AIAA JOURNAL
Vol. 46, No. 4, April 2008

Solving Discontinuous Galerkin Formulations of Poisson's Equation Using Geometric and p Multigrid

Brian T. Helenbrook*
Clarkson University, Potsdam, New York 13699-5725
and
H. L. Atkins†
NASA Langley Research Center, Hampton, Virginia 23681-2199

DOI: 10.2514/1.31163

Methods for solving discontinuous Galerkin formulations of the Poisson equation by coupling p multigrid to geometric multigrid are investigated. The simple approach of performing iterative relaxation on solution approximations of decreasing polynomial degree p down to $p = 0$ and then applying geometric multigrid is ineffective. The transition from $p = 1$ to $p = 0$ causes the performance of the entire iteration to degrade because the long wavelength eigenfunctions of the $p = 1$ discontinuous system are not represented well in the $p = 0$ space. A new approach is proposed that coarsens from the $p = 1$ discontinuous space to the $p = 1$ continuous space. This approach eliminates the problems caused by the $p = 1$ to $p = 0$ transition. Furthermore, the $p = 1$ continuous space is a standard finite element space for which geometric multigrid is well-defined. In addition, when the discontinuous Galerkin equations are restricted to a continuous space, one recovers the Galerkin formulation of continuous finite elements. Thus, applying geometric multigrid to this system is straightforward and effective. Numerical tests agree well with the analysis and confirm that the new approach gives rapid convergence rates that are grid-independent and only weakly sensitive to the polynomial order.

Nomenclature

A	=	discrete matrix equations
F	=	vector of discrete source terms
f	=	source function.
h	=	characteristic mesh length normal to the edge
I_{p,p_c}	=	prolongation operator
K	=	segment or quadrilateral element
$L^2(\Omega)$	=	space of square-integrable functions on the domain Ω
N	=	number of elements
n	=	outward normal to the element boundary
n_p	=	number of iterations after prolongation
n_r	=	number of iterations before restriction
$P(K)$	=	space of polynomial functions
$\mathcal{P}_p(K)$	=	space of one-dimensional polynomial functions of degree at most p
p	=	polynomial order
p_g	=	order at which geometric multigrid is applied
R	=	relaxation matrix
$\mathcal{T}$	=	set of segments or quadrilateral elements K that triangulate the spatial domain
u	=	Poisson solution
$\tilde{u}$	=	Fourier vector
$\hat{u}$	=	boundary flux function
u_j	=	solution coefficients for segment j
$u_{j,k}$	=	solution coefficients for element j,k
V	=	scalar test function
x	=	position vector
α_j	=	flux function constant
α_r	=	flux function lifting operator
β	=	local discontinuous Galerkin scheme constant
∂K	=	element boundary
η	=	$\mathcal{O}(1)$ flux function constant
θ	=	nondimensional wave number
ν	=	iteration counter
$\Sigma(K)$	=	$[P(K)]^2$
σ	=	flux vector
$\hat{\sigma}$	=	boundary flux function
τ	=	vector test function
ϕ	=	vector of polynomial basis functions
Ω	=	spatial domain
$\{\}$	=	average of a quantity along the element edge
$[[\,]]$	=	jump in quantity along the element edge

Subscripts

c	=	coarse-grid quantity
h	=	element length
$[i]$	=	quantity associated with the multigrid level i
K	=	element

I. Introduction

HIGH-POLYNOMIAL-ORDER finite element discretizations can be solved using an iterative technique known as p multigrid, where p denotes the polynomial order. This entails relaxing the solution at increasingly coarse (lower polynomial degree) approximations. For example, to solve equations derived using polynomials of order 4, relaxations could be performed on solution approximations of order 4, 2, and then 1. When a low polynomial degree is reached, one can then use standard techniques such as geometric multigrid, generalized minimal residual algorithms (GMRES), or direct solvers to solve the remaining equations. The p component of this algorithm was proposed by Rönquist and Patera [1] and analyzed by Maday and Munoz [2] for a Galerkin spectral element discretization of the Laplace equation. Helenbrook [3] was the first to combine p multigrid with geometric multigrid to solve a continuous high-order finite element formulation of the Navier–Stokes equations.

Presented as Paper 2007-4082 at the 18th AIAA Computational Fluid Dynamics Conference, Miami, FL, 25–28 June 2007; received 20 March 2007; revision received 12 September 2007; accepted for publication 25 November 2007. Copyright © 2007 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved. Copies of this paper may be made for personal or internal use, on condition that the copier pay the $10.00 per-copy fee to the Copyright Clearance Center, Inc., 222 Rosewood Drive, Danvers, MA 01923; include the code 0001-1452/08 $10.00 in correspondence with the CCC.

*Assistant Professor, Mechanical and Aeronautical Engineering Department; helenbrk@clarkson.edu. Member AIAA.
†Senior Research Scientist, Computational Modeling and Simulation Branch, Mail Stop 128; Harold.L.Atkins@nasa.gov. Member AIAA.

We have recently investigated the use of p multigrid to solve discontinuous Galerkin (DG) formulations of the Poisson problem [4–6]. After evaluating many different implementations, we found a few that could provide rapid convergence. All of them reduced p by a factor of 2 after each relaxation until either $p = 1$ or $p = 0$ was reached and then required this low-order problem to be solved. We will show in this paper that the convergence rate of the iteration is strongly affected by the formulation and solution of this low-order problem. Various ways of coupling p multigrid to geometric multigrid will be investigated to obtain fast solutions to the entire problem. Although there has been much work in applying p multigrid to solve DG formulations of various partial differential equations [7–12], none have performed a comprehensive investigation of the possible implementations, nor has anyone recognized the difficulties of coupling to geometric multigrid. Consequently, none have achieved the full potential of the methods.

In the following, we give a brief overview of DG formulations, relaxation schemes, and the p-multigrid algorithm and review some of our previous results to limit the number of cases to be studied. We then examine various ways of coupling to geometric multigrid. Fourier analysis is used to analyze the performance of the various implementations and numerical tests are performed to confirm the results.

II. Formulation

In our previous paper [4], we studied six different DG formulations of the Poisson equation. After a preliminary analysis, that number was reduced to three because many of the schemes have similar properties. Here, we will limit the study to the same three schemes: the local discontinuous Galerkin (LDG) scheme [13] with two different sets of parameters and the Bassi et al. scheme [14]. These will be briefly described here. For more details, the reader can see the preceding references or Arnold et al. [15], who also give a thorough overview.

To formulate the DG schemes, the Poisson equation is written in the following form:

$$\sigma = -\nabla u \tag{1}$$

$$\nabla \cdot \sigma = f(x) \tag{2}$$

where u is the solution to the Poisson problem, x is the position vector, σ is a flux vector, and $f(x)$ is a given source function.

A finite-dimensional space of functions is then introduced to represent the solution. The domain is subdivided into either uniform length elements (1D) or rectangular elements (2D), and on each element, a polynomial basis is used to describe the solution u and the flux vector σ. Following the notation in Arnold et al. [15], we define the following spaces:

$$V_h := \{v \in L^2(\Omega)\colon v|_K \in P(K)\ \forall\ K \in \mathcal{T}_h\} \tag{3}$$

$$\Sigma_h := \{\tau \in [L^2(\Omega)]\colon \tau|_K \in \Sigma(K)\ \forall\ K \in \mathcal{T}_h\} \tag{4}$$

where $L^2(\Omega)$ is the space of square-integrable functions on the domain Ω, $\mathcal{T}_h$ is the set of segments or quadrilateral elements K that triangulate the domain, and subscript h refers to the element length associated with a particular mesh. In one dimension, $P(K) = \mathcal{P}_p(K)$ is the space of polynomial functions of degree at most p on segment K. In two dimensions, $\mathcal{P}_p$ is formed from the tensor product of the one-dimensional set of polynomial functions. $\Sigma(K)$ is equal to $[P(K)]$.

All of the DG formulations analyzed are based on the weak forms of Eqs. (1) and (2). Multiplying these equations by a vector test function $\tau_h \in \Sigma(K)$ and a scalar test function $v_h \in P(K)$, respectively, and integrating by parts gives the weak form that is used to find $u_h \in V_h$ and $\sigma_h \in \Sigma_h$:

$$\int_K \sigma_h \cdot \tau\,\mathrm{d}x = \int_K u_h \nabla \cdot \tau\,\mathrm{d}x - \int_{\partial K} \hat{u}_K n_K \cdot \tau\,\mathrm{d}s \quad \forall\ \tau \in \Sigma(K) \tag{5}$$

Table 1 DG schemes analyzed and their numerical fluxes

Scheme	$\hat{u}_K$	$\hat{\sigma}_K$
LDG [13]	$\{u_h\} - \beta \cdot [[u_h]]$	$\{\sigma_h\} + \beta[[\sigma_h]] - \alpha_j[[u_h]]$
Bassi et al. [14]	$\{u_h\}$	$\{\nabla_h u_h\} - \alpha_r([[u_h]])$

$$-\int_K \sigma_h \cdot \nabla v\,\mathrm{d}x = \int_K f v\,\mathrm{d}x - \int_{\partial K} \hat{\sigma}_K \cdot n_K v\,\mathrm{d}s \quad \forall\ v \in P(K) \tag{6}$$

where n_K is the outward unit normal to the element boundary, and ∂K, $\hat{u}_K$, and $\hat{\sigma}_K$ are boundary flux functions. These functions are evaluated along the element edges using information from both sides of the element and thus provide the interelement coupling in a DG scheme.

The choice of the boundary fluxes distinguishes the various DG schemes [15]. The schemes that are investigated here and their flux functions are listed in Table 1. The notation again follows that of Arnold et al. [15]: Braces { } denote the average of a quantity along an edge. Double brackets [[ ]] denote the jump in a quantity along an edge. For an edge with adjacent elements labeled 1 and 2, the jump in a scalar q is a vector given by $q_1 n_1 + q_2 n_2$, where q_i ($i \in [1, 2]$) is the value of q evaluated along the edge using the solution from adjacent element i, and n_i is the normal to the edge oriented to point outward from element i. If q is a vector, the jump is given by $q_1 \cdot n_1 + q_2 \cdot n_2$. Because all of the DG formulations choose the flux $\hat{u}_K$ to be independent of σ, there is no interelement coupling of σ in Eq. (5), and σ can be locally eliminated from the discrete equations. In the following analysis, we will always work with the discrete equations in which σ has been eliminated as an unknown.

In the LDG scheme, the constant β can take values between $-\frac{1}{2}$ and $\frac{1}{2}$. The constant α_j is given by ηh^{-1}, where η is an $\mathcal{O}(1)$ constant, and h is a characteristic mesh length normal to the edge. We will study two different versions of the LDG scheme. The first takes $\beta = \frac{1}{2}$ and $\eta = 0$. This will be denoted as the LDG one-sided scheme (LDG-OS): the $\hat{u}$ flux for the edge between element j and $j+1$ involves only u_{j+1} and $\hat{\sigma}$ involves only u_j. In our previous paper [4], we showed that this results in a three-element stencil in 1D and that this scheme has some superconvergence properties. However, this scheme is not general because only certain sets of boundary conditions are compatible with the way the fluxes are evaluated. For all of the results presented here, the boundary conditions are periodic, which eliminates this issue. The other version of LDG that we examine is $\beta = 0$ and a baseline value of $\eta = 4$. This will be denoted as LDG centered with penalty (LDG-CP). This scheme has a five-element stencil but is applicable independent of the boundary conditions.

The last scheme is the Bassi et al. [14] scheme, which has a three-element stencil. In this scheme, $\alpha_r([[u_h]])$ is defined by a "lifting operator" [15]; in one dimension, $\alpha_r([[u_h]])$ simplifies to the multiplication of $[[u_h]]$ by a constant that depends on the order of the basis and the length of the element (for more information, see [4] or [15]) and α_r again has an adjustable constant η. We will use this scheme with a baseline value of $\eta = 1$. In some of the analyses, we will vary η to understand the sensitivity to this parameter. In this case, η_0 will denote the baseline value ($\eta_0 = 0$ for the LDG-OS scheme, $\eta_0 = 4$ for the LDG-CP scheme, and $\eta_0 = 1$ for the Bassi et al. [14] scheme).

III. Relaxation Schemes

Before explaining the relaxation schemes, we introduce matrix notation for the discrete equations generated by the DG formulations. The vector of solution coefficients for element j is given by u_j. The solution on this element is represented as $\phi^T u_j$ where ϕ is the vector of polynomial basis functions spanning $P(K)$. In one dimension, the vector of coefficients and basis functions for an element is of length $p+1$. In two dimensions, it is of length $(p+1)^2$. In 2D, we use a multidimensional indexing for the coefficient vectors (i.e., $u_{j,k}$, where j and k are the element's horizontal and vertical mesh indices, respectively).

All of the iterative schemes are written in the form

$$R\Delta u + (Au - F) = 0 \tag{7}$$

where R is a relaxation matrix, u is the vector of unknown coefficients for all elements, Δu is the iterative correction to u, A is the stiffness matrix derived from Eq. (6) after eliminating the σ unknowns using Eq. (5), and F is the vector of source terms obtained from the source function in Eq. (6). The analysis is restricted to periodic problems and algorithms so that Fourier decomposition can be applied. Consequently, both A and R are always constrained to be circulant matrices [16]. This condition is imposed even if the true relaxation scheme of interest does not naturally generate a circulant R matrix.

All of the relaxation schemes investigated are block relaxation schemes. A block is defined as the matrix with rows corresponding to the equations for a particular element and columns for the unknown coefficients of an individual element. Each block is thus of size $(p+1) \times (p+1)$ in 1D or $(p+1)^2 \times (p+1)^2$ in 2D. The first scheme analyzed is the block Jacobi scheme, where R is defined to be the block diagonal matrices of A. Because the block matrices can be obtained from the weighted integral form, the results for this scheme and the following schemes are independent of the polynomial basis ϕ used to represent $P(K)$.

The second scheme investigated is the block Gauss–Seidel scheme. This is similar to the block Jacobi scheme, except that the residual $Au - F$ and the solution are updated element by element. If the elements are ordered left to right then bottom to top, this corresponds to adding to R the block matrices of A that couple the element being updated to the elements to the left and down (essentially, the lower triangle of A). Note that in the Fourier analysis, R is forced to be circulant by adding these block matrices even when the position of the corresponding element has wrapped around the domain due to periodicity. However, the true R matrix for the numerical algorithm is not circulant.

The last two iterative schemes are only used in 2D. The first is a block line solver resulting from the coupled treatment of a horizontal row of elements. The line matrices are block tridiagonal for the LDG one-sided scheme and Bassi et al. [14] scheme, and they are block pentadiagonal for the LDG-CP scheme. The second scheme is similar, except that the line updates are performed sequentially from the bottom row to the top row of the mesh and each line uses the most recent information in its update; that is, it is a line solve in x and Gauss–Seidel in y.

IV. Multigrid

The high-order part of the DG discretization is solved using the p-multigrid algorithm [1,3]. As in a standard multigrid algorithm, restriction, prolongation, and relaxation operators are needed. The restriction operation consists of moving solution residuals from a space of high polynomial order to a lower order. We choose the order of the polynomial spaces such that the coarse space has a degree, $p_c = p/2$. Prolongation is the reverse operation in which the solution correction from the low-order space is transferred to the higher-order space. For a basis ϕ_c of degree p_c that is contained in the space spanned by a higher-order basis, ϕ of degree p, the prolongation operator on an element is given by

$$I_{l,l+1} = \left(\int_K \phi\phi^T \,\mathrm{d}x\right)^{-1} \int_K \phi\phi_c^T \,\mathrm{d}x \tag{8}$$

The subscript l indicates the multigrid level; $l = 0$ corresponds to the finest level. The restriction operator is the transpose of prolongation. When using tensor-product bases in a spatial dimension d, the preceding matrix is of dimension $(p+1)^d \times (p_c+1)^d$ and takes a correction represented using the basis ϕ_c and gives an equivalent representation using the basis ϕ. For this work, we use the Gauss–Lobatto–Lagrange basis, which is not hierarchical. It is a nodal basis with the nodes located at the Gauss–Lobatto points. The prolongation operator for this basis can be found using the preceding approach or, equivalently, by calculating the value of the functions at the nodes of each basis (see [1] for more details). It should be noted that although we only study this one basis, the analytical results apply to all polynomial tensor-product bases.

After restriction to a low value of p, one must solve a system of equations that has a small number of unknowns per element. Our goal is to enable the application of conventional and proven techniques, such as standard geometric multigrid, to solve this low-order system. The value of p at which we apply the conventional technique is denoted as p_g. If $p_g = 0$, there is one unknown per element, and the system is similar to cell-centered finite volume equations. In the following, we investigate different ways of formulating the low-order system and coupling to p multigrid when p_g is equal to 0 or 1.

The p-multigrid and geometric-multigrid algorithms are combined using a linear multigrid algorithm with a V-cycle. This algorithm can be written as a recursive subroutine as follows:

```
cycle($l$)
  If $l$ is the coarsest level $u_{[l]} = A_{[l]}^{-1}(F_{[l]})$
    return
  Relaxation:
    $u_{[l]} = u_{[l]} + R_{[l]}^{-1}(F_{[l]} - A_{[l]}u_{[l]})$    $\nu = 0, \ldots, n_r$
  Restriction:
    $F_{[l+1]} = I_{l,l+1}^T(F_{[l]} - A_{[l]}u_{[l]})$    $u_{[l+1]} = 0$
  Recursion:
    cycle ($l + 1$)
  Prolongation:
    $u_{[l]} = u_{[l]} + I_{l,l+1}u_{[l+1]}$
  Relaxation:
    $u_{[l]} = u_{[l]} + R_{[l]}^{-1}(F_{[l]} - A_{[l]}u_{[l]})$    $\nu = 0, \ldots, n_p$
  return
```

The subscript $[l]$ again indicates the multigrid level, with $l = 0$ corresponding to the finest level, and $u_{[0]}$ is the desired solution to the discrete Poisson equation. For higher values of l, $u_{[l]}$ is a solution correction that will be prolongated to a higher-order space. Similarly, $F_{[0]}$ is the vector of source terms in the governing equations on the finest mesh, and on coarser meshes, it is the restriction of the residual from the previous level. For $l = 0$ to $l = \log_2(p_{[0]}) - p_g$, the algorithm is the p-multigrid algorithm, where $p_{[0]}$ is the value of p used for the simulation. For example, for $p_{[0]} = 4$ and $p_g = 0$, levels 0, 1, 2, and 3 would have $p = 4$, 2, 1, and 0, respectively. For these levels, the prolongation and restriction operators are determined using Eq. (8). For higher levels, the prolongation and restriction operators are geometric-multigrid operators. At the coarsest level, the matrix equations are directly inverted. The constants n_r and n_p determine how many relaxations are performed before restriction and after prolongation, respectively. For all of the results presented here, $n_r = 1$ and $n_p = 0$. Some of the results presented are for a cycle with only two levels. The two-cycle iteration gives an upper bound for the convergence rate because the second-level equations are solved directly rather than iteratively.

For the p-multigrid levels of the iteration, the stiffness matrices $A_{[l]}$ at the coarse levels are determined using an algebraic approach. In this approach, the coarse-level stiffness matrices are evaluated using the restriction and prolongation operators:

$$A_{[l+1]} = I_{l,l+1}^T A_{[l]} I_{l,l+1} \tag{9}$$

In our previous work [4], we showed that this technique is more reliable and usually gives better results than the rediscretization approach in which the coarse matrices are formed by evaluating the DG formulation on the coarser space.

V. Analysis Techniques

For the analysis, we assume that the source function f is zero because the source term does not affect convergence. To determine the convergence rates, we examine the eigenvalues of the multigrid iteration. For a two-level iteration with no source term, one multigrid cycle can be simplified to the following form:

$$u^{[\nu+1]} = \left(I - R_{[0]}^{-1}A_{[0]}\right)^{n_p}\left(I - I_{0,1}A_{[1]}^{-1}I_{0,1}^{T}A_{[0]}\right)\left(I - R_{[0]}^{-1}A_{[0]}\right)^{n_r}u^{[\nu]} \tag{10}$$

where ν is the iteration counter and I is the identity matrix. The spectral radius of the preceding matrix determines the damping factor, which is the amount that the error is decreased each multigrid cycle.

Because the matrices are circulant, the discrete Fourier transform can be used to determine the eigenvalues and thus the spectral radius. In one dimension, we assume the solution on each element has the form

$$u_j = \tilde{u}e^{ij\theta} \tag{11}$$

where $\tilde{u}$ is a vector of dimension $(p+1)$ and θ can take values of $-\pi$ to π by increments of $2\pi/N$, where N is the number of elements. For all of the results presented, N is chosen large enough such the results are essentially continuous functions of θ. Substitution of the form given by Eq. (11) reduces the dimension of the eigenvalue problem to $p+1$. This is then solved numerically for each value of θ. The maximum eigenvalue over the range $-\pi < \theta < \pi$, excluding $\theta = 0$, is the damping factor of the iterative scheme. Zero is excluded because Laplace's equation on a periodic domain is only determined up to a constant. This free constant results in a undamped mode at $\theta = 0$, regardless of the iterative scheme used.

The results for two dimensions are obtained in a similar way. Assuming quadrilateral meshes and a tensor-product basis, Fourier transforms are performed in both directions using $e^{i(j\theta_x+k\theta_y)}$. This reduces the problem to a $(p+1)^2$ eigenvalue problem that must be solved for each combination of θ_x and θ_y in the domain $[-\pi,\pi]^2$. This analysis technique can also be extended to non-tensor-product bases on periodic structured meshes.

VI. Failure of the $p = 0$ Algorithm

The intuitive approach for coupling to geometric multigrid is to set $p_g = 0$ and use geometric-multigrid algorithm designed for a cell-centered finite volume scheme. Unfortunately, the p-multigrid algorithm converges slowly for cycles that include the transition between $p = 1$ and 0. Table 2 shows the 1D damping factors for two-level block Jacobi and block Gauss–Seidel iterations. The dashes for the LDG-OS scheme indicate that this scheme does not depend on η. The important point to notice from this table is that the damping factor for the $p = 1$ to $p = 0$ two-level iteration is closer to one (slower converging) than most of the other entries in the table. This means that the $p = 1$ to $p = 0$ transition will limit the convergence rate of any cycle that coarsens to $p = 0$, regardless of how the $p = 0$ equations are solved.

To demonstrate this, we analyze multilevel cycles for the cases in Table 2, in which the coarsest level in the cycle is $p = 0$ and the $p = 0$ equations are solved directly. Table 3 shows these results. Compared with the two-level iterations, the damping factors are much larger, with many being close to one. In contrast, if we run the $p = 4$ and 8 cases through a multilevel cycle to $p = 1$, we get results that are identical to those shown in Table 2. It is therefore the process of coarsening to $p = 0$ that causes the convergence rates to degrade, and coupling to geometric multigrid at $p = 0$ will not be effective.

We initially suspected that the $p = 1$ to $p = 0$ transition does not work because the $p = 0$ equations are not a consistent discretization of the Poisson problem. Although this is true of the LDG-CP scheme and the Bassi et al. [14] scheme, the LDG-OS scheme produces a standard central-difference formula for a second derivative at $p = 0$. The Bassi et al. scheme can also be forced to be consistent at $p = 0$ by choosing η to be 2 [8]. We tried this value of η and found that the results are no better, and so this is not the cause of the convergence degradation. We also switched back to using the rediscretization method to find the $p = 0$ coarse-grid operators, but this made the iteration diverge. Thus, it is not the consistency of the scheme at $p = 0$ that causes the degradation in performance. We also tried various relaxation schemes and under-relaxation parameters, but this did not eliminate the problem either.

To understand why the transition from $p = 1$ to $p = 0$ converges slowly, we examine the slowest converging eigenfunctions of the

Table 2 One-dimensional damping factors for a two-level block Jacobi iteration between various levels of p

		Jacobi		Gauss–Seidel	
		η/η_0		η/η_0	
Scheme	p	1	4	1	4
LDG-CP	1	0.70	0.89	0.80	0.94
LDG-OS	1	0.60	——	0.75	——
Bassi et al. [14]	1	0.50	0.78	0.55	0.87
LDG-CP	2	0.27	0.09	0.17	0.05
LDG-OS	2	0.00	——	0.00	——
Bassi et al. [14]	2	0.53	0.10	0.34	0.05
LDG-CP	4	0.52	0.20	0.33	0.20
LDG-OS	4	0.00	——	0.00	——
Bassi et al. [14]	4	0.61	0.08	0.43	0.02
LDG-CP	8	0.8	0.49	0.67	0.29
LDG-OS	8	0.00	——	0.00	——
Bassi et al. [14]	8	0.76	0.10	0.63	0.02

Table 3 One-dimensional damping factors for a multilevel iteration to $p = 0$

		Jacobi		Gauss–Seidel	
		η/η_0		η/η_0	
Scheme	p	1	4	1	4
LDG-CP	2	0.989	0.9977	0.90	0.95
LDG-OS	2	0.96	——	0.88	——
Bassi et al. [14]	2	0.85	0.988	0.78	0.94
LDG-CP	4	0.89	0.93	0.95	0.97
LDG-OS	4	0.93	——	0.96	——
Bassi et al. [14]	4	0.87	0.96	0.92	0.98
LDG-CP	8	1.0	1.0	0.98	0.98
LDG-OS	8	0.998	——	0.98	——
Bassi et al. [14]	8	0.995	1.0	0.975	0.993

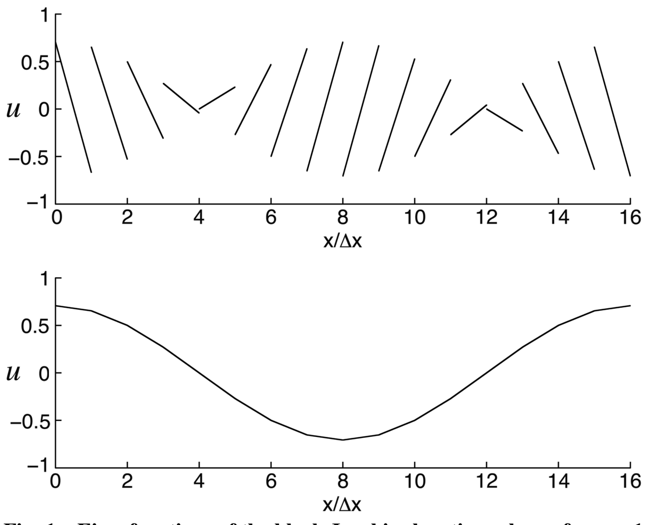

Fig. 1 Eigenfunctions of the block Jacobi relaxation scheme for $p = 1$ at $\theta = 2\pi/16$.

block Jacobi relaxation scheme at $p = 1$. These eigenfunctions must be represented well by the next level of the multigrid cycle if the iteration is to perform well. Figure 1 shows the eigenfunctions for $\theta = 2\pi/16$ and $p = 1$ for the LDG-CP scheme. The eigenfunctions of the other schemes are similar. Because there are two degrees of freedom per element when $p = 1$, there are two eigenfunctions at each value of θ. The eigenfunction in the upper half of the figure is a high-wave-number mode and is damped quickly by the relaxation scheme. The damping factor of this mode is 0.49. The eigenfunction in the lower half of the figure is a small-wave-number mode and is not damped well by the relaxation scheme. Its damping factor is 0.988. This mode must be damped by the next level of multigrid. The important feature to notice about this mode is that it is essentially continuous; at the resolution of the figure, no jumps in the function are visible. Thus, this eigenmode will not be well-represented by a piecewise constant basis.

VII. Discontinuous-to-Continuous Transition

The fact that the long wavelength modes are almost continuous inspired the idea of transitioning from the discontinuous $p = 1$ space to a continuous space of functions. We note that this idea has also very recently been analyzed in [17]. In the discontinuous $p = 1$ space, the Gauss–Lobatto–Lagrange basis consists of the two functions

$$\phi = \begin{bmatrix} (1-\xi)/2 \\ (1+\xi)/2 \end{bmatrix} \tag{12}$$

where ξ is a local coordinate on the element, which goes from -1 to 1 across the element. At the next level of multigrid, we enforce continuity of these functions across element boundaries. Thus, there will be one piecewise-linear continuous function associated with each vertex in the 1D mesh. This is exactly the basis that is typically used in continuous linear finite element formulations. In the continuous space, the number of degrees of freedom per element is one, and so this corresponds to halving the number of degrees of freedom in the approximation space.

If we choose the convention that in the continuous space the left vertex is the degree of freedom associated with each element, the restriction operator for the transition from the $p = 1$ discontinuous-to-continuous space for element k is given by

$$I^T_{c-d,k,k-1} = [0 \quad 1] \qquad I^T_{c-d,k,k} = [1 \quad 0] \qquad I^T_{c-d,k,k+1} = [0 \quad 0] \tag{13}$$

where $I^T_{c-d,k,k-1}$ is the block of the discontinuous-to-continuous restriction operator that multiplies the residuals on element $k-1$ to form the residual in the continuous space on element k. Basically, the restriction operator for element k consists of summing the residual associated with the right vertex function $(1+\xi)/2$ on element $k-1$ with the residual associated with the left vertex function $(1-\xi)/2$ on element k. The prolongation operator is the transpose of this operation. This corresponds to applying the correction found for a vertex in the continuous space to both the left and right vertex functions in the linear discontinuous space.

Table 4 shows multilevel V-cycle results that coarsen to a $p = 1$ continuous space instead of $p = 0$. Examining the block Jacobi results, we see damping factors that are near one or greater than one in some cases. However, the block Gauss–Seidel results are now much better than those shown in Table 3 which indicates that we are on the right track. In fact, all of the convergence results for Gauss–Seidel are identical to the two-level results given in Table 2, except for the case of $p = 1$, for which the convergence rate is much improved over the earlier results. This indicates that the $p = 1$ discontinuous-to-continuous transition does not limit the convergence rate. Now the convergence is limited by the two-level iteration at the highest level of p.

To understand the reason that block Jacobi does not perform well, we examine the eigenvalues of $R^{-1}A$ for the $p = 1$ discontinuous space as a function of θ, shown in Fig. 2. At any value of θ, there are

Table 4 One-dimensional damping factors for a multilevel iteration ending at a continuous $p = 1$ space

		Jacobi		Gauss–Seidel	
		η/η_0		η/η_0	
Scheme	p	1	4	1	4
LDG-CP	1	1.15	1.05	0.14	0.05
LDG-OS	1	0.60	——	0.0	——
Bassi et al. [14]	1	1.0	1.0	0.28	0.11
LDG-CP	2	0.89	0.95	0.17	0.05
LDG-OS	2	0.96	——	0.00	——
Bassi et al. [14]	2	1.0	1.0	0.34	0.05
LDG-CP	4	1.09	1.05	0.33	0.20
LDG-OS	4	0.93	——	0.00	——
Bassi et al. [14]	4	1.0	1.0	0.43	0.02
LDG-CP	8	1.23	1.18	0.67	0.29
LDG-OS	8	1.00	——	0.0	——
Bassi et al. [14]	8	1.0	1.0	0.63	0.02

two values of λ: one of the values corresponds to an eigenmode that is a higher wavelength than the element scale. For this reason, we shifted the large eigenvalue to a wave number of $|\theta - 2\pi|$. This figure shows that the highest-wave-number modes have an eigenvalue of magnitude -2, therefore in a one-step explicit update of the solution, these modes are undamped. To eliminate this problem, we can use an under-relaxation factor of $\frac{2}{3}$ when $p = 1$. For any level of p greater than one, our previous experiments with the block Jacobi and p multigrid [18] showed that the fastest convergence rates are obtained when no under-relaxation factor is used. So in the rest of the results obtained for block Jacobi, an under-relaxation factor of $\frac{2}{3}$ is used when $p = 1$, and no under-relaxation factor is used for higher p. We also reemphasize that none of the results shown in Table 2 improve with the addition of a relaxation factor.

Table 5 shows results for a multilevel algorithm to a continuous $p = 1$ space using block Jacobi in which the $p = 1$ discontinuous relaxation uses an under-relaxation factor of $\frac{2}{3}$. The damping factors are now much smaller and many of the results are the same as the two-level iteration shown in Table 2. Some show a moderate increase in damping factor, especially the LDG-OS scheme, but the damping factor for this scheme was previously identically 0. In any case, block Jacobi now performs well, which confirms that coarsening to the continuous space is effective.

The same approach can be applied in two dimensions. In this case, the discontinuous basis at $p = 1$ is given by the tensor product of the 1D basis. The tensor product results in four bilinear vertex functions that are zero at three of the quadrilateral vertices and one at the other, as shown next for an element with a domain $x \in [-1, 1]$ and $y \in [-1, 1]$:

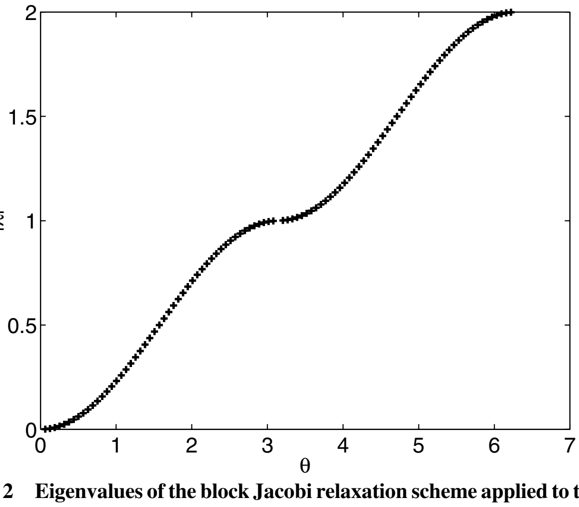

Fig. 2 Eigenvalues of the block Jacobi relaxation scheme applied to the $p = 1$ discontinuous space and the LDG-OS formulation.

Table 5 One-dimensional damping factors for a multilevel block Jacobi iteration ending at a continuous $p = 1$ space and using an under-relaxation factor of $\frac{2}{3}$ for the $p = 1$ discontinuous level

		Jacobi η/η_0	
Scheme	p	1	4
LDG-CP	1	0.43	0.37
LDG-OS	1	0.07	——
Bassi et al. [14]	1	0.33	0.33
LDG-CP	2	0.28	0.31
LDG-OS	2	0.37	——
Bassi et al. [14]	2	0.5	0.33
LDG-CP	4	0.53	0.35
LDG-OS	4	0.28	——
Bassi et al. [14]	4	0.62	0.33
LDG-CP	8	0.8	0.49
LDG-OS	8	0.34	——
Bassi et al. [14]	8	0.77	0.33

$$\phi = \begin{bmatrix} (1-x)(1-y)/4 \\ (1+x)(1-y)/4 \\ (1-x)(1+y)/4 \\ (1+x)(1+y)/4 \end{bmatrix} \tag{14}$$

Enforcing continuity of the solution results in one function per vertex in the quadrilateral mesh. This is again the standard linear continuous finite element space. If a discontinuous $p = 1$ basis is used that does not have this form, the residual and solution must first be projected onto this basis using a transformation similar to Eq. (8).

Table 6 shows block Jacobi and block Gauss–Seidel results for a multilevel iteration to a continuous $p = 1$ space. Two-level results are shown in parentheses. For the block Jacobi case, an under-relaxation factor of $\frac{2}{3}$ is used for the $p = 1$ discontinuous relaxation. The multilevel results are either identical or very close to the two-level results, indicating that the transition to the $p = 1$ continuous space is effective in 2D as well. Although both schemes perform well, the 2D damping factors are not as small and are more sensitive to p than the 1D damping factors. For most cases, there is a significant improvement when adding the Gauss–Seidel terms. Because the cost of these is small relative to the block inversion, block Gauss–Seidel should be more computationally efficient.

Figure 3 shows the dependence of the damping factor on η/η_0 for the LDG-CP scheme and the Bassi et al. scheme [14]. Block Jacobi and block Gauss–Seidel results are shown for a multilevel iteration from $p = 4$ to the $p = 1$ continuous space. For the Bassi et al. scheme, no results are shown below $\eta/\eta_0 = 1$, because the scheme loses stability when η is small. The best performance is achieved around $\eta/\eta_0 = 1$ and performance slowly degrades with increasing

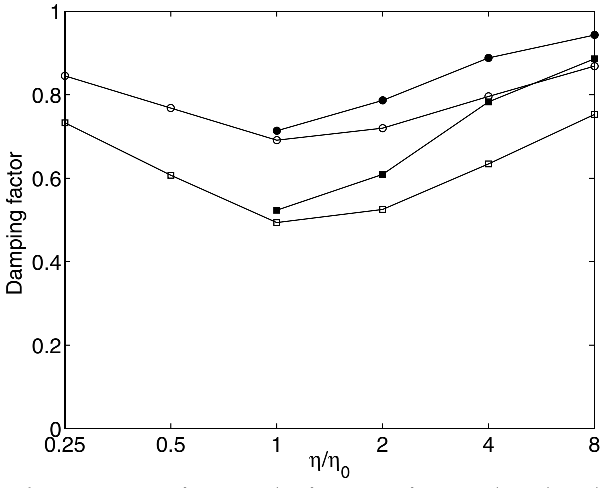

Fig. 3 Dependence of the damping factor on η for a multilevel iteration from $p = 4$ to a $p = 1$ continuous space: LDG-CP scheme (open symbols), Bassi et al. scheme [14] (closed symbols), block Jacobi (circles), and block Gauss–Seidel (squares).

η. In the limit of η approaching infinity, DG solutions should approach that of continuous finite element methods. However, the eigenvalue spectrum of the discrete DG operator does not approach the continuous spectrum, because there are additional eigenvalues associated with the allowed jumps in the solution. These become extremely large as η increases, which increases the stiffness of the problem and thus results in the degradation in performance. In our previous work [4], we investigated the accuracy of the eigenvalues for various DG discretizations of the unsteady heat equation. Although it was not reported, during these investigations we found that the eigenvalues associated with long wavelength modes are more accurate with smaller values of η. Thus, there is no reason to use a large value of η.

The final analysis result is for 2D meshes that have different aspect ratios $\Delta y/\Delta x$. The damping factors of the block Jacobi and block Gauss–Seidel schemes both approach one as the aspect ratio becomes large [18]; therefore, only results for the line relaxation scheme and the line Gauss–Seidel scheme are shown. The lines are oriented in the x direction, which is the compressed direction of the mesh. The iteration is a multilevel iteration to the $p = 1$ continuous space, and the numbers in parentheses are for a two-level iteration. For the line relaxation of the $p = 1$ discontinuous level, an under-relaxation parameter of $\frac{2}{3}$ is used; the entire line is inverted and then the correction added is reduced by a factor of $\frac{2}{3}$. Without this under-relaxation, the damping factors increase significantly. The results shown in Table 7 show that both the line and line Gauss–Seidel relaxation schemes actually converge faster on high-aspect-ratio

Table 6 Two-dimensional damping factors for a multilevel iteration ending at a continuous $p = 1$ space; numbers in parentheses are for a two-level iteration

		Jacobi η/η_0		Gauss–Seidel η/η_0	
Scheme	p	1	4	1	4
LDG-CP	1	0.43	0.58	0.19	0.23
LDG-OS	1	0.53	——	0.17	——
Bassi et al. [14]	1	0.5	0.54	0.26	0.21
LDG-CP	2	0.63 (0.60)	0.81 (0.77)	0.46 (0.46)	0.71(0.71)
LDG-OS	2	0.65 (0.61)	——	0.49 (0.49)	——
Bassi et al. [14]	2	0.62 (0.61)	0.80 (0.76)	0.44 (0.44)	0.69 (0.69)
LDG-CP	4	0.69 (0.69)	0.80 (0.78)	0.50 (0.50)	0.63 (0.61)
LDG-OS	4	0.79 (0.73)	——	0.62 (0.58)	——
Bassi et al. [14]	4	0.71 (0.71)	0.89 (0.83)	0.53 (0.52)	0.78 (0.75)
LDG-CP	8	0.85 (0.85)	0.85 (0.84)	0.73 (0.73)	0.72 (0.64)
LDG-OS	8	0.89 (0.85)	——	0.80 (0.73)	——
Bassi et al. [14]	8	0.83 (0.83)	0.94 (0.92)	0.69 (0.69)	0.90 (0.85)

Table 7 Two-dimensional damping factors for a multilevel iteration ending at a continuous $p = 1$ space; numbers in parentheses are for a two-level iteration

		Line solve $\Delta y/\Delta x$		With sweeping $\Delta y/\Delta x$	
Scheme	p	1	10	1	10
LDG-CP	1	0.56	0.39	0.15	0.15
LDG-OS	1	0.63	0.34	0.23	0.12
Bassi et al. [14]	1	0.56	0.34	0.28	0.25
LDG-CP	2	0.60 (0.55)	0.27 (0.27)	0.40 (0.39)	0.16 (0.16)
LDG-OS	2	0.67 (0.63)	0.33 (0.24)	0.49 (0.49)	0.16 (0.19)
Bassi et al. [14]	2	0.50 (0.50)	0.48 (0.48)	0.34 (0.34)	0.32 (0.32)
LDG-CP	4	0.65 (0.63)	0.51 (0.51)	0.44 (0.42)	0.33 (0.32)
LDG-OS	4	0.78 (0.73)	0.39 (0.24)	0.62 (0.58)	0.26 (0.19)
Bassi et al. [14]	4	0.62 (0.61)	0.62 (0.61)	0.43 (0.42)	0.43 (0.42)
LDG-CP	8	0.81 (0.81)	0.79 (0.79)	0.66 (0.66)	0.66 (0.66)
LDG-OS	8	0.89 (0.85)	0.56 (0.24)	0.80 (0.73)	0.43 (0.19)
Bassi et al. [14]	8	0.80 (0.76)	0.77 (0.76)	0.65 (0.62)	0.62 (0.62)

meshes. Furthermore, the multilevel results are usually close to the two-level results so that the low-order problems are not impacting the convergence rate. Adding the Gauss–Seidel terms to the line relaxation gives a significant improvement in convergence rate. Because these terms are inexpensive relative to the line inversion, the Gauss–Seidel iteration is more computationally efficient.

VIII. Coupling to Geometric Multigrid

Because geometric multigrid is commonly used to solve $p = 1$ continuous finite element discretizations or equivalently finite volume vertex-based discretizations of the Poisson equation, it can be applied with no modification to solve the $p = 1$ continuous problem. The only difference is that we obtain the discrete diffusion operator for the $p = 1$ continuous space using the algebraic approach, and so it may not be the standard discretization. It turns out that this is not the case. Every DG scheme for diffusion when restricted to a $p = 1$ continuous space gives the 1, −2, 1 central-difference stencil in one dimension. This can be seen from Eq. 3.11 in the analysis by Arnold et al. [15], which is the "primal form" of the DG schemes. In the primal form, the variable σ is eliminated. If the solution and the test functions are continuous, the standard Galerkin formulation for diffusion is recovered. Thus, in the continuous space, all of the schemes are guaranteed to reproduce the Galerkin discretization independently of parameters such as η and β, and standard geometric-multigrid techniques for a vertex-based Poisson discretization can be applied.

IX. Evaluation of Alternative *p*-Multigrid Strategies

Before proceeding to the numerical tests, we take a small detour to investigate some alternative p-multigrid strategies. One such alternative is coarsening to $p - 1$ instead of $p/2$. We did not investigate this in our previous paper, and many researchers are choosing this approach [11,12]. Table 8 compares the results obtained from a two-level iteration for $p_c = p - 1$ and $p_c = p/2$. The numbers in parentheses are for $p_c = p/2$. Comparing the results, we see that in most cases, coarsening to $p_c = p - 1$ offers only a small improvement over $p_c = p/2$. In a few cases (for example, Gauss–Seidel, with $\eta/\eta_0 = 1$), the damping factor actually increases marginally, which is difficult to understand. For the $\eta/\eta_0 = 1$ cases with the two generally applicable schemes (LDG-CP and Bassi et al. [14]), coarsening to $p_c = p - 1$ offers an insignificant improvement. Because decreasing by one instead of halving is more computationally expensive, there is no advantage to coarsening to $p - 1$.

Another alternative strategy is to use the relation $p_c + 1 = (p + 1)/2$ [19]. This gives the sequence $p = 7, 3, 1$ instead of $p = 8, 4, 2, 1$. The logic behind this is that in each transition, the total number of unknowns is reduced by $(\frac{1}{2})^d$ because there are $(p + 1)^d$ unknowns per element in a discontinuous formulation. The original sequence was chosen for continuous formulations in which there are p^d unknowns per element. Table 9 shows the two-level iteration results for $p = 3$ with $p_c = 1$ and $p = 7$ with $p_c = 3$. The numbers in parentheses are for $p = 4$ with $p_c = 2$ and $p = 8$ with $p_c = 4$, respectively. There is not a consistent trend between the two results for different schemes and different η, indicating that either approach is functional. For efficiency, the new approach may be better because it reduces the number of multigrid levels needed.

X. Numerical Tests

To confirm that the analytic predictions are correct, we implemented p multigrid with a discontinuous-to-continuous transition at $p = 1$. The p-multigrid algorithm is implemented in an existing multiphysics DG code that supports both triangular and quadrilateral elements in 2D and tetrahedron elements in 3D. To match the analysis, the numerical test presented here makes use of only quadrilateral elements on a unit Cartesian domain. However, the numerical test will still differ from the analysis in several ways. First, all tests that restrict to a continuous space use Dirichlet boundary conditions, due to constraints in the current implementation of the solver for the continuous $p = 1$ formulation. The influence of this boundary condition is evaluated by comparing numerical tests with periodic boundary conditions to numerical tests with Dirichlet

Table 8 Two-dimensional damping factors obtained from a two-level iteration for $p_c = p - 1$ and $p_c = p/2$; numbers in parentheses are for $p_c = p/2$

		Jacobi η/η_0		Gauss–Seidel η/η_0	
Scheme	p	1	4	1	4
LDG-CP	4	0.69 (0.69)	0.65 (0.78)	0.51 (0.50)	0.49 (0.61)
LDG-OS	4	0.66 (0.73)	——	0.47 (0.58)	——
Bassi et al. [14]	4	0.71 (0.71)	0.75 (0.83)	0.56 (0.52)	0.63 (0.75)
LDG-CP	8	0.84 (0.85)	0.72 (0.84)	0.73 (0.73)	0.55 (0.64)
LDG-OS	8	0.76 (0.85)	——	0.60 (0.73)	——
Bassi et al. [14]	8	0.81 (0.83)	0.83 (0.92)	0.70 (0.69)	0.72 (0.85)

Table 9 Two-dimensional damping factors obtained from a two-level iteration for $(p_c + 1) = (p + 1)/2$; numbers in parentheses are for $p = 4$ to 2 and $p = 8$ to 4 transitions

		Jacobi		Gauss–Seidel	
		η/η_0		η/η_0	
Scheme	p	1	4	1	4
LDG-CP	3	0.77 (0.69)	0.88 (0.78)	0.60 (0.50)	0.77 (0.61)
LDG-OS	3	0.75 (0.73)	——	0.67 (0.58)	——
Bassi et al. [14]	3	0.67 (0.71)	0.85 (0.83)	0.46 (0.52)	0.82 (0.75)
LDG-CP	7	0.83 (0.85)	0.87 (0.84)	0.69 (0.73)	0.66 (0.64)
LDG-OS	7	0.85 (0.85)	——	0.74 (0.73)	——
Bassi et al. [14]	7	0.81 (0.83)	0.92 (0.92)	0.66 (0.69)	0.85 (0.85)

boundary conditions for multilevel cycles that terminate with the discontinuous $p = 1$ formulation.

A second difference between the numerical test and the analysis is that the existing code uses a polynomial basis of the form

$$T(p) = \{x^m y^n | m + n \leq p\} \tag{15}$$

instead of the tensor-product basis used in the analysis thus far. Restriction of the $T(1)$ basis to the $p = 1$ continuous space is still defined by Eq. (8), even though $T(1)$ lacks the xy term. The result is equivalent to assuming that the xy term is present but has a zero coefficient. After transferring to the bilinear space, the continuous solution and residual are formed by summing the nodal contributions from the surrounding elements, as described earlier. In cases in which $p > 1$, the xy term is in the $T(p)$ basis and the restriction is well-defined.

The last difference is in the implementation of Gauss–Seidel schemes. The analysis assumes that the R matrix is circulant, whereas the actual matrix generated by the Gauss–Seidel iteration is not. Furthermore, the numerical implementation uses a modified Gauss–Seidel iteration, described in [6], that is considerably less expensive than the true Gauss–Seidel iteration. The modified Gauss–Seidel iteration arises because the numerical code implements DG in the two distinct steps defined by Eqs. (5) and (6), as opposed to directly implementing the primal form. Consequently, the gradient flux σ becomes out of date each time a neighbor is updated. This, in turn, requires that σ be reevaluated in almost all neighboring elements, as well as the local element, before the residual can be properly evaluated and the solution updated. The current Gauss–Seidel implementation strikes a compromise in that the gradient flux is evaluated only twice in each element: once just before the update and again immediately after the update. For this reason, the scheme will be labeled quasi Gauss–Seidel. It should be noted that on a Cartesian grid, the quasi-Gauss–Seidel iteration is equivalent to true Gauss–Seidel for the Bassi et al. [14] discretization and for one particular sweep direction for the LDG one-sided discretization. Previous numerical tests comparing all sweep directions for LDG-OS and LDG-CP [6] showed little difference between the performance of the true-Gauss–Seidel and quasi-Gauss–Seidel iterations. Alternate sweeping patterns, such as a checkerboard pattern, do not encounter this difficulty.

The first set of numerical tests was performed on a series of grids consisting of 16×16, 32×32, and 64×64 elements. The results were found to be mesh-independent between the 32×32 grid and 64×64 grid, and so only the 64×64 results are shown. In all cases, the coarsest level of the multigrid cycle is the $p = 1$ continuous space. Although the $p = 1$ continuous equations could be solved easily with geometric multigrid, for simplicity, we just iterated enough times to reduce the error on this level by two orders of magnitude. This approximates a direct inversion well enough that the results are insensitive to the number of iterations used on the continuous equations. In an effort to fully populate the error spectrum, the initial solution is constructed from the sum of several broadband functions of the form $f(x) = \exp(\cos(\omega x))$. Table 10 gives convergence results for 24 cases. The values in parentheses are from analysis of the corresponding multilevel sequence using the $T(p)$ basis. When performing the tests, we found that a few cases did

Table 10 Two-dimensional damping factors from numerical tests on a 64 × 64 element mesh using the $T(p)$ space of functions; numbers in parentheses are the damping factors as predicted by Fourier analysis of a multilevel iteration to the $p = 1$ continuous space using $T(p)$ functions

		Jacobi		Quasi Gauss–Seidel	
		η/η_0		η/η_0	
Scheme	p	1	4	1	4
LDG-CP	1	0.54 (0.55)	u[a] (0.63)	0.27 (0.22)	u[a] (0.29)
Bassi et al. [14]	1	0.71 (0.50)	0.61 (0.54)	0.39 (0.30)	0.53 (0.22)
LDG-CP	2	0.58 (0.63)	0.77 (0.83)	0.42 (0.50)	0.67 (0.81)
Bassi et al. [14]	2	0.94 (0.62)	0.88 (0.89)	0.89 (0.45)	0.64 (0.82)
LDG-CP	4	0.63 (0.74)	0.78 (0.87)	0.47 (0.50)	0.69 (0.70)
Bassi et al. [14]	4	0.58 (0.70)	0.88 (0.92)	0.50 (0.50)	0.84 (0.85)

[a]The iteration was unstable.

not converge to a constant asymptotic convergence rate; Fig. 4 gives several examples. In some of the rapidly converging cases, the damping factor does not become constant before the iteration reaches machine zero. In other cases, the convergence rate is highly oscillatory and may also have some drift. The table entry gives our best interpretation of the highest rate encountered before the effects of the residual nearing machine zero are evident.

The numerical results generally compare well with the analysis and do not exhibit the slowdown that was seen when restricting to the $p = 0$ discontinuous space. Examining the $p = 4$ cases more closely, all of the results are as good or better than those predicted by the analysis. Results better than the analysis are possible because the initial condition may not have fully populated the space of error modes, or the residual may approach machine zero before the slowest mode dominates the convergence. For $p = 2$, all results are again as good or better than the analysis, except the $\eta/\eta_0 = 1$ Bassi et al. [14]

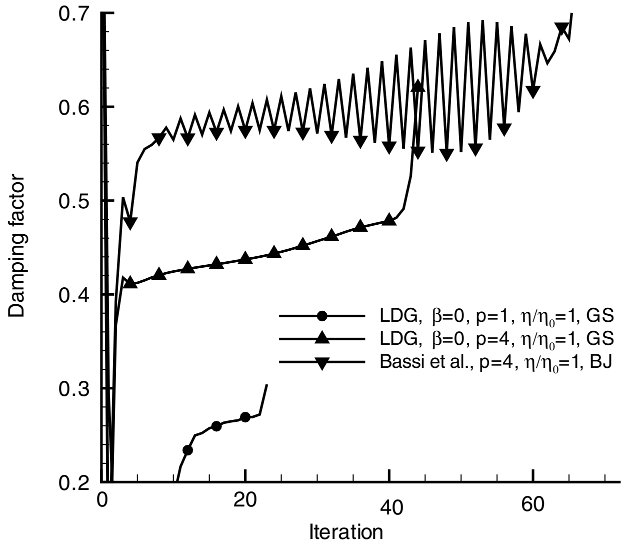

Fig. 4 Convergence rates typical of cases that do not have clear asymptotes.

Table 11 Two-dimensional damping factors for a p-multigrid cycle terminating at the $p = 1$ discontinuous space; all results are obtained with the Bassi et al. [14] scheme using $p = 2$

		Jacobi		Quasi Gauss–Seidel	
		η/η_0		η/η_0	
Boundary conditions	Method	1	4	1	4
Dirichlet	Numerical	0.94	0.89	0.89	0.82
Periodic	Numerical	0.55	0.88	0.89	0.82
Periodic	Two-level analysis	0.62	0.89	0.45	0.82

case when using either block Jacobi or Gauss–Seidel. The reason for this discrepancy is most likely the difference in boundary conditions between the analysis and the tests. This will be established subsequently by examining two-level results to a $p = 1$ discontinuous space using periodic boundary conditions. For $p = 1$, several of the results are slower than the analysis, and the LDG-CP results with $\eta/\eta_0 = 4$ are both unstable. This difference may again be because of boundary conditions, but we cannot verify this using two-level results, because restricting to the $p = 0$ discontinuous space is known to converge much slower.

Table 11 gives results using both Dirichlet and periodic boundary conditions for the Bassi et al. scheme [14], in which the p-multigrid cycle was terminated at the $p = 1$ discontinuous space. The block Jacobi result with periodic boundary conditions and $\eta/\eta_0 = 1$ now has a much smaller damping factor, in agreement with the analysis. For Gauss–Seidel, however, the results are still much larger than those predicted by the analysis. This difference may be because the Gauss–Seidel iteration is nonperiodic even though periodic boundary conditions are applied to the steady formulation. In spite of the few differences between the results and the analysis, all of the iterations perform well; thus, restricting to a continuous space is effective.

XI. Conclusions

We developed a new method of coupling p multigrid to geometric multigrid that involves transitioning from a $p = 1$ discontinuous function space to a $p = 1$ continuous function space. For all of the DG schemes, the equations for the continuous space correspond to a vertex-based finite volume scheme or, equivalently, to a piecewise-linear continuous finite element scheme; that is, they are second-order central-difference stencils, which can be easily solved using geometric multigrid. Using this technique, the convergence rates for a multilevel iteration are essentially the same as those achieved in a two-level iteration, showing that the low-order equations do not limit the convergence rate. This method converges more rapidly than the more intuitive approach of coarsening to $p = 0$ and then applying geometric multigrid for cell-centered finite volume equations. Before this, the practical applicability of p multigrid for DG discretizations was limited because there was no effective way of solving the low-order problem.

References

[1] Rönquist, E. M., and Patera, A. T., "Spectral Element Multigrid, Part 1: Formulation and Numerical Results," *Journal of Scientific Computing*, Vol. 2, No. 4, 1987, pp. 389–406.
doi:10.1007/BF01061297

[2] Maday, Y., and Munoz, R., "Spectral Element Multigrid, Part 2: Theoretical Justification," Inst. for Computer Applications in Science and Engineering, TR 88-73, Hampton, VA, 1988.

[3] Helenbrook, B. T., "A Two-Fluid Spectral Element Method," *Computer Methods in Applied Mechanics and Engineering*, Vol. 191, Nos. 3–5, Nov. 2001, pp. 273–294.

[4] Helenbrook, B. T., and Atkins, H. L., "Application of 'p' Multigrid to Discontinuous Galerkin Formulations of the Poisson Equation," *AIAA Journal*, Vol. 44, No. 3, Mar. 2005, pp. 566–575.

[5] Helenbrook, B. T., and Atkins, H. L., "Application of P Multigrid to Discontinuous Galerkin Formulations of the Poisson Equation," 17th AIAA Computational Fluid Dynamics Conference, Toronto, CA, AIAA Paper 2005-5111, June 2005.

[6] Atkins, H. L., and Helenbrook, B. T., "Numerical Evaluation of P Multigrid Method for the Solution of Discontinuous Galerkin Discretizations of Diffusive Equations," 17th AIAA Computational Fluid Dynamics Conference, Toronto, CA, AIAA Paper 2005-5110, June 2005.

[7] Fidkowski, K. J., and Darmofal, D. L., "Development of a Higher-Order Solver for Aerodynamic Applications," *42nd AIAA Aerospace Sciences Meeting and Exhibit*, AIAA, Reston, VA, 2004, pp. 695–705; also AIAA Paper 2004-436, 2004.

[8] Fidkowski, K. J., Oliver, T. A., Lu, J., and Darmofal, D. L., "p Multigrid Solution of High-Order Discontinuous Galerkin Discretizations of the Compressible Navier–Stokes Equations," *Journal of Computational Physics*, Vol. 207, No. 1, July 2005, pp. 92–113.
doi:10.1016/j.jcp.2005.01.005

[9] Nastase, C. R., and Mavriplis, D. J., "High-Order Discontinuous Galerkin Methods Using a Spectral Multigrid Approach," 43rd AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, AIAA Paper 2005-1268, Jan. 2005.

[10] Luo, H., Baum, J. D., and Lohner, R., "A p Multigrid Discontinuous Galerkin Method for the Euler Equations on Unstructured Grids," *Journal of Computational Physics*, Vol. 211, No. 2, 2006, pp. 767–783.
doi:10.1016/j.jcp.2005.06.019

[11] Nastase, C. R., and Mavriplis, D. J., "Discontinuous Galerkin Methods Using an HP Multigrid Solver for Inviscid Compressible Flows on Three-Dimensional Unstructured Meshes," 44th AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, AIAA Paper 2006-107, Jan. 2006.

[12] Luo, H., Baum, J. D., and Lohner, R., "A Fast p Multigrid Discontinuous Galerkin Method for Compressible Flows at All Speeds," 44th AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, AIAA Paper 2006-110, Jan. 2006.

[13] Cockburn, B., and Shu, C.-W., "The Local Discontinuous Galerkin Method for Time-Dependent Convection-Diffusion Systems," *SIAM Journal on Numerical Analysis*, Vol. 35, No. 6, 1998, pp. 2440–2463.
doi:10.1137/S0036142997316712

[14] Bassi, F., Rebay, S., Mariotti, G., Pedinotti, S., and Savini, M., "A High-Order Accurate Discontinuous Finite Element Method for Inviscid and Viscous Turbomachinery Flows," *Proceedings of the 2nd European Conference on Turbomachinery, Fluid Dynamics, and Thermodynamics*, edited by R. Decuypere, and G. Dibelius, Technologisch Inst., Antwerp, Belgium, 1997, pp. 99–108.

[15] Arnold, D. N., Brezzi, F., Cockburn, B., and Marini, L. D., "Unified Analysis of Discontinuous Galerkin Methods for Elliptic Problems," *SIAM Journal on Numerical Analysis*, Vol. 39, No. 5, 2002, pp. 1749–1779.
doi:10.1137/S0036142901384162

[16] Davis, P. J., *Circulant Matrices*, 2nd ed., Chelsea, New York, 1994.

[17] Brix, K., Pinto, M. C., and Dahmen, W., "A Multilevel Preconditioner for the Interior Penalty Discontinuous Galerkin Method," Inst. für Geometrie und Praktische Mathematik, RWTH Aachen Univ., Aachen, Germany, July 2007.

[18] Helenbrook, B. T., "Preconditioning for Incompressible Flows with Free-Surfaces and Two-Fluid Interfaces," *Journal of Computational Physics*, Vol. 207, No. 1, 2005, pp. 282–308.
doi:10.1016/j.jcp.2005.01.012

[19] Den Abeele, K. V., Broeckhoven, T., and Lacor, C., "Dispersion and Dissipation Properties of the 1D Spectral Volume Method and Application to a P-Multigrid Algorithm," *Journal of Computational Physics*, Vol. 224, No. 2, June 2007, pp. 616–636.
doi:10.1016/j.jcp.2006.10.022

Z. Wang
Associate Editor