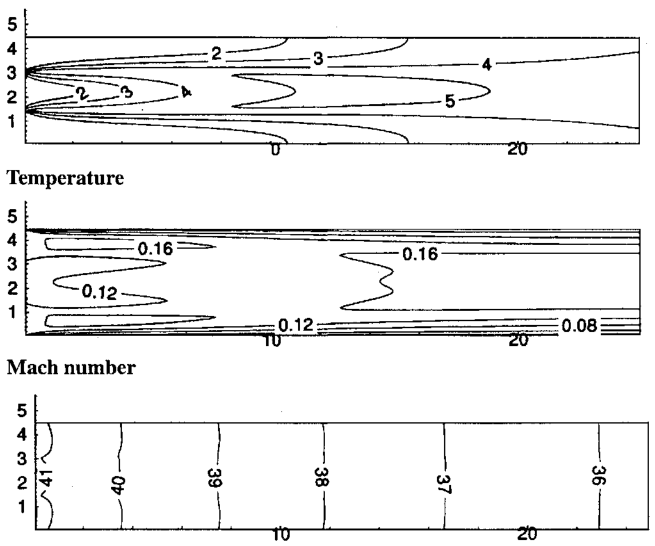

Fig. 2 Temperature, Mach number, and pressure contours for 180 × 60 grid solution.

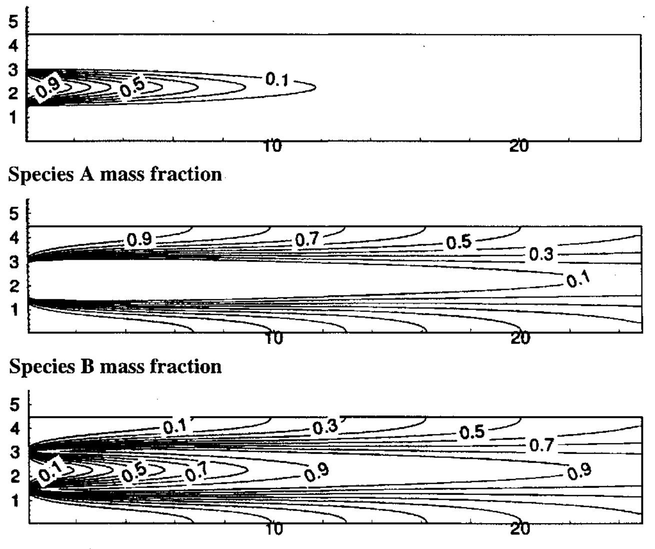

Fig. 3 Species mass fraction contour plots for 180 × 60 grid solution.

This model problem will now be used to investigate the performance of a variety of NK algorithm implementations. On a single grid with a good initial guess, we compare the performance of the various NK implementations. Then we demonstrate the significant CPU savings that can be obtained using mesh sequencing, as compared with starting on the fine grid with a poor initial guess. The single grid MFNK and MNK implementations are compared with a conventional pseudotransient NK implementation (CNK), where the Jacobian and preconditioner are formed on each iteration and Jacobian-vector products are computed in the normal fashion. First, an interpolation of a converged 45 × 15 uniform grid solution is used as an initial guess on a 90 × 30 uniform grid using ILU(0) and ILU(1) preconditioning. Figure 4 shows the convergence history of five different calculations, including the CNK implementation, MFNK and MNK implementations using a fixed time step ($\Delta t = 0.3$), plus MFNK and MNK implementations using the SER algorithm. The fixed time step was chosen to keep the inner iterations per Newton step on the order of 10. This value yields an x-direction Courant

Table 1 Algorithm performance data for six different pseudotransient NK implementations using a 90 × 30 uniform grid

Implementation	Total Newton iterations	Total GMRES iterations	Jacobian plus preconditioner CPU, %	Inner iterations CPU, %	Total CPU, h
CNK, ILU(0), $\Delta t = 0.3, m = 1$[a]	222	1457	87.7	11.7	10.25
MFNK, ILU(0), $\Delta t = 0.3, m = 10$[a]	208	1383	38.3	60.1	2.5
MNK, ILU(0), $\Delta t = 0.3, m = 10$[a]	218	1365	46.7	52.2	2.2
CNK, ILU(0), SER,[b] $m = 1$[a]	26	149	91.0	8.0	2.3
MFNK, ILU(1), SER,[b] $m = 10$[a]	26	223	44.6	52.8	0.58
MNK, ILU(1), SER,[b] $m = 10$[a]	42	192	62.2	35.5	0.7

[a]Jacobian and preconditioner are evaluated every m Newton iterations.
[b]SER: switched evolution algorithm controls time step with $\Delta t^0 = 0.3$ and $\eta = 3.0$.

Table 2 Algorithm performance data for four different pseudotransient NK implementations using a 180 × 60 uniform grid

Implementation	Total Newton iterations	Total GMRES iterations	Jacobian plus preconditioner CPU, %	Inner iterations CPU, %	Total CPU, h
CNK, ILU(0), $\Delta t = 0.3, m = 1$[a]	215	2622	80.9	18.5	43.4
MFNK, ILU(0), $\Delta t = 0.3, m = 10$[a]	215	2609	28.2	71.2	15.1
MFNK, ILU(1), $\Delta t = 0.3, m = 20$[a]	214	1126	37.9	61.4	11.5
MFNK, ILU(1), SER,[b]$m = 10$[a]	23	330	37.0	60.9	2.9
MNK, ILU(1), SER,[b]$m = 10$[a]	26	299	43.6	53.9	2.5

[a]Jacobian and preconditioner are evaluated every m Newton iterations.
[b]SER: switched evolution algorithm controls time step with $\Delta t^0 = 0.1$ and $\eta = 3.0$.

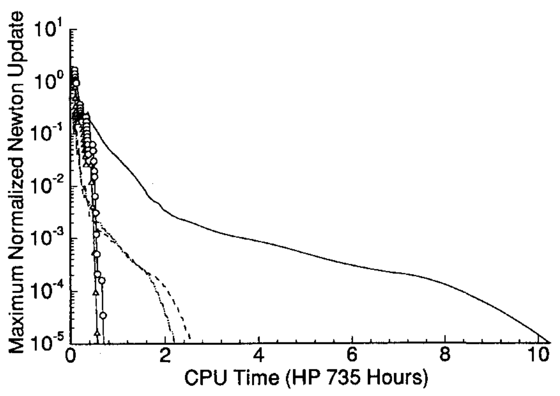

Fig. 4 Convergence history for five different NK solutions on a 90 × 30 grid: ——, CNK ILU(0), d*t* = 0.3; - - - -, MFNK *m* = 10 ILU(0), d*t* = 0.3; —··—, MNK *m* = 10 ILU(0), d*t* = 0.3; —△—, MFNK *m* = 10 ILU(1), SER; and —○—, MNK *m* = 10 ILU(1), SER.

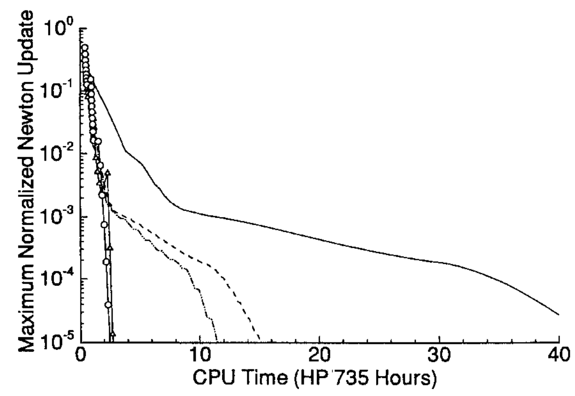

Fig. 5 Convergence history for five different NK solutions on a 180 × 60 grid: ——, CNK ILU(0), d*t* = 0.3; - - - -, MFNK *m* = 10 ILU(0), d*t* = 0.3; —··—, MFNK *m* = 20 ILU(1), d*t* = 0.3; —△—, MFNK *m* = 10 ILU(1), SER; and —○—, MNK *m* = 10 ILU(1), SER.

number (acoustic) of approximately 15. The ability to initiate the solution on this grid with a relatively high CFL is caused by the good initial guess. A value of 10 is used for the Jacobian/preconditioner evaluation interval (m) in the MFNK and MNK implementations. For the SER calculations, $\Delta t^0 = 0.3$ and $\eta = 3.0$ were used in Eq. (18). Performance data for these five calculations, plus an SER calculation using a CNK implementation, are given in Table 1. In the tables, column 4 represents the percentage of the CPU time spent forming the Jacobian and preconditioning matrices, whereas column 5 represents the percentage of CPU time spent performing Krylov iterations. For this problem, the MFNK and MNK implementations performed similarly, each enabling a CPU savings of roughly a factor of 4 as compared with the CNK implementation. The fact that the MNK implementation was competitive with the MFNK implementation for this case is understandable considering the calculation was initiated from an interpolation of a converged solution on a coarser grid. Consequently, one may guess that only minor changes in the Jacobian were required, for which the MNK implementation is well suited. In the CNK implementation, the Jacobian and preconditioner formations required 87.7% of the total CPU time, whereas in the MFNK implementation (second row) they required only 38.3% of the CPU time. However, one difficulty with the MFNK implementation is a shift in the computational workload from the Jacobian/preconditioner formations to the Krylov iterations, which increased from 11.7% of the CPU time for the CNK implementation to 60.1% of the CPU time for the MFNK implementation. This behavior reveals the importance of maintaining low Krylov iteration counts when using the MFNK implementation. The SER algorithm, which enabled an infinite CFL number to be approached, produced additional CPU savings. The algorithm was running with a CFL of approximately 10,000 at the end of the calculation, producing nearly quadratic convergence. The ability to obtain such high CFL numbers is a measure of the robustness of the NK algorithm. Again, the MFNK and MNK implementations required similar CPU times, and both outperformed the CNK implementation by about a factor of 4. The MNK method needed almost twice as many Newton iterations then did the MFNK method. However, since each linear iteration is less expensive in the MNK method, as compared with the MFNK implementation, the CPU cost was similar. In previous research on plasma flow modeling,[25] which also compared the MFNK and MNK implementations, the MNK implementation was not able to use a Jacobian/preconditioner formation frequency of 10 due to an even more serious degradation in nonlinear convergence.

To investigate algorithm performance as a function of problem size, Fig. 5 shows the convergence history of five different calculations using a 180 × 60 uniform grid. An interpolated solution from the 90 × 30 grid was used as an initial guess. The runs include the

Table 3 Matrix-free NK performance data comparing multiplicative Schwarz and ILU(2) preconditioning effectiveness for a low-Mach-number (0.014) combustion problem (90 × 30 uniform grid)

Implementation	Total Newton iterations	Total GMRES iterations	Relative memory requirement[a]	Jacobian plus preconditioner CPU, %[b]	Inner iterations CPU, %	Total CPU, h
ILU(2), $\Delta t = 0.03$	200	418	1	74.1	25.5	5.0[c]
MS 6×3,[d] SER[e]	43	401	1	28.0	70.4	0.9
Full solve, SER[e]	42	85	3	72.5	26.0	1.0

[a]Ratio of preconditioner memory relative to the memory required for the ILU(2) preconditioner (2.9MW).
[b]The Jacobian and preconditioning matrices are evaluated every 10 Newton iterations.
[c]Terminated before final convergence was obtained.
[d]MS: multiplicative Schwarz preconditioner with 6×3 blocking strategy (x cells by y cells).
[e]SER: switched evolution algorithm controls time step with $\Delta t^0 = 0.1$ and $\eta = 3.0$.

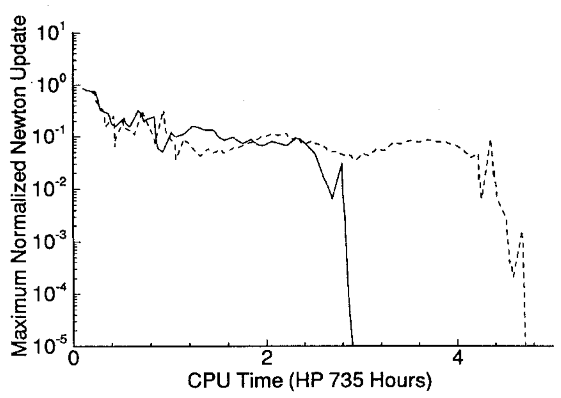

Fig. 6 Convergence history for two different NK solutions on a 90 × 30 grid with a poor initial guess: ——, MFNK $m = 10$ ILU(1), SER and - - - -, MNK $m = 10$ ILU(1), SER.

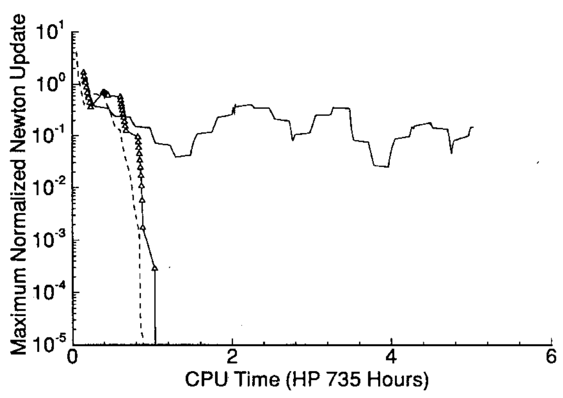

Fig. 7 Convergence history for three different NK solutions on a 90 × 30 grid with the reference Mach number reduced by an order of magnitude: ——, MFNK $m = 10$ ILU(2), $dt = 0.3$; - - - -, MFNK $m = 10$ MS 6×3, SER; and —△—, MFNK $m = 10$ full solve, SER.

CNK implementation, two MFNK implementations with a fixed time step of 0.3, and two SER calculations with $\Delta t^0 = 0.1$ and $\eta = 3.0$, one MFNK and one MNK. The fixed Δt of 0.3 on this grid results in an acoustic, x-direction CFL number of 30. Table 2 presents algorithm performance data for this grid. Comparing Fig. 4 with Fig. 5, and Table 1 with Table 2, the same general trends are observed. MFNK and MNK implementations significantly outperform the CNK implementation. SER allows an even greater CPU savings by evolving to a nearly infinite CFL number, which results in the solution of the steady-state equations directly. Also note that, on average, the required CPU time increased by roughly a factor of 4 compared with the 90×30 grid solutions. This increase represents a linear scaling with the number of unknowns. Likewise, the storage requirements for the ILU preconditioner also scales linearly with problem size. A less favorable scaling would be expected if the number of species were increased rather than the number of grid points. However, the attractiveness of the MFNK implementation increases with the number of governing equations. On this grid, the number of Newton iterations required for the MFNK implementation and the MNK implementation were similar. We believe this is a reflection of the high quality of the first Jacobian matrix yielded by the interpolated 90×30 grid solution. Again, this situation is well suited to the use of the MNK implementation.

Next, we consider the cost of obtaining a solution both with and without the use of mesh sequencing. Figure 6 is a convergence plot for a 90×30 grid solution from a poor initial guess. The performance of the MFNK implementation is compared with the MNK implementation with $m = 10$, $\Delta t^0 = 0.05$, and $\eta = 1.0$. We see that the MFNK implementation is a more CPU efficient algorithm when starting from a poor initial guess. In contrast to the previous calculations, because of the poor initial guess, the Jacobian is forced to undergo significant changes, which the MFNK method is better equipped to capture. This is evidenced by the fact that the MFNK implementation required 147 Newton iterations, whereas the MNK implementation required 342 Newton iterations. The CPU times required for 45×15 and 180×60 grid solutions using the MFNK implementation and $m = 10$ are 0.5 and 40 CPU h, respectively, on an HP-735 platform starting from a similar poor initial guess. Recall that with a good initial guess the SER, MFNK calculations required 0.58 and 2.9 h on the 90×30 and 180×60 grids, respectively. Thus, mesh sequencing provides roughly a factor of 3 speedup on the 90×30 grid, whereas on the 180×60 grid mesh sequencing provides roughly an order of magnitude speedup. This demonstrates the severe cost of developing the global solution structure when starting on a fine grid.

Finally, the effectiveness of domain-based preconditioning is compared with ILU preconditioning. For this study, a Mach number of 0.014 is used. This low value significantly increases the disparity in the time scales associated with the acoustic sound speed and the fluid velocity. Note that the nondimensional pressure scales as Ma^{-2}. This scaling leads to correspondingly large off-diagonal terms in the momentum equations associated with the pressure gradient terms. This results in a larger Jacobian condition number, which in turn negatively impacts the convergence of the Krylov algorithm, making effective preconditioning extremely important. For this problem, a 90×30 grid is employed. The calculation is initiated from an interpolated 45×15 grid solution obtained with an inlet Mach number of 0.14. Figure 7 shows the convergence history for three different MFNK implementations, all of which evaluated the Jacobian and preconditioner every 10th iteration (i.e., $m = 10$). The first implementation used ILU(2) preconditioning and a fixed time step of 0.03, which appeared to be an upper limit for this preconditioning choice. Use of larger CFL numbers resulted in stagnation of the GMRES algorithm. The second implementation used a 6×3 blocking (x blocks by y blocks) strategy with multiplicative Schwarz preconditioning. This preconditioner required the same memory as the ILU(2) preconditioner but allowed the use of SER with an initial time step of 0.1 ($\eta = 3$). The third implementation used a full solve (LINPACK banded Gaussian elimination[20]) as a preconditioner with SER and an initial time step of 0.1 ($\eta = 3$). This preconditioning choice required three times the memory of the previous two. Because of excessive CPU time, the ILU(2) problem was not allowed to run to final convergence. Again, note from Fig. 7 the rapid convergence obtained from the two SER calculations. Table 3 contains the algorithm performance data for these three different calculations. Using the lagged full solve as a preconditioner resulted in the fewest inner GMRES iterations but a slightly higher total CPU time and a significantly larger memory requirement. The ILU(2) preconditioner was found to be inefficient for this low-Mach-number problem. These results show that domain-based preconditioning, with full solves on subdomains, can significantly enhance algorithm robustness. A more detailed study of the benefits of domain-based preconditioning for combustion problems will be published in a separate paper.[22]

Summary and Conclusions

An NK algorithm was used to solve a nonpremixed, laminar diffusion flame model problem with three chemical species and one kinetic reaction. The right preconditioned GMRES(40) algorithm was the selected Krylov solver. MFNK and MNK implementations were studied as a means to reduce the cost associated with forming expensive numerical Jacobians. Both ILU(k) and domain-based multiplicative Schwarz preconditioners were employed. Observations for this model problem, indicated that use of the SER algorithm coupled with an MFNK implementation or an MNK implementation offered significant CPU benefits over a fixed time step, CNK implementation. It was demonstrated that when starting from a poor initial guess, where the Jacobian is forced to undergo significant changes, the MFNK implementation outperformed the MNK implementation. Note that the potential CPU savings of the MFNK algorithm grows with the number of unknowns per control volume since the residual evaluation scales linearly with this number and the Jacobian evaluation scales quadratically with this number. Mesh sequencing was shown to provide significant CPU savings for this steady-state, highly nonlinear, multiple time scale problem. Additionally, the domain-based multiplicative Schwarz preconditioner, with full solves on subdomains, is more effective than ILU preconditioning at low Mach numbers without requiring more computer memory. Consequently, our near-term research on this problem will involve studying subdomain solvers, blocking strategies, and overlap for Schwarz preconditioners.

Acknowledgments

This work was supported under the auspices of the U.S. Department of Energy under DOE Idaho Field Office Contract DE-AC07-94ID13223 and by the Institute for Computer Applications in Science and Engineering under NASA Contract NAS-19480. The authors are indebted to the referees for their many valuable suggestions.

References

[1]Smooke, M. D., Mitchell, R. E., and Keyes, D. E., "Numerical Solution of Two-Dimensional Axisymmetric Laminar Diffusion Flames," *Combustion Science and Technology*, Vol. 67, 1989, pp. 85–122.

[2]Xu, Y., "Numerical Calculations of an Axisymmetric Laminar Diffusion Flame with Detailed and Reduced Reaction Mechanisms," Ph.D. Dissertation, Mechanical Engineering Dept., Yale Univ., New Haven, CT, 1991.

[3]Ern, A., "Vorticity-Velocity Modeling of Chemically Reacting Flows," Ph.D. Dissertation, Mechanical Engineering Dept., Yale Univ., New Haven, CT, 1994.

[4]Brown, P. N., and Hindmarsh, A. C., "Matrix-Free Methods for Stiff Systems of ODE's," *SIAM Journal on Numerical Analysis*, Vol. 23, No. 3, 1986, pp. 610–638.

[5]Brown, P. N., and Saad, Y., "Hybrid Krylov Methods for Nonlinear Systems of Equations," *SIAM Journal on Scientific and Statistical Computing*, Vol. 11, No. 3, 1990, pp. 450–481.

[6]McHugh, P. R., and Knoll, D. A., "Comparison of Standard and Matrix-Free Implementations of Several Newton–Krylov Solvers," *AIAA Journal*, Vol. 32, No. 12, 1994, pp. 2394–2400.

[7]Cai, X. C., Gropp, W. D., Keyes, D. E., and Tidriri, M. D., "Newton–Krylov–Schwarz Methods in CFD," *Numerical Methods for the Navier-Stokes Equations*, edited by F. Hebeker, R. Rannacher, and G. Wittum, Vieweg, Brunswick, Germany, 1994, pp. 17–30.

[8]Knoll, D. A., McHugh, P. R., and Mousseau, V. A., "Newton–Krylov–Schwarz Methods Applied to the Tokamak Edge Plasma Fluid Equations," *Domain-Based Parallelism and Problem Decomposition Methods in Computational Science and Engineering*, edited by D. Keyes, Y. Saad, and D. Truhlar, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1995, pp. 75–95.

[9]Pletcher, R. H., and Chen, K. H., "On Solving the Compressible Navier–Stokes Equations for Unsteady Flows at Very Low Mach Numbers," *Proceedings of the 11th AIAA Computational Fluid Dynamics Conference* (Orlando, FL), AIAA, Washington, DC, 1993, pp. 765–775 (AIAA Paper 93-3368).

[10]Ramshaw, J. D., and Mousseau, V. A., "Damped Artificial Compressibility Method for Steady-State Low-Speed Flow Calculations," *Computers and Fluids Journal*, Vol. 20, No. 2, 1991, pp. 177–186.

[11]Shuen, J. S., Chen, K. H., and Choi, Y., "A Coupled Implicit Method for Chemical Non-Equilibrium Flows at All Speeds," *Journal of Computational Physics*, Vol. 106, No. 2, 1993, pp. 306–318.

[12]White, F. M., *Viscous Fluid Flow*, McGraw–Hill, New York, 1974.

[13]Mulder, W. A., and van Leer, B., "Experiments with Implicit Upwind Methods for the Euler Equations," *Journal of Computational Physics*, Vol. 59, No. 2, 1985, pp. 232–246.

[14]Averick, B. M., and Ortega, J. M., "Solutions of Nonlinear Poisson-Type Equations," *Applied Numerical Mathematics*, Vol. 8, No. 6, 1991, pp. 443–455.

[15]Dembo, R. S., Eisenstat, S. C., and Steihaug, T., "Inexact Newton Methods," *SIAM Journal on Numerical Analysis*, Vol. 19, No. 2, 1982, pp. 400–408.

[16]Saad, Y., and Schultz, M. H., "GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems," *SIAM Journal on Scientific and Statistical Computing*, Vol. 7, No. 7, 1986, pp. 856–869.

[17]Meijerink, J. A., and van der Vorst, H. A., "An Iterative Solution Method for Linear Systems of Which the Coefficient Matrix is a Symmetric M-Matrix," *Mathematics of Computation*, Vol. 31, No. 137, 1977, pp. 148–162.

[18]Watts, J. W., "A Conjugate Gradient-Truncated Direct Method for the Iterative Solution of the Reservoir Simulation Pressure Equation," *Society of Petroleum Engineering Journal*, Vol. 21, June 1981, pp. 345–353.

[19]Dryja, M., and Widlund, O. B., "Domain Decomposition Algorithms with Small Overlap," *SIAM Journal on Scientific and Statistical Computing*, Vol. 15, No. 3, 1994, pp. 604–620.

[20]Dongarra, J. B., Bunch, J., Moler, C., and Stewart, G., *LINPACK User's Guide*, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1979.

[21]Keyes, D. E., "Domain Decomposition Methods for the Parallel Computation of Reacting Flows," *Computer Physics Communication*, Vol. 53, May 1989, pp. 181–200.

[22]McHugh, P. R., Knoll, D. A., and Keyes, D. E., "Schwarz-Preconditioned Newton–Krylov Algorithm for Low Speed Combustion Problems," AIAA Paper 96-0911, Jan. 1996.

[23]Venkatakrishnan, V., "Preconditioned Conjugate Gradient Methods for the Compressible Navier–Stokes Equations," *AIAA Journal*, Vol. 29, No. 7, 1991, pp. 1092–1100.

[24]Ramshaw, J. D., and Dukowicz, J. K., "APACHE: A Generalized-Mesh Eulerian Computer Code for Multicomponent Chemically Reactive Fluid Flow," Los Alamos Scientific Lab., TR LA-7427, Los Alamos, NM, 1979.

[25]Knoll, D. A., and McHugh, P. R., "Newton–Krylov Methods Applied to a System of Convection-Diffusion-Reaction Equations," *Computer Physics Communication*, Vol. 88, Aug. 1995, pp. 141–160.